2022

# Approximating Bayesian Optimal Sequential Designs using Gaussian Process Models Indexed on Belief States

Joseph Burris
*Virginia Commonwealth University*

# DISSERTATION APPROXIMATING BAYESIAN OPTIMAL SEQUENTIAL DESIGNS USING GAUSSIAN PROCESS MODELS INDEXED ON BELIEF STATES

A Dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at Virginia Commonwealth University.

by

JOSEPH BURRIS

B.S., The University of Maryland, U.S.A. - September 2011 to December 2014

M.S., Virginia Commonwealth University - August 2015 to May 2017

Director: Dissertation,

Dr. David Edwards, Department of Statistical Sciences and Operations Research

Virginia Commonwealth University

Richmond, Virginia

August, 2022

**Acknowledgements**

I would first like to thank my brother Daniel for being my best friend and constant companion. Second, I would like to thank the rest of my family: Mom, Dad, and Leah for your constant support. Finally, I would like to thank Dr. Edwards for guiding me though my Graduate and Doctoral education. Without your DOE II course, I wouldn't have pursued an interest in this field. Without your help with the NASA assistantship this Ph.D. probably wouldn't have happened at all.

## TABLE OF CONTENTS

Chapter Page

## LIST OF TABLES

## LIST OF FIGURES

**Abstract**

DISSERTATION APPROXIMATING BAYESIAN OPTIMAL SEQUENTIAL DESIGNS USING GAUSSIAN PROCESS MODELS INDEXED ON BELIEF STATES

By Joseph Burris

A Dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at Virginia Commonwealth University.

Virginia Commonwealth University, 2022.

Director: Dissertation,

Dr. David Edwards, Department of Statistical Sciences and Operations Research

Fully sequential optimal Bayesian experimentation can offer greater utility than both traditional Bayesian designs and greedy sequential methods, but practically cannot be solved due to numerical complexity and continuous outcome spaces. Approximate solutions can be found via approximate dynamic programming, but rely on surrogate models of the expected utility at each trial of the experiment with hand-chosen features or use methods which ignore the underlying geometry of the space of probability distributions. We propose the use of Gaussian process models indexed on the belief states visited in experimentation to provide utility-agnostic surrogate models for approximating Bayesian optimal sequential designs which require no feature engineering. This novel methodology for approximating Bayesian optimal

sequential designs is then applied to conjugate models and to particle approximations for different batch sizes.

# CHAPTER 1

# INTRODUCTION

Experimental design always relies on some prior knowledge of the process of interest in order to choose both possible models and the design region for variables of interest. The Bayesian philosophy extends this idea by formalizing the use of prior information into the mathematics itself via a prior distribution. Beliefs about the unknown parameters $\theta$ are encoded in a probability distribution known as the prior distribution $p(\theta)$. The experimenter's beliefs about the parameters change after having observed data $y$, and the data is assumed to follow a known likelihood given the unknown parameters $p(y \mid \theta)$. The experimenter's beliefs about $\theta$ can then be updated using Bayes' Theorem:

$$p(\theta \mid y) = \frac{p(y, \theta)}{p(y)} = \frac{p(y \mid \theta)\, p(\theta)}{\int_{\Theta} p(y \mid \theta)\, p(\theta)\, d\theta}$$

Like most problems in the Bayesian paradigm, deriving Bayesian optimal designs algebraically is extremely difficult, if not impossible, except in special cases making numerical methods the most common choice [1]. Bayesian optimal designs optimize the expected utility gained over all parameter values and outcomes. Most commonly, Bayesian optimal designs use utility functions based solely on the posterior distribution to measure information about model parameters. However, utility functions

are flexible enough to measure other quantities of interest such as the number of successful treatments of patients or total costs of experimentation.

Traditionally, Bayesian optimal designs are static in that the choice of design points is fixed at the beginning of the experiment which allows for trials to be randomized. However, design points can also be chosen sequentially. Sequential experimentation, also known as adaptive experimentation, is commonly considered in medical research [2]. While a large randomized trial may be more robust, sequences of experiments can allow for later patients to benefit from information gained from earlier trials. Consider the following example, an experimenter is comparing three treatments with prior probabilities of success defined below.

$$\theta_1 \sim Beta\,(1, 2)$$
$$\theta_2 \sim Beta\,(45, 35)$$
$$\theta_3 \sim Beta\,(3, 3)$$

We are interested in maximizing the number of successes within a fixed cohort, but want to balance that with the number of successes in a wider population once the best performing treatment is identified. For this example, we will consider a cohort of eight patients and measure the broader success by the expected success rate of the best performing treatment multiplied by 40. Most commonly, sequential experimentation is performed in a myopic fashion: individual experiments are performed in sequence without knowledge of the experiments that will follow due

to computational convenience. The Beta distribution is conjugate to the Bernoulli likelihood. For $\theta \sim Beta(\alpha, \beta)$ with outcome $y$, we have

$$p(\theta|y = \text{Failure}) \sim Beta(\alpha, \beta + 1)$$

$$p(\theta|y = \text{Success}) \sim Beta(\alpha + 1, \beta)$$

Since $E[\theta] = \frac{\alpha}{\alpha+\beta}$, no number of failures within the eight trials will have $E[\theta_2]$ fall less than $E[\theta_1]$ or $E[\theta_3]$. This yields an optimal myopic design with an expected utility of $(40 + 8) \cdot \frac{45}{(45+35)} = 27$. Alternatively, we can consider all possible outcomes of all future trials choices for each trial of the experiment. This yields the design illustrated in Figure 1. The color of each cell denotes the optimal treatment choice at that point of the experiment. Below each cell are two more cells denoting a success or failure of the last trial with cell widths proportional to the likelihood of observing that outcome given the current prior. A cell to the left of its parent cell is chosen if a failure was observed while a cell to the right of its parent is should be chosen if a success was observed. Cell width is proportional to the probability of arriving in that cell. For example, choosing treatment 3 is optimal for trial 1. Since $\theta_3 \sim Beta(3, 3)$, both success and failure are equally likely outcomes resulting in the cells for trial 2 having equal width. If trial 1 results in a success, we proceed to the right side shaded by circles. In this case, treatment 3 is still optimal. If failure is observed for trial 1, instead we proceed to the left shaded by crosses. In this case, treatment 1 is the optimal next choice given we have less confidence in treatment 3. This proceeds like so until after the final trial where the best treatment is administered to

40 more experimental units. Compared to the myopic case, the full Bayesian optimal sequential design (BOSD) has expected utility of 28.407 compared to 27.

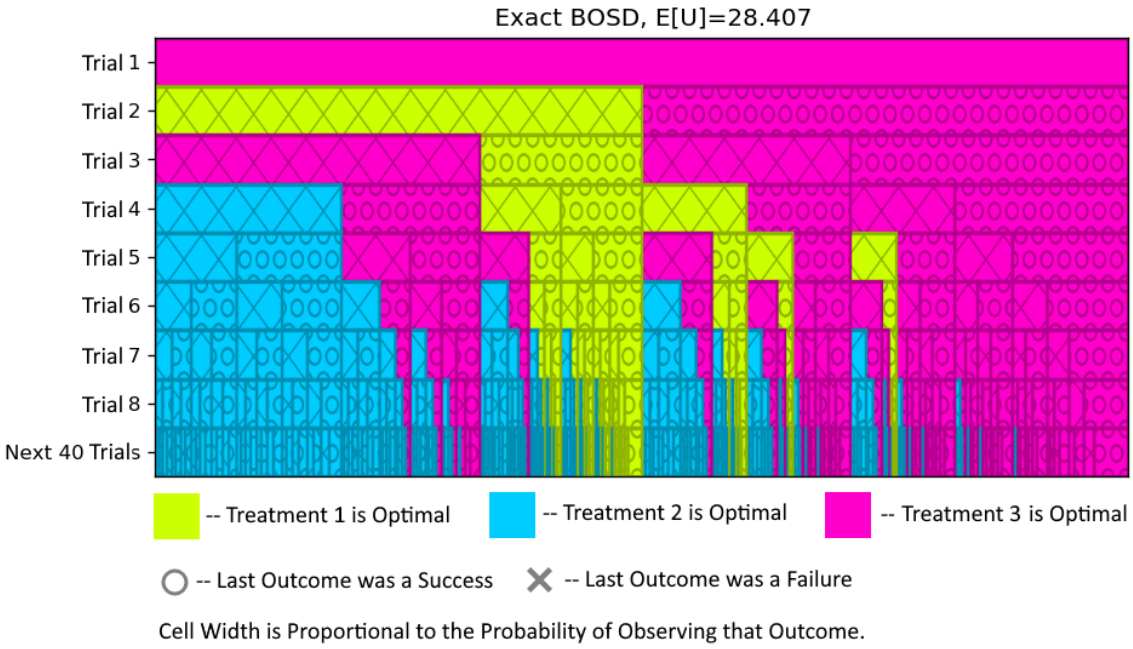


Fig. 1. Exact BOSD Calculated via Backward Induction.

Practical calculation of BOSDs has been held back by its computational complexity until recently except in special cases [3]. For example, optimizing the Kullback–Leibler divergence from the prior to the posterior is a very common utility. This same concept is used in the derivation of the classic D-optimality criterion from frequentist statistics [1]. For the case of most BOSDs for linear models under the KL-divergence utility, the static optimal design has global optimality eliminating the need for sequential experimentation given a fixed design space [1]. In cases with a finite set of outcomes, the BOSD can be simply calculated exactly and represented similar

to a flowchart (e.g., Figure 1) though the size of the problem may make it computationally prohibitive. The finite outcome has can be extended to an infinite trial case with diminishing returns of the outcome. This is equivalent to the well explored multi-armed bandit problem and has simpler asymptotic solutions in addition to other heuristic solutions which perform well [4, 5]. However, in most cases, the BOSD can neither be expressed as a static design nor even as a flowchart. Instead, the BOSD is a sequence of functions dictating which design point to choose for any possible outcome given all previous outcomes and decisions. In these cases, approximations must be used. The field of approximating solutions to these types problems is called approximate dynamic programming (ADP). ADP uses surrogate models to approximate the functions which dictate the optimal design points without actually performing the optimization needed to get an exact value [6]. The application of ADP to BOSD was first explored in Huan and Marzouk's 2016 paper on the subject [7]. While this paper laid the groundwork, its methods predict the utility for each trial of the experiment using statistics of the possible posterior distributions as features for a regression model. However, improper choice of features can result in the algorithm converging to a suboptimal solution. Recent advances utilize Deep Neural Networks to choose design points based on the sequence of design points and outcomes alone [8, 9, 10]. While these methods can made fast predictions due to the lack of intermediate calculations of the posterior, they cannot take advantage of the underlying structure of the space of reachable probability distributions. The method proposed in this dissertation uses the statistical distance between different posterior distributions to allow for a utility-agnostic approach which can be applied to prob-

lems regardless of utility function and takes advantage of the underlying geometry of the set of reachable distributions. Chapter 2 gives background on the BOSD problem and the approximate dynamic programming algorithm. In Chapter 3, we outline how to apply Gaussian processes to find approximately optimal Bayesian sequential designs for models with conjugate priors. In Chapter 4, we introduce a more complex methodology which can apply to a broader class of models at the cost of computation time. In Chapter 5, we apply this method to cases where multiple trials are independently performed in a sequence of batches. For this dissertation, we will assume the only element of the experiment state is the belief state though the methods presented easily extend to the broader case. Chapter 6 recaps the presented methodology and discusses future work.

# CHAPTER 2

# BACKGROUND

Beliefs about the unknown parameters $\theta$ are encoded in a probability distribution known as the prior distribution $p(\theta)$. The experimenter's beliefs about the parameters change after having observed data $y$. For parameters $\theta$, possible outcomes $y$, design $\xi$, the static Bayesian optimal design problem for an experiment with $T$ trials has the following form:

$$\xi^* = \arg\max_{\xi \in \chi^T} \int_Y \int_\Theta \omega(\xi, \boldsymbol{y}, \theta)\, p(\theta \mid \boldsymbol{y}, \xi)\, p(\boldsymbol{y} \mid \xi)\, d\theta dy \tag{2.1}$$

Here, $\omega(\xi, y, \theta)$ is the utility function, $p(\boldsymbol{y} \mid \xi)$ is the marginal probability of observing outcome vector $\boldsymbol{y}$ under design $\xi$, $p(\theta \mid \boldsymbol{y}, \xi)$ is the posterior probability given vector $\boldsymbol{y}$ under design $\xi$, and $\chi$ is the design space of a single trial which can have any number of factors. More generally, the utility, $u$, can be thought of as a functional of the posterior pdf $p_{\theta|\boldsymbol{y},\xi}$.

$$\xi^* = \arg\max_{\xi \in \chi^T} \int_Y u\left(\xi, \boldsymbol{y}, p_{\theta|\boldsymbol{y},\xi}\right) p(\boldsymbol{y} \mid \xi)\, dy \tag{2.2}$$

Bayesian optimal designs optimize the expected utility gained over all parameter values and outcomes. For example, optimizing the Kullback–Leibler divergence from the prior to the posterior is expressed in Equation 2.3.

$$\xi^* = \arg\max_{\xi \in \chi^T} \int_Y \left[ \int_\Theta \log\left( \frac{p\left(\theta|\boldsymbol{y}, \xi\right)}{p\left(\theta\right)} \right) p\left(\theta|\boldsymbol{y}, \xi\right) d\theta \right] p\left(\boldsymbol{y} \mid \xi\right) dy \tag{2.3}$$

More about traditional Bayesian optimal design can be found in Chaloner and Verdinelli's seminal review article on the subject[1]. As opposed to a single static design, a sequential design is optimized by an optimal policy. This policy dictates the actions of the experimenter at each trial given the previous design points chosen and previous responses observed. Bellman defines the optimal policy as follows:

> "An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision." [11]

This is known as Bellman's Principle of Optimality, a fundamental result in Dynamic Programming [11]. Bellman's Principle leads to the Bellman Equations for the Bayesian sequential design problem.

BOSDs fall under the umbrella of Markov Decision Processes (MDPs). MDPs are defined over a state space $S$ and action space $A$ and have the Markov property that the probability of transitioning from state $s$ to state $s'$ is determined only by the current state $s$ and the chosen action $a$. The state for a simple BOSD is exactly the current prior distribution which we will call the belief state. Note that while by and large belief state transitions are performed via Bayes' Rule, belief state transitions could just as easily fit other probabilistic frameworks such as the Dempster-Shafer calculus [12]. BOSDs can be considered a specific type of Partially Observable MDP

(POMDP). POMDPs extend the MDP to include uncertainty about the current state of the process [13]. At each transition, an observation is observed causing the belief state to change. In general, POMDPs have applications in AI and control theory. For example, in a quality control application, the states of machines on an assembly line can transition from being functional to nonfunctional. Belief about the state of the machines transitions based on observations from samples taken off the assembly line instead of direct observations. For cases where for any given state, the set of possible outcomes is isomorphic to the set of reachable states, the POMDP reduces to an MDP [14]. BOSDs have this exact property since we are never interested in a particular realization of $\theta$, only its distribution. As BOSDs are a subset of POMDPs even though they reduce to MDPs, algorithms to solve for optimal POMDP policies can be applied to the BOSD problem. In particular, Thurn used a nearest-neighbor approach using KL-Divergence and MCMC sampling to approximate the value function of an infinite-horizon stationary POMDP [15]. The methodology used in this paper similarly uses a distance-based model of the belief state.

In general, most MDP literature assumes stationarity and an infinite horizon. The infinite-horizon, stationary MDP is the backbone of reinforcement learning and has a vast corpus of literature outside the Bayesian experiment design context. In the Bayesian experimental design context, the infinite-horizon, stationary case is well explored in particular for medical research. The problem of assigning independent treatments is equivalent to the well explored multi-armed bandit problem and has simpler asymptotic solutions in addition to other heuristic solutions which perform well [4, 5]. Optimal stopping problems are also well explored with a focus on de-

veloping algorithms which perform better computationally than backward induction [16, 17]. Nasrollahzadeh and Khademi give a review of current optimal stopping algorithms for adaptive dose-finding trials [18].

This manuscript focuses on BOSDs with a fixed number of trials. In this case, the BOSD is non-stationary and has a finite-horizon. BOSDs are finite-horizon in that the experiment has a fixed number of trials and are non-stationary in that the reward function depends on the trial. For example, when maximizing for information gain, only the KL-Divergence at the final belief state is judged. Examination of this problem from a MDP perspective is fairly recent with Huan and Marzouk in 2016 developing a methodology using linear surrogate models. Since then, multiple amortized approaches which formulate optimal policies which map a history of designs and outcomes to a new design point using deep neural networks. Foster et al. and Blau et al. use the Prior Contrastive Estimation bound to develop neural networks which choose optimal design points for maximizing information gain without having to evaluate intermediate posteriors [9, 10]. Shen and Huan take a policy-gradient approach which uses gradient descent to train a neural network policy for maximizing information gain which is significantly faster than Huan and Marzouk's previous approximate dynamic programming approach [8]. Note Shen and Huan's method extends to other utility functions just as well.

The methodology presented in this paper uses Gaussian process surrogate models indexed on the belief states themselves and thus requires evaluating the belief state at each stage of the experiment. While slower in most cases than recent methods which avoid calculating intermediate belief states, use of the belief states directly

allows for surrogates to take advantage of relationships between belief states whose design point-outcome vectors are different, but have similar belief states regardless.

All information about the experiment at trial $t$ is encoded in the experiment state $x_t$. While in general, for a BOSD, $x_t$ can contain information beyond the current belief state such as running costs, the vast majority of applications have $x_t$ as the belief state exactly. As mentioned before, this dissertation will assume $x_t$ is the belief state exactly though the methods presented can be easily extended to cover the broader case. At each trial $x_t$ transitions to state $x_{t+1}$ via Markov transition function $\mathcal{F}_t(x_t, y_t, \xi_t)$. The belief component of the experiment state propagates via Bayes' Rule.

Let $u_t(x_t, y_t, \xi_t)$ denote the immediate utility gained from trial $t$. Note $u_t$ can be different for each trial. For example, $u_t$ might be zero for all $t \neq T$ with $u_T(x_T)$ measuring the all information gained about the parameters at the final trial. In this case, the final utility is all that matters— there is no immediate utility gain from the intermediary trials. Denote the total expected utility at trial $t$ for state $x_t$ to be $U_t(x_t, y_t, \xi_t)$. $U_t(x_t, y_t, \xi_t)$ denotes the expected utility we can gain from future experiments given we make all optimal choices from there on out. Equation 2.4 follows [7].

$$\begin{aligned} U_t(x_t) = \max_{\xi_t \in \chi_t} \int_{y_t} & p(y_t \mid x_t, \xi_t) \\ & \left[u_t(x_t, y_t, \xi_t) + U_{t+1}(\mathcal{F}_t(x_t, y_t, \xi_t))\right] dy_t, \ \forall t = 1, 2, \ldots, T \qquad (2.4) \\ U_{T+1}(x_{T+1}) = u_{T+1}(x_{T+1}) & \end{aligned}$$

Each trial's expected utility is a combination of the immediate utility gain $u_t(x_t, y_t, \xi_t)$ and the expected utility gain from all future trials $U_{t+1}(\mathcal{F}_t(x_t, y_t, \xi_t))$. Since we assume the experiment state is the belief state exactly, the transition function is simply Bayes' Rule. BOSD problems can be solved via *Backward Induction* for a finite set of outcomes, design points, and states since all possible sequences can be enumerated (see Algorithm 1) [19].

**Algorithm 1** Backward Induction

**for** $x_{T+1} \in \mathcal{X}_{T+1}$ **do**
    $U_{T+1}(x_{T+1}) \leftarrow u_{T+1}(x_{T+1})$
**end for**
**for** $t = T, \ldots, 1$ **do**
    **for** $x_t \in \mathcal{X}_t$ **do**
        $U_t(x_t) \leftarrow \max_{\xi_t \in \chi_k} \sum_{y_t \in \mathcal{Y}_t} p(y_t|x_t, \xi) \cdot [u_t(x_t, y_t, \xi_t) + U_{t+1}(\mathcal{F}_t(x_t, y_t, \xi_t))]$
    **end for**
**end for**

Backward Induction is best visualized as a decision tree. The deepest nodes of the tree are all possible final designs. The utility of these designs are calculated. Their parent nodes then use this information to decide the best course of action given all possible outcomes. This process is repeated backwards all the way up to the root of the tree. The optimal policy can be traced by following the decision tree back down given the outcomes observed. Backward Induction suffers the "Curse of Dimensionality." Because the size of $\mathcal{X}_{T+1}$, the set of all terminal decisions, grows exponentially with $T$, the total number of calculations is also exponential with $T$.

Since the problem is so complex, approximation can be used instead of direct computation. The Bellman equations at each trial can be approximated to generate the following myopic design problem [7]:

$$\tilde{U}_t(x_t) = \max_{\xi_t \in \chi_t} \int_y p(y_t \mid x_t, \xi_t) \cdot \left[ u_t(x_t, y_t, \xi_t) + \tilde{U}_{t+1}(\mathcal{F}_t(x_t, y_t, \xi_t)) \right] dy_t, \ \forall t = 1, 2, \ldots, T$$

where $\tilde{U}_{t+1}(\mathcal{F}_t(x_t, y_t, \xi_t))$ is some approximation to $U_{t+1}(\mathcal{F}_t(x_t, y_t, \xi_t))$. For good enough approximations at each trial, a policy close to optimality can be reached. In particular, BOSDs with continuous likelihoods can be implemented under this scheme. As mentioned previously, this type of approach is called approximate dynamic programming and has its own vast corpus of literature outside the statistical paradigm.

Approximate dynamic programming problems can be solved by an iterative algorithm with two basic steps: *simulation* and *approximate value iteration*. The *simulation* step simulates experiments sequentially by choosing a design point, simulating an experimental trial, and repeating until the experiment is finished. Design points can be chosen using two different methods: *exploration* and *exploitation*. *Exploration* selects each new design point using heuristics. For example, one heuristic might randomly select each new design point [7] while another might use Latin hypercube sampling to ensure a more even coverage across the design space. Heuristic solutions such as myopic designs can also be used to explore the design region. The

chosen heuristic is called the *exploration policy* and is denoted $\pi^{\text{explore}}$. Note that since the utility is exactly known for the final trial of the experiment, $\pi^{\text{explore}}$ is simply optimizing the exact expected utility. $\pi^{\text{explore}}$ is generally a function of the number of value iterations with the proportion of explorations to exploitations tending to zero to guarantee convergence. On the other hand, *exploitation* selects the most optimal design point based on the approximate expected utility, $E\left[\tilde{U}_t\right]$. Since the approximate utility functional is built using the results from previous iterations, we are *exploiting* the information we have learned so far during the ADP algorithm. Each full experiment simulation is called an experimental *trajectory*.

Last is the *approximation* step where the design points from the *simulation* step are used to develop the approximate expected utility functional. Ideally, the approximate utility functional is both fast and accurate. However, these are often two competing goals. For example, polynomial regression models based on the mean and log-variance of the current belief state have been used to approximate utility functionals [7]. These features were chosen based on the analytical expression for KL-divergence between two normal distributions. However, while the exact utility is known to be highly correlated with these features, the same guarantee cannot be made for earlier trials in the experiment. Furthermore, If the chosen features produce biased estimates during the approximation step, the algorithm will solve a problem that is too far removed from the original optimal design problem. On a broader scale, feature engineering remains a topic of interest in many fields including statistics and machine learning and has been explored extensively. Even still, feature engineering

remains as much an art as it a science. Even with expert knowledge of the data, models must be evaluated and refined over multiple iterations via trial and error [20].

**Algorithm 2** Solving the sOED problem using Approximate Dynamic Programming [7].

1: **Set parameters:** Select number of experiments $T$, features $\{\boldsymbol{\phi}_t\}_{t=1}^{T}$, exploration policy $\pi^{\text{explore}}$, number of policy updates $N$, number of exploration trajectories $M^{\text{explore}}$, number of exploitation trajectories $M^{\text{exploit}}$. Denote the total number of experimental trajectories $M := M^{\text{explore}} + M^{\text{exploit}}$.

2: **for** $i = 1, \ldots, N$ **do**.

3: **Simulation:** *Exploration*— Simulate $M^{\text{explore}}$ exploration trajectories by sampling $\theta_t$ from the current belief state $x_t$, $\xi_t$ from exploration policy $\pi^{\text{explore}}$, and $y_t$ from the likelihood $p\left(y_t \mid \xi_t, \theta_t\right)$. Transition to $x_{t+1} \leftarrow \mathcal{F}_t\left(x_t, y_t, \xi_t\right)$ Repeat for $t = 0, \ldots, T$.

4: Store all posterior belief states visited in $\mathcal{X}^i_{t,\text{explore}} = \left\{x_t^j\right\}_{j=1}^{M^{\text{explore}}}$, $t = 1, \ldots, T+1$

5: *Exploitation*— If $i > 1$, simulate $M^{\text{exploit}}$ exploitation trajectories by sampling $\theta_t$ from the current belief state $x_t$, calculating

$$\xi_t \leftarrow \arg\max_{\xi_t^* \in \chi_t} \int_y p\left(y_t | x_t, \xi\right) \left[u_t\left(x_t, y_t, \xi_t^*\right) + \tilde{U}_{t+1}^{i-1}\left(\mathcal{F}_t\left(x_t, y_t, \xi_t^*\right)\right)\right] dy_t$$

and sample $y_t$ from the likelihood $p\left(y_t \mid \xi_t, \theta_t\right)$. Transition to $x_{t+1} \leftarrow \mathcal{F}_t\left(x_t, y_t, \xi_t\right)$ Repeat for $t = 0, \ldots, T$.

6: Store all posterior belief states visited in $\mathcal{X}^i_{t,\text{exploit}} = \left\{x_t^j\right\}_{j=1}^{M^{\text{exploit}}}$, $t = 1, \ldots, T+1$

7: **Approximate value iteration:** Construct functions $\tilde{U}_t^i$ via backward induction using new regression points $\left\{\mathcal{X}_{t,\text{explore}}^i \cup \mathcal{X}_{t,\text{exploit}}^i\right\}, t = 1, \ldots, T$ using the method below

8: Fit $\tilde{U}_{T+1}^i$ by using features $\boldsymbol{\phi}_T$ calculated from $\left\{x_{T+1}^j, U_{T+1}^i\left(x_{T+1}^j\right)\right\}_{j=1}^M$

9: **for** $t = T, \ldots, 1$ **do**

10: **for** $j = 1, \ldots, M$ where $x_t^j$ are the sampled states from $\left\{\mathcal{X}_{t,\text{explore}}^i \cup \mathcal{X}_{t,\text{exploit}}^l\right\}$ **do**

11: Compute

$$\hat{U}_t^i\left(x_t^j\right) = \max_{\xi_t^* \in \chi_t} \int_y p\left(y_t | x_t, \xi_t^*\right) \left[u_t\left(x_t^j, y_t, \xi_t\right) + \tilde{U}_{t+1}^{i-1}\left(\mathcal{F}_t\left(x_t^j, y_t, \xi_t'\right)\right)\right] dy_t$$

12: Then fit $\tilde{U}_t^i$ by using features $\boldsymbol{\phi}_t$ calculated from $\left\{x_t^j, \hat{U}_t^i\left(x_t^j\right)\right\}_{j=1}^M$

13: **end for**

14: **end for**

15: **end for**

16: **Return final policy parameterization:** $\tilde{U}_t^N$, $t = 1, \ldots, T$

Consider again an experiment with overall utility being the difference in Kullback–Leibler divergence between prior at trial 1 and posterior at trial $T$. Since the overall utility is a functional of the final posterior exclusively, for some trial $t$ of the experiment, the final achieved utility of the experiment depends only upon on the current belief state $x_t$, subsequent design points $\{\xi_s\}_{s=t+1}^T$, and subsequent outcomes $\{y_s\}_{s=t+1}^T$. For an optimal sequential design, we will always choose optimal design points no matter the outcome observed. This leaves the expected utility

$E\left[U_{t+1}\left(\mathcal{F}_t\left(x_t, y_t, \xi_t\right)\right)\right]$ given a certain outcome and design point as a functional of the posterior exclusively. This same logic applies to any utility functional which only depends on the final posterior. This includes most common utility functionals used in Bayesian optimal experimental design including mutual information between models, quadratic loss, and Shannon information gain [3]. It follows that the most complete model possible only needs to be a functional of the posterior belief state. This paper addresses the potential problems caused due to inaccurate features by implementing distance-based approximations based on the current belief state.

# CHAPTER 3

# APPROXIMATE VALUE ITERATION USING DISTANCE-BASED MODELING WITH CONJUGATE PRIORS

Consider the set of belief states visited during the *simulation* at step $t$: $\left\{x_t^j\right\}_{j=1}^M$. For two similar belief states $x_t^0$ and $x_t^1$, we expect their corresponding expected utilities $U\left(x_t^0\right)$ and $U\left(x_t^1\right)$ to also be similar. When we visit a new belief state $x_t^*$ in the next iteration of the ADP algorithm, we could impute $U\left(x_t^*\right)$ based on how similar $x_t^*$ is to each belief state in $\left\{x_t^j\right\}_{j=1}^M$. This concept is known as distance-based modeling and is the backbone of many statistical and machine learning models including $k$-means clustering, regression trees, and kernel regression [21, 22]. Using a distance-based model eliminates the need for feature engineering since the entire belief state is one feature. However, in order to develop a distance-based model for belief states, we need to be able to compute the similarity between the belief state of interest $x_t^*$ and the previously visited belief states $\left\{x_t^j\right\}_{j=1}^M$. This is done by developing a positive-definite *kernel* based on a chosen measure of distance between probability distributions.

## 3.1 Positive Definite Kernels for Probability Distributions

The locations on which the distance-based model fit is called where it is *indexed*. For example, most Gaussian processes in spatial statistics applications are indexed pairs of latitude and longitude. Correlation between two points is defined based on

the distance between them along the curvature of the Earth. We aim to build a distance-based model indexed on the belief states visited during the experimental process. To do this, the distance between two belief states must be defined. While these are several ways to measure distance between probability distributions, we focus on covariant metrics since they are computationally simple and do not depend on the geometry of the parameter space. Note that modeling functionals of distributions via kernel regression is not novel. Distribution regression has a deep history in machine learning though typical applications only have access to empirical distributions observed from real world data [23].

### 3.1.1 Covariant Metrics

Covariant metrics define the distance between two distributions by comparing their probability density functions. While there many different ways to compare two probability densities, only the ones which are conditionally positive definite and symmetric are useful to us. Specifically, a real valued function $k$ on $\mathcal{X} \times \mathcal{X}$ is conditionally positive definite if and only if $k$ is symmetric and $\sum_{i,j}^{n} c_i c_j k\left(x_i, x_j\right) \geq 0$ for all $n \in \mathbb{N}$, $x_i \in \mathcal{X}$, $i = 1, \ldots, n$, and $c_i \in \mathbb{R}$, $i = 1, \ldots, n$, with $\sum_{i}^{n} c_i = 0$ [24]. Conditionally positive definite kernels can be used to construct positive definite kernels using two methods [24, 25]:

$$K\left(P, Q\right) \coloneqq \frac{1}{2}\left[-D^2\left(p, q\right) + D^2\left(p, 0\right) + D^2\left(0, q\right)\right] \tag{3.1}$$

and

$$K(P,Q) \coloneqq \exp\left[-t \cdot D^2(p,q)\right] \tag{3.2}$$

Here, $D^2$ is our distance function comparing probability densities $p$ and $q$ belonging to random variables $P$ and $Q$, and $t$ is a positive constant. Equation 3.1 adjusts the kernel by "centering" it around a measure with zero probability. On the other hand, Equation 3.2 exponentiates the distance guaranteeing positive definiteness and fixes $K$ between 0 and 1. This makes Equation 3.2 a natural choice when modeling correlation. Additionally, the Schur Product Theorem combined with other basic properties of positive definite matrices guarantees that for any positive definite matrix $H$, matrix $H^*$ defined as

$$H^*_{ij} = a_1 H^1_{ij} + a_2 H^2_{ij} + \cdots + a_k H^k_{ij} \tag{3.3}$$

is also positive definite if $a_1, a_2, \ldots, a_k > 0$ for any $k \in \mathbb{Z}^+$ [26]. This leads to a flexible family of covariance matrices even if only a single distance measure is available. A covariance function completely specified by the distance between locations is called *isotropic.* Isotropic covariance is a common assumption made in spatial modeling and is usually assumed unless diagnostics illustrate otherwise [27].

Consider measures $P$ and $\mu$ on $\mathcal{X}$. Measure $\mu$ dominates measure $P$ if and only if for any set $x \in \mathcal{X}$, $\mu(x) \geq p(x)$. The existence of $\mu$ is trivial for most cases; for example, all probability distributions on $\mathbb{R}^n$ with finite probability are dominated by the Lebesgue measure which leads to "normal" integration [24]. However, some distributions are not dominated by the Lebesgue measure such as the Dirac mixture

distributions used in Section 4. Given two probability measures $P$ and $Q$ dominated by measure $\mu$ on $\mathcal{X}$, the kernel

$$D^2_{\alpha|\beta}(P,Q) = \int_{\mathcal{X}} d^2_{\alpha|\beta}(p(x), q(x))\, d\mu(x) \tag{3.4}$$

is conditionally positive definite and symmetric, and is independent of dominating measure $\mu$ given $d^2_{\alpha|\beta}$ belongs to the family defined in Equation 3.5 below [24].

$$d^2_{\alpha|\beta}(p(x), q(x)) = \frac{2^{\frac{1}{\beta}}\left(p(x)^{\alpha} + q(x)^{\alpha}\right)^{\frac{1}{\alpha}} - 2^{\frac{1}{\alpha}}\left(p(x)^{\beta} + q(x)^{\beta}\right)^{\frac{1}{\beta}}}{2^{\frac{1}{\beta}} - 2^{\frac{1}{\alpha}}} \tag{3.5}$$

for $\alpha \in [1, \infty]$, $\beta \in [-\infty, -1] \cup \left[\frac{1}{2}, \alpha\right]$ with

$$\lim_{\alpha\to\beta} d^2_{\alpha|\beta}(p(x), q(x)) = \frac{\left(p(x)^{\beta} + q(x)^{\beta}\right)^{\frac{1}{\beta}-1}}{\log 2} \cdot \left[p(x)^{\beta} \log\left(\frac{2p(x)^{\beta}}{p(x)^{\beta} + q(x)^{\beta}}\right) + q(x)^{\beta} \log\left(\frac{2q(x)^{\beta}}{p(x)^{\beta} + q(x)^{\beta}}\right)\right]$$

Many common measures of distance between probability distributions belong to this family as shown in Table 1.

Table 1. Common distance measures belonging to $D^2_{\alpha|\beta}$

| | Distance Measure |
|---|---|
| $\frac{1}{2}D^2_{1\|\frac{1}{2}}$ | Squared Hellinger Distance |
| $D^2_{1\|-\infty}$ | Total Variation Distance |
| $D^2_{1\|1}$ | Jensen-Shannon Distance |
| $D^2_{1\|-1}$ | Symmetric $\chi^2$-measure |

While it is possible to find an optimal covariant metric belonging to this set as was done in Hein and Bosquest's paper, we will focus on squared Hellinger Distance for computational considerations [24].

$$d^2_{1|\frac{1}{2}}\left(p\left(x\right),q\left(x\right)\right)=p\left(x\right)+q\left(x\right)-2\sqrt{p\left(x\right)q\left(x\right)} \tag{3.6}$$

Squared Hellinger distance can be interpreted as the total difference in mass between each distribution $p$ and $q$ and their geometric mean. Squared Hellinger distance ($H^2$) has an explicit parametric form for many common distributions including Normal, Gamma, and Beta distributions [28, 29]. Making a prediction using the model will require calculating the squared Hellinger Distance from the belief state of interest to each belief state in the set of training data. For models with a finite number of outcomes, the computation time is manageable. However, a more complex outcome space would make Monte Carlo integration of the expectation too

costly to implement without a formula for the distance directly. Gaussian quadrature however still may be a viable option in the general case.

### 3.2 Kernel Regression

The term kernel regression refers to two distinctly different techniques. The first estimates $\hat{y} = E(Y|\mathcal{X} = x)$ using probability kernels. Not all probability kernels are positive definite kernels and vice-versa which is why we will focus instead on the second type. The second type of kernel regression is also called *kernel ridge* regression as well as *RKHS* regression and derived as follows. Given a symmetric positive definite kernel $K$ on set $\mathcal{X}$, there exists a unique Hilbert space of functions of $\mathcal{X}$ for which $K$ is a reproducing kernel otherwise known as a Reproducing Kernel Hilbert Space (RKHS) [30]. The details of RKHS will not be discussed in this proposal; however, they are required to use a special case of the Representer Theorem [31]. The problem

$$\min_{f \in \mathcal{H}} \sum_{i=1}^{n} \mathcal{C}(y_i, f(x_i)) + \lambda \|f\|^2 \tag{3.7}$$

where $\mathcal{C}$ is convex in $f$ admits a representation as

$$f_\lambda(\cdot) = \sum_{i=1}^{n} c_i K(x_i, \cdot). \tag{3.8}$$

In particular, when $C$ is quadratic, Equation 3.7 can be solved using a linear system. Kernel ridge regression uses $C(y_i, f(x_i)) = (y_i - f(x_i))^2$ giving Equation 3.7 the same form as classical linear Ridge regression. The kernel ridge regression estimate for $x \in \mathcal{X}$equals

$$\hat{y} = k(x)^T [G + \lambda I]^{-1} \boldsymbol{y} \tag{3.9}$$

where $G_{ij} = K(x_i, x_j)$ and $[k(x)]_i = k(x, x_i)$ [31]. The difficulty in kernel ridge regression comes in choosing optimal $\lambda$ and optimal parameters for $k(\cdot)$. We use k-fold cross-validation in Sections 4 and 5, but use maximum likelihood in Section 3.

### 3.2.1 Gaussian Process Models

The Maximum Likelihood Estimation method for choosing optimal kernel ridge regression parameters assumes an underlying Gaussian process model. While this approaches the problem from a different angle, the resulting prediction is identical aside from the method of parameter estimation. Gaussian processes have seen extensive use in many different fields such as spatial statistics, modeling computer simulations, and machine learning. Gaussian process models have been used for ADP since their introduction to the field by Deisenroth in 2008 in many applications including optimizing fishing management and haptic feedback [32, 33, 34]. Their proven history in the field and their familiarity for practitioners motivate our investigation of the model in the optimal BOSD context. However, the methods used in this paper are rooted in spatial statistics. These models are called kriging models in honor of John Krige who first applied them to the analysis of mining valuation in 1951 [22]. Since then, kriging models have seen widespread use in many different applications including valuing real estate, modeling air quality, and studying galaxies [35, 36, 37]. The

form of a Kriging model can be seen below:

$$\begin{aligned} y(\boldsymbol{s}) &= \boldsymbol{X}(\boldsymbol{s})\boldsymbol{\beta} + \boldsymbol{\varepsilon}(\boldsymbol{s}) \\ \boldsymbol{\varepsilon}(\boldsymbol{s}) &\sim N\left(0, \sigma^2 H(\boldsymbol{\phi}; \boldsymbol{s}) + \tau^2 \boldsymbol{I}\right) \\ H(\boldsymbol{\phi}; s_i, s_j)_{ij} &= \rho(\boldsymbol{\phi}; \|s_i - s_j\|) \end{aligned} \tag{3.10}$$

The universal kriging model is a modified linear model where all data is assumed to be tied to a *site* $s$. Consider modeling the vegetation index of different locations based on average annual rainfall. The sites $s$ are the locations where data is observed. The average annual rainfall is a predictor of the vegetation index, but is also itself location dependent. It along with the intercept make up $\boldsymbol{X}(\boldsymbol{s})$. Still, there are more factors involved in predicting the vegetation index than just rainfall that are also location dependent, e.g., altitude and locations of urban centers. These are wrapped up in the site-dependent error $\boldsymbol{\varepsilon}(\boldsymbol{s})$. Some amount of error is assumed to be non-site dependent, for example measurement error. This type of error is called the *nugget* effect and is denoted by $\tau^2$. Site dependent error depends has a magnitude component $\sigma^2$ and a correlation component $H(\phi; \boldsymbol{s})$. For two observations at locations $s_i$ and $s_j$, the correlation between them is assumed to depend on the distance between the two sites and parameter vector $\boldsymbol{\phi}$. For $\rho(\boldsymbol{\phi}; \cdot)$ to be a valid correlation function, we need to meet two requirements: $\rho \in [0, 1]$, and $\rho(\boldsymbol{\phi}; \cdot)$ is a positive definite function. $\rho \in [0, 1]$ is a simple condition to fulfill for any function with known extrema, and given a conditionally positive definite distance, either Equation 3.1 or Equation 3.2 can be used. For example, $\rho(\boldsymbol{\phi}; \|s_i - s_j\|) = \exp\left[-\phi \|s_i - s_j\|_2\right]$ is commonly used for spatial statistics. To apply the kriging model to BOSD, the sites used are the

belief states observed during the *simulation* step of Algorithm 2 $\left\{x_t^j\right\}_{j=1}^M$ . Any belief state dependent covariates such as the mean and log-variance of the current belief state are encoded in $\boldsymbol{X}(\boldsymbol{s})$. Finally, $\|s_i - s_j\|$ refers to some statistical distance described in Section 3.1. For some new site $s$,

$$\hat{y}(s) = x^T(s)\,\hat{\beta} + \hat{\boldsymbol{\gamma}}(s)^T\,\hat{\Sigma}^{-1}\left(y - X\hat{\beta}\right) \tag{3.11}$$

where $\hat{\boldsymbol{\gamma}}(s)_i = \hat{\sigma}^2\rho\left(\hat{\boldsymbol{\phi}}; \|s_i - s\|\right)$, $\hat{\beta} = \left[X^T\hat{\Sigma}^{-1}X\right]^{-1}X^T\hat{\Sigma}^{-1}y$, and $\hat{\Sigma} = \hat{\sigma}^2 H\left(\hat{\boldsymbol{\phi}}; \boldsymbol{s}\right) + \hat{\tau}^2\boldsymbol{I}$[27]. Since the model assumes Gaussian likelihood, parameters $\hat{\boldsymbol{\phi}}$, $\hat{\sigma}^2$, and $\hat{\tau}^2$ are calculated using Maximum Likelihood or Restricted Maximum Likelihood. This can be optimized directly, via approximate methods such as Newton-Raphson or Scoring, or some combination of multiple methods [38]. This is notably faster than the grid search used for GLS; however, the assumption of Gaussian error is loose unless $\tilde{U}$ is being approximated using Monte Carlo.

### 3.3 Potential Issues

Kernel regression is fundamentally a spline which has its pros and cons when used in ADP. On one hand, using a spline makes few assumptions ensuring accurate interpolation for values inside the range of the data. On the other hand, splines will not make assumptions outside the range of the data. This places high importance on $\pi^{\text{explore}}$ to make sure the necessary regions of the design space are covered. Some common exploration policies include *epsilon-greedy* where a combination of optimal and random choices are used and the addition of a noise component to optimal design points. Kernel regression also requires inversion of an $M$ by $M$ matrix to fit

a model which is order $O(M^3)$ in time and $O(M^2)$ in memory [39]. Depending on the number of trajectories simulated and the number of trials, kernel regression may not be computationally feasible. Some solutions include using a divide and conquer approach and randomly selecting a subset of trajectories to use as data instead of the whole sample though these approaches may impact the convergence of the ADP algorithm [39, 40]. When using a Gaussian process model, the correlation function can also be extended to include a temporal component based on experiment trial requiring only a single model to be fit, but the larger model increases prediction time [41].

### 3.4 Applications using Conjugate Models

We considered two examples for conjugate models: a Linear-Gaussian used as a benchmark by Huan and Marzouk in their paper as well as the simple Beta-Bernoulli model used in Section 1.

### 3.5 Linear-Gaussian Problem

The Linear-Gaussian model has a single parameter of interest $\theta$ with prior $N(\mu_0, \sigma_0^2)$ and has error $\epsilon \sim N(0, \sigma_\epsilon^2)$.

$$y_k = \theta d_k + \epsilon$$

Given observation $d_k$, the posterior belief state evolves to

$$\left(\mu_{k+1}, \sigma_{k+1}^2\right) \sim N\left(\frac{\frac{y_k/d_k}{\sigma_\epsilon^2/d_k^2} + \frac{\mu_k}{\sigma_k^2}}{\frac{1}{\sigma_\epsilon^2/d_k^2} + \frac{1}{\sigma_k^2}}, \frac{1}{\frac{1}{\sigma_\epsilon^2/d_k^2} + \frac{1}{\sigma_k^2}}\right). \tag{3.12}$$

We consider the benchmark design from Huan and Marzouk of $N = 2$ experiments with prior parameters $\mu_0 = 0$ and $\sigma_0^2 = 9$, error variance $\sigma_\epsilon^2 = 1$, and with design points $d \in [.1, 3]$. The proposed utility function is has no intermediate rewards with the terminal reward being the KL-divergence from posterior to prior penalized by the log variance:

$$\begin{aligned} u_2(x_2) &= D_{KL}(x_2 \parallel x_0) - 2\left(\log \sigma_N^2 - \log 2\right)^2 \\ &= \log \frac{\sigma_N^2}{\sigma_0^2} + \frac{\sigma_0^2 + (\mu_0 - \mu_N)^2}{2\sigma_N^2} - \frac{1}{2} - 2\left(\log \sigma_N^2 - \log 2\right)^2 \end{aligned}$$

Huan showed the expected utility for this experiment attains a maximum at any pair of designs $(d_0^*, d_1^*)$ such that

$$\begin{aligned} d_0^{*2} + d_1^{*2} &= \frac{1}{9}\left[\exp\left(\frac{18014398509481984\log 3 - 5117414861322735}{9007199254740992}\right) - 1\right] \\ &\approx 0.45546311542230206 \end{aligned}$$

with $U(d_0^*, d_1^*) \approx 0.783289$ [7]. Figure 2 shows the exact utility of each pair of design points in the region of interest.

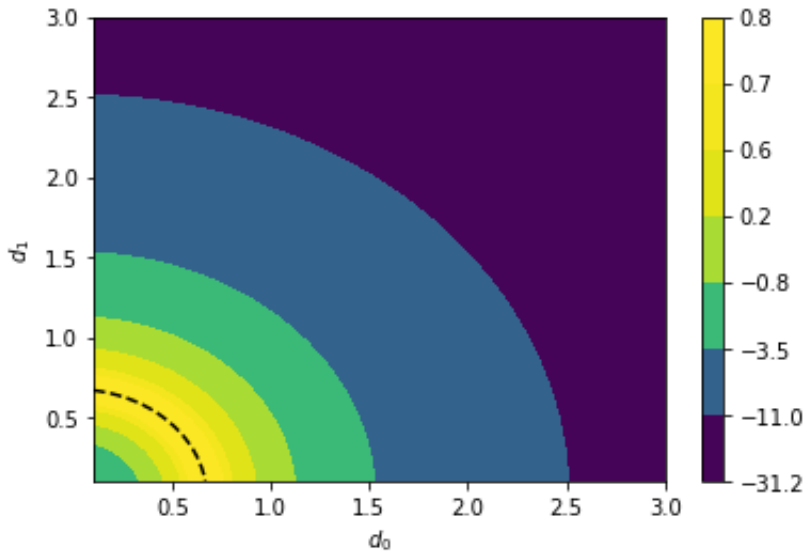


Fig. 2. Expected utility for all possible designs. The dashed line denotes optimal designs.

The surrogate model used in their paper is the linear model shown in Equation 3.13 with $\mu_i$ and $\sigma_i^2$ being the mean and variance $\theta$ at belief state $s_i$

$$\tilde{U}(s_i) = \beta_0 + \beta_1\mu_i + \beta_2 \log\sigma_i^2 + \beta_3\mu_i^2 + \beta_4\mu_i \log\sigma_i^2 + \beta_5 \left(\log\sigma_i^2\right)^2 + \varepsilon_i \qquad (3.13)$$

$$\varepsilon(s_i) \sim N\left(0, \tau^2\right)$$

The exploration policy used by Huan and Marzouk randomly selects $d \sim N\left(1.25, 0.5^2\right)$ for each design point, projecting any values outside $[0.1, 3]$ back inside. $L = 3$ iterations of the ADP algorithm were used. The resulting contour plots are shown replicated in Figure 3.

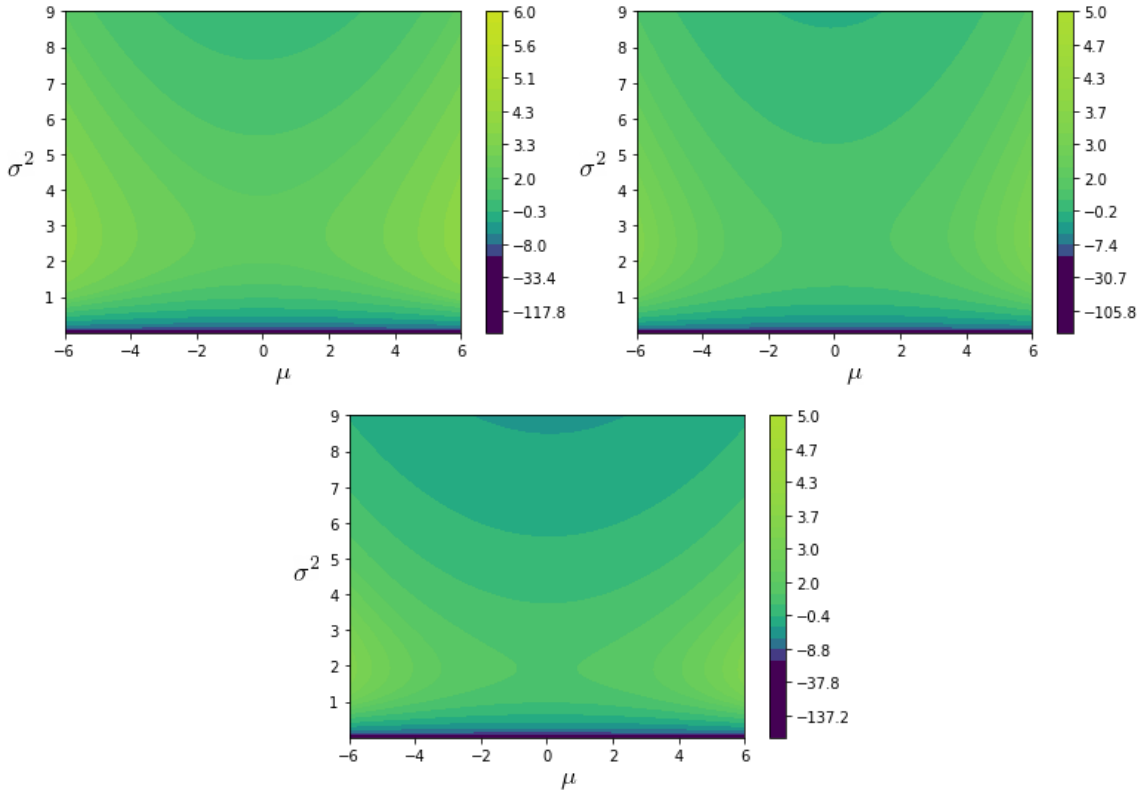


Fig. 3. Contour plots of $\tilde{U}_1$ for $L = 1, 2, 3$ from left to right fit using the method in Huan and Marzouk's original paper.

We examined four surrogate models in addition to replicating the results of the original paper: an intercept only kriging model and a kriging model with features derived from Huan and Marzouk's original paper; each with two different exploration policies. Both models have the following covariance structure:

$$\boldsymbol{\varepsilon}(\boldsymbol{s}) \sim N\left(0, \sigma_1^2 H^1 + \sigma_2^2 H^2 + \sigma_3^2 H^3 + \tau^2 \boldsymbol{I}\right) \tag{3.14}$$

$$H_{ij}^k = \left[1 - H^2(s_i, s_j)\right]^k$$

$$H^2(s_i, s_j) = 1 - \sqrt{\frac{2\sigma_i \sigma_j}{\sigma_i^2 + \sigma_j^2}} \exp\left[-\frac{1}{4}\frac{(\mu_i - \mu_j)^2}{\sigma_i^2 + \sigma_j^2}\right] \tag{3.15}$$

$s_i$ and $s_j$ denote the belief states at sites $i$ and $j$. These belief states are normal random variables with means $\mu_i$ and $\mu_j$ and variances $\sigma_i^2$ and $\sigma_j^2$ respectively. $L = 3$ iterations of the ADP algorithm were used for each design. All estimates were calculated using Restricted Maximum Likelihood (REML). For each surrogate model, two exploration methods were used. The first method follows Huan and Marzouk's original paper, discarding previous simulated trajectories each iteration. The second method kept all previous simulated trajectories each iteration. Both methods simulated 100 exploration trajectories the first iteration, then 30 the following two iterations. We used SciPy's implementation of the L-BFGS-B algorithm for the *exploitation* step and the Nelder-Mead simplex method to find the REML estimate of the Gaussian process model [42]. More information on both algorithms can be found in Kelley's texbook *Iterative Methods for Optimization.*

### 3.5.1 Ordinary Kriging Model

An intercept only kriging model is also called an *ordinary* kriging model. Contour plots of $\tilde{U}_1$ for belief states reached after the first trial are shown in Figure 4. We can see the kriging model is accurate in areas sufficiently explored by the ADP algorithm. The optimal sequential design found is $d_0 = 0.1238$, with $d_1 \approx 0.6634 \pm 4e-5$

depending on the outcome observed for $d_0$. Since $d_0^2+d_1^2 = 0.1238^2+0.6634^2 \approx 0.4554$, we know this design combination lies of the circle of optimal designs.

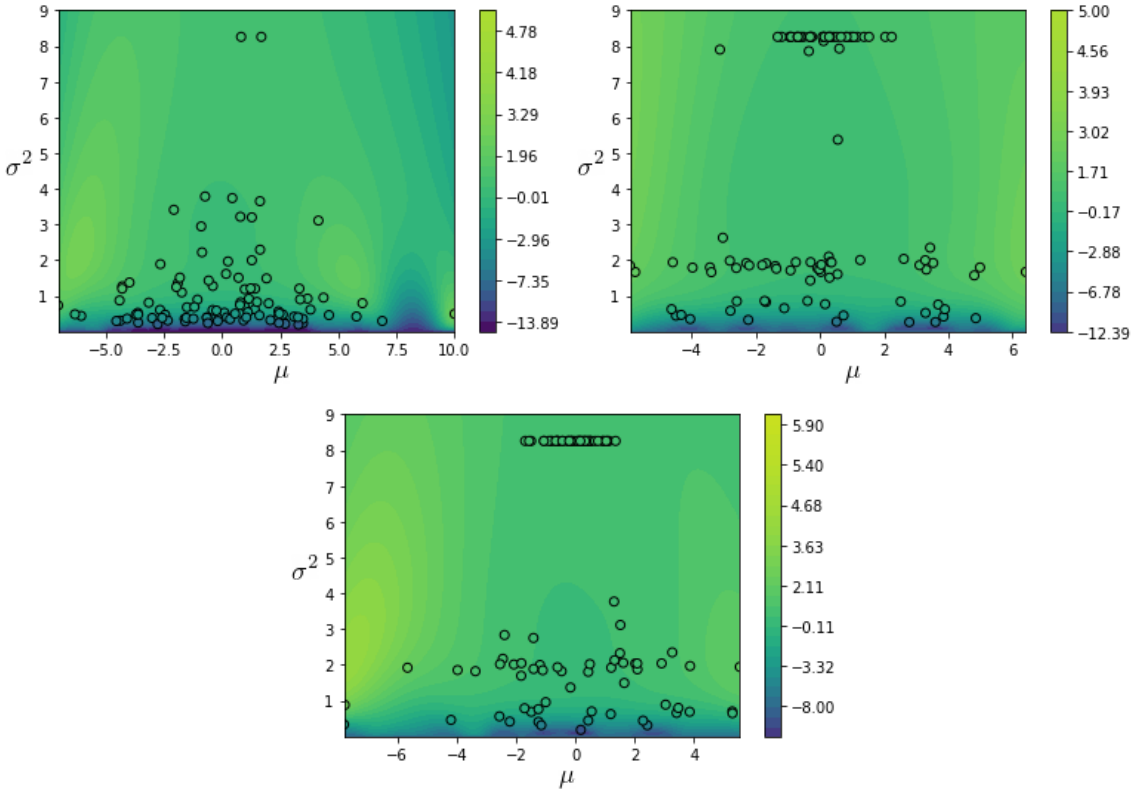


Fig. 4. Contour plots of $\tilde{U}_1$ for $L = 1, 2, 3$ from left to right fit using an Ordinary Kriging model using the original exploration policy.

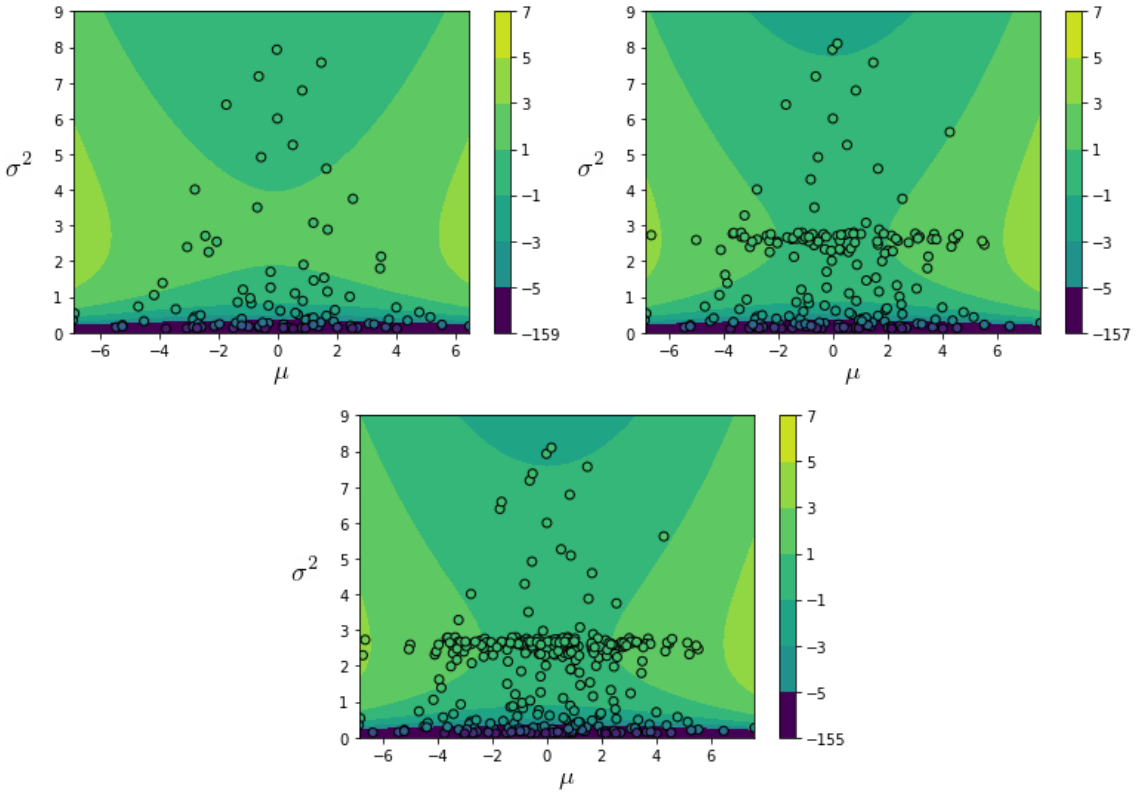


Fig. 5. Contour plots of $\tilde{U}_1$ for $L = 1, 2, 3$ from left to right fit using an Ordinary Kriging model using the retaining exploration policy.

### 3.5.2 Universal Kriging

The universal kriging model includes the following linear predictors in addition to the intercept: $\mu$, $\log \sigma^2$, $\mu^2$, $\log^2 \sigma^2$, and $\mu \log \sigma^2$. Contour plots of $\tilde{U}_1$ using the universal kriging model are shown in Figure 6. Unlike the ordinary kriging model, the quadratic terms in the universal kriging model predicts high utility for $\sigma^2 \approx 2.5$ and $|\mu| > 4$. This quality is present in Huan and Marzouk's original model as seen in Figure 3 and is mainly due to the $\mu^2$ effect included in the model. The

flat central region present in Figure 4 is also present showing the impact of using the Kriging method. The optimal sequential design found is $d_0 = 0.1351$, then $d_1 \approx 0.6612 \pm 4e-4$. Since $d_0^2 + d_1^2 = 0.1351^2 + 0.6611^2 \approx 0.4554$, we know this design combination lies of the circle of optimal designs.

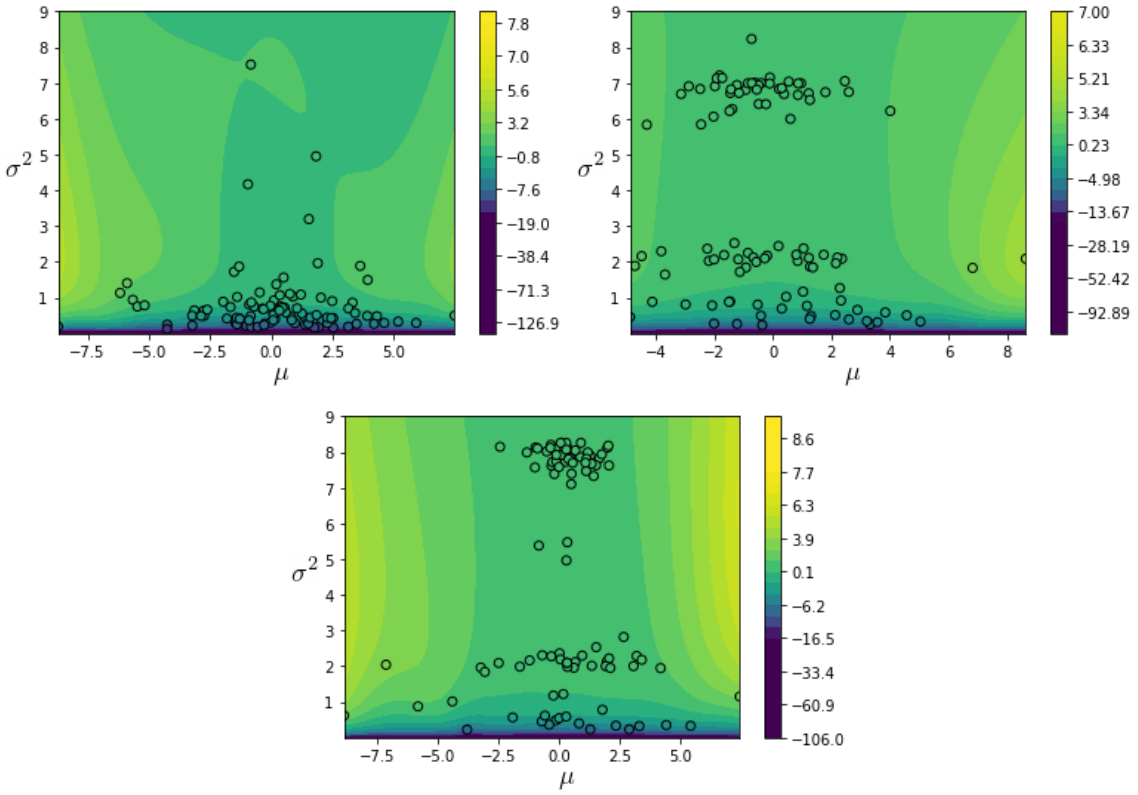


Fig. 6. Contour plots of $\tilde{U}_1$ for $L = 1, 2, 3$ from left to right fit using an Universal Kriging model using the original exploration policy.

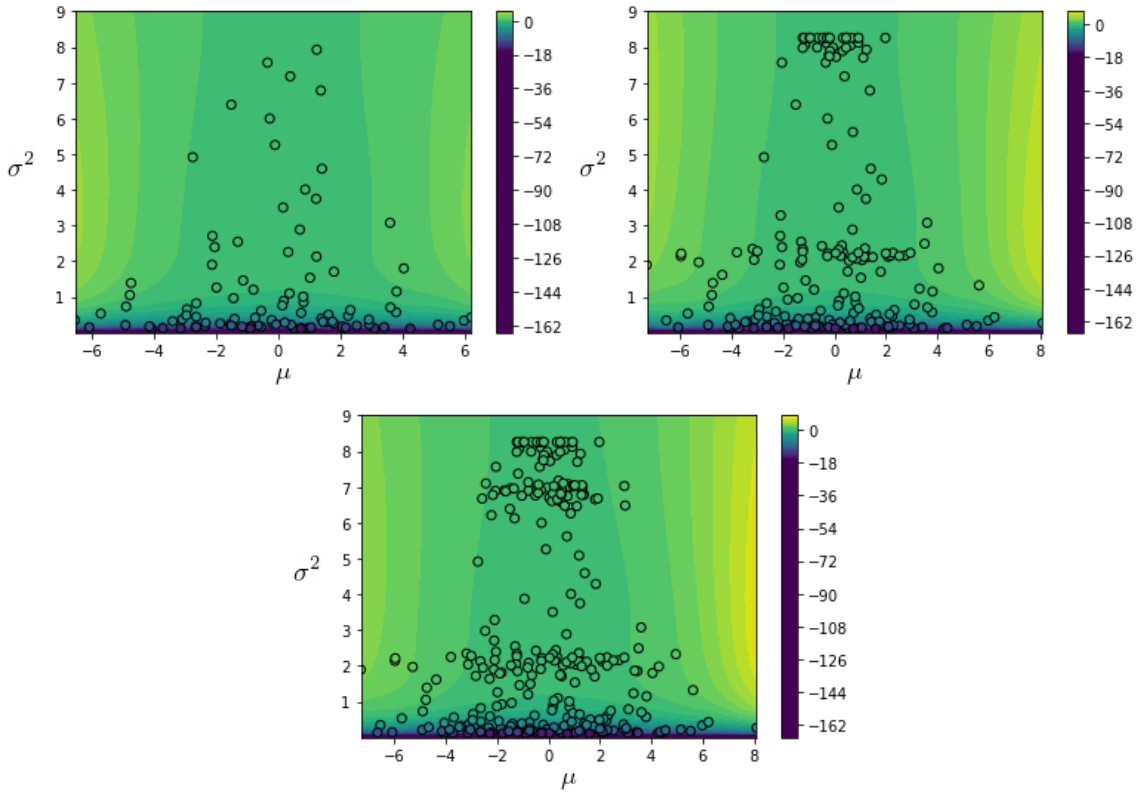


Fig. 7. Contour plots of $\tilde{U}_1$ for $L = 1, 2, 3$ from left to right fit using an Universal Kriging model using the retaining exploration policy.

### 3.6 Beta-Binomial Model

We once again consider the Beta-Binomial model from Section 1. Recall that since we have a finite number of outcomes and possible design points, we can use backward induction to solve the problem exactly. Figure 1 displays the exact BOSD. We can decompose the squared Hellinger distance as shown in Equation 3.16.

$$
\begin{aligned}
1 - H^2\left(x_i, x_j\right) &= \int\int\int \sqrt{p_i\left(\theta_1, \theta_2\theta_3\right) p_j\left(\theta_1, \theta_2\theta_3\right)} d\theta_1 d\theta_2 d\theta_3 \\
&= \int\int\int \sqrt{p_{1i}\left(\theta_1\right) p_{2i}\left(\theta_2\right) p_{3i}\left(\theta_3\right) p_{1j}\left(\theta_1\right) p_{2j}\left(\theta_2\right) p_{3j}\left(\theta_3\right)} d\theta_1 d\theta_2 d\theta_3 \\
&= \int \sqrt{p_{1i}\left(\theta_1\right) p_{1j}\left(\theta_1\right)} d\theta_1 \cdot \int \sqrt{p_{2i}\left(\theta_2\right) p_{2j}\left(\theta_2\right)} d\theta_2 \cdot \int \sqrt{p_{3i}\left(\theta_3\right) p_{3j}\left(\theta_3\right)} d\theta_3 \\
&= \left(1 - H^2\left(\theta_{1i}, \theta_{1j}\right)\right)\left(1 - H^2\left(\theta_{2i}, \theta_{2j}\right)\right)\left(1 - H^2\left(\theta_{3i}, \theta_{3j}\right)\right)
\end{aligned}
\tag{3.16}
$$

The expression for the squared Hellinger distance between two Beta random variables $x_i$, $x_j$ is known to be Equation 3.17.

$$
1 - H^2\left(x_i, x_j\right) = \frac{B\left(\frac{\alpha_i + \alpha_j}{2}, \frac{\beta_i + \beta_j}{2}\right)}{\sqrt{B\left(\alpha_i, \beta_i\right) B\left(\alpha_j, \beta_j\right)}} \tag{3.17}
$$

Combining Equation 3.16 and Equation 3.17 then yields the expression for squared Hellinger distance used in the surrogate models. The surrogate model used can be seen in Equation 3.18.

$$
\begin{aligned}
\boldsymbol{y}\left(\boldsymbol{s}\right) &= \boldsymbol{\mu} + \boldsymbol{\varepsilon}\left(\boldsymbol{s}\right) \\
\boldsymbol{\varepsilon}\left(\boldsymbol{s}\right) &\sim N\left(0, \sigma_1^2 K^1 + \sigma_2^2 K^2 + \sigma_4^2 K^4 + \tau^2 \boldsymbol{I}\right) \\
K_{ij}^k &= \left[1 - H^2\left(s_i, s_j\right)\right]^k
\end{aligned}
\tag{3.18}
$$

The exploration policy used randomly selects from the three treatments with equal probability. Exploitation simply calculates the estimated utility at for each

possible outcome and selects the best performer. The first iteration used 200 exploration trajectories while the subsequent 10 iterations used 100 exploration and 100 exploitation. All trajectories were carried over from previous iterations to fit the surrogate models. Again, REML was carried out using the Nelder-Mead simplex method.

### 3.7 Results

In the $L = 11$ iterations run, the algorithm did not converge to the exact optimal solution. Iteration 10 performed the best of the 11 iterations with an expected utility of 28.147 and can be seen in Figure 8. The approximate BOSD is 99% efficient compared to the myopic design's 95% efficiency which is a notable increase and extremely close to optimal.

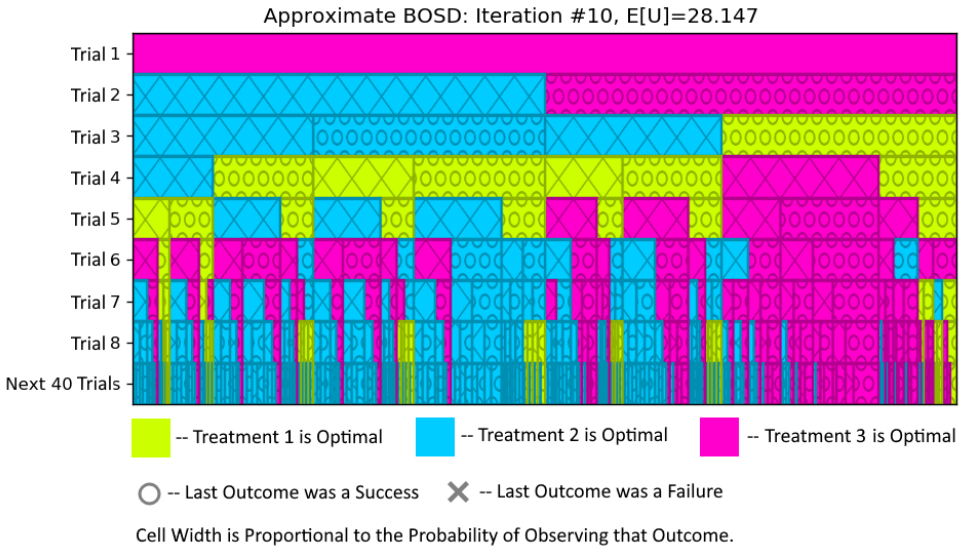


Fig. 8. Approximate BOSD for Example 2. Iteration 10 of the ADP algorithm.

## 3.8 Conclusions

Gaussian process surrogate models indexed on belief states found optimal solutions for the Linear-Gaussian model and a near optimal solution for the Beta-Bernoulli model. Applications where the conjugate distribution has an known expression of statistical distance are limited however. The Beta-Bernoulli model is the most obviously useful in a practical application and is well documented for the infinite horizon case [5]. In other cases such, linear regression with unknown variance, a Monte Carlo integration in combination with a known formula may perform well. For a simple hierarchical model, we have

$$\begin{aligned}
H^2\left(p_{xy}, q_{xy}\right) &= \int_Y \int_X \sqrt{p_{xy}\left(x,y\right) q_{xy}\left(x,y\right)} dx dy \\
&= \int_Y \sqrt{p_y\left(y\right) q_y\left(y\right)} \int_X \sqrt{p_{x|y}\left(x,y\right) q_{x|y}\left(x,y\right)} dx dy \\
&= \int_Y \sqrt{p\left(y\right) q\left(y\right)} \left(1 - H^2\left(p_{x|y}, q_{x|y}\right)\left(y\right)\right) dy \\
&= \left(1 - H^2\left(p_y, q_y\right)\right) \int_Y \sqrt{p_y\left(y\right) q_y\left(y\right)} H^2\left(p_{x|y}, q_{x|y}\right)\left(y\right) dy.
\end{aligned}$$

Depending on the distribution of $y$, $\int_Y \sqrt{p_y\left(y\right) q_y\left(y\right)} H^2\left(p_{x|y}, q_{x|y}\right)\left(y\right) dy$ may be reasonable to estimate via numerical methods. The proposed methodology could be adapted to a much broader class of models where this is the case.

# CHAPTER 4

# BAYESIAN OPTIMAL SEQUENTIAL DESIGNS VIA APPROXIMATE DYNAMIC PROGRAMMING USING SEQUENTIAL IMPORTANCE SAMPLING

Application of the ADP algorithm using distance-based modeling requires a representation of the belief state at each experimental outcome and the ability to quickly compute some statistical difference between these representations. As shown in 3, these criteria are can be easily satisfied for cases with conjugate priors. In cases without conjugate priors, approximations to the posterior must be used. Unfortunately, many approximations to the posterior lack either a quick method to compute statistical differences between them or do not perform well in sequential scenarios. For instance, the Normal asymptotic and variational approximations to the posterior have closed forms for both Hellinger distance and Jensen-Shannon divergence, but may not always be appropriate [43]. On the other hand, variational approximations using Normal mixture distributions would be much more accurate; however, normal mixture distributions do not have a statistical distance between them which is quick to compute [44]. Sampling via measure transport would be accurate, but transfer maps lack an informative distance measure [45, 46]. Sequential Monte Carlo (SMC) and similar methods such as grid approximations are the only methods we found fitting both criteria.

## 4.1 Sequential Monte Carlo

Sequential Monte Carlo (SIS), known as Particle Filtering for Markov models, is an extension of importance sampling where distributions are approximated using weighted samples. To perform SIS, draw a sample from the prior $\{\theta_k\}_{k=1}^{K} \sim p(\theta)$. We can approximate $\{\tilde{w}_k^t\}_{k=1,t=1}^{K,T}$

$$p(\theta) \overset{\text{appx}}{\sim} \sum_{k=1}^{K} \frac{1}{K} \delta_{\theta_k}(\theta). \tag{4.1}$$

$\delta_{\theta_k}(\theta)$ denotes a Dirac delta measure which has probability 1 when $\theta = \theta_k$ and 0 otherwise. Then following Bayes Theorem,

$$p(\theta|y,\xi) \overset{\text{appx}}{\sim} \frac{\sum_{k=1}^{K} p(y|\theta,\xi)\, \delta_{\theta_k}(\theta)}{\sum_{k=1}^{K} p(y|\theta_k,\xi)\, \delta_{\phi_k}(\theta_k)}. \tag{4.2}$$

Consider the initial sample to be a weighted sample with weights $w_k = K^{-1} \forall k = 1, 2, \ldots, K$. The posterior is then another weighted sample with

$$w_k = \frac{p(y|\theta,\xi)}{\sum_{k=1}^{K} p(y|\theta_k,\xi)\, \delta_{\theta_k}(\theta_k)}. \tag{4.3}$$

Importance sampling effectively creates a conjugate prior. By continuing to re-weight the initial samples, we can compute approximations to belief states deeper into the experiment. However, the accuracy of the approximation decreases as the distance between the initial sampling distribution and the approximated belief state increases. Typically, the effective sample size ($ESS$) of the current weight vector is used as a benchmark to determine the accuracy of the current approximation.

$$ESS = \left[\sum_{k=1}^{K} w_k^2\right]^{-1} \tag{4.4}$$

Once $ESS$ falls below some predetermined threshold, the current sample must be replaced. Typically, this is done using the *Resample-Move* algorithm by *filtering* out samples with low probability via bootstrap sampling, then perturbing the resampled particles to add diversity via Markov Chain Monte Carlo [47].

## 4.2 Application to BOSD

In order to apply methods from Section 3.1, we need a measure $\mu$ dominating all Dirac delta mixtures we need to compare. Since the sample is fixed at this point, we dominate the measure of the mixture weights by measuring 1 at each mixed Dirac delta. This leads to simply replacing any integration with a summation over the sampled values instead. Since Particle Filtering approximates the current belief state with a mixture of Dirac delta functions, the only probability mass the approximation has is at the particle locations. Consider three PF approximations with probability mass following the table below:

Table 2. Example Particle Filter Approximations

| $x$ | 0 | 1 | 1.01 | 2 |
|---|---|---|---|---|
| $\hat{p}_1(x)$ | 0.5 | 0.5 | 0 | 0 |
| $\hat{p}_2(x)$ | 0.5 | 0 | 0.5 | 0 |
| $\hat{p}_3(x)$ | 0.5 | 0 | 0 | 0.5 |

Intuition and non-covariant metrics would suggest $\hat{p}_1$ and $\hat{p}_2$ should be closer together than $\hat{p}_1$ and $\hat{p}_3$. However, every statistical distance belonging to the family given by Equation 3.5 would determine the two distances are the same. While technically correct, these distances are uninformative, especially since these distributions are approximations of other smooth distributions. For the surrogate model to perform properly, the distance between two approximations must be reflective of the distance between the distributions being approximated. The simplest solution is to ensure the particles used for each approximation are the same before computing distances. While this does rule out the use of MCMC methods, importance sampling can work perfectly fine. Using importance sampling, generate a sample from the prior based on some proposal distribution. Then, re-weight the new sample using the recorded design points and observations from previous trials.

How to choose such a proposal distribution may differ problem to problem or may differ even trial to trial. For example, consider a scenario where simulated belief states concentrates at several different modes. Ensuring adequate representation among all simulated belief states may be difficult if not impossible if the spread among belief states is too large. Importance sampling using a mixture distribution comprised of approximations to the current stage's belief states as the proposal is a natural fit, but calculating the mixture weights ultimately proved too difficult. Instead, we simply use the prior as the proposal distribution. The pdf of belief state $x_N$ for an sequential design has

$$p\left(\theta \mid x_T\right) \propto p\left(\theta \mid x_0\right) \prod_{t=1}^{T} p\left(y_t \mid \theta, \xi_t\right). \tag{4.5}$$

Since we need to compare distances for different combinations of $\{y_t\}_{t=1}^{T}$, the prior is the only common element. Ultimately, it would be ideal for particles to be concentrated in areas of interest however we had little luck finding optimal weight configurations despite trying heuristics, convex optimization, and subgradient methods [48, 49]. If necessary, the total number of particles can be increased when changing the base particle set to accommodate unexpected spread among belief states though it will increase computation time. Under a given particle approximation $\Theta = \{\theta_k\}_{k=1}^{K}$, the squared Hellinger Distance between two distributions is

$$\begin{aligned} H^2\left(p\left(x\right), q\left(x\right)\right) &= \frac{1}{2}\left[\sum_{\theta_k \in \Theta} p\left(\theta_k\right) + q\left(\theta_k\right) - 2\sqrt{p\left(\theta_k\right) q\left(\theta_k\right)}\right] \\ &= 1 - \sum_{\theta_k \in \Theta} \sqrt{p_k q_k} = 1 - \sum_{k=1}^{K} \exp\left[\frac{1}{2}\left(\log p_k + \log q_k\right)\right]. \end{aligned}$$

Where $p_k$ and $q_k$ are the importance weights for distributions $p$ and $q$ under sample $\Theta$. Note that if stored in log-scale, calculation of the squared Hellinger distance requires one exponentiation as the only slow computation. Approximate Value Iteration using a distance-based model requires calculating the distance from each belief state in $\mathcal{X}_t^i$ to each belief state from the trial after it $\mathcal{X}_{t+1}^i$. These calculations are for the most part simple arithmetic between large matrices of log-scale weights which allows for substantial improvements in runtime by using GPU acceleration via CUDA. Compared to computing on a CPU where most tasks are performed in singular threads, GPUs excel at performing simple operations in extreme parallel. Nvidia's CUDA is a framework allows for GPUs to be used in non-graphical ap-

plications. GPU acceleration via CUDA is commonly used in machine learning via platforms such as TensorFlow, PyTorch, and CuPy. We used CuPy and NumPy to implement our algorithm due to their low barrier of entry [50, 51].

In addition to changes required due to shifting particle approximations, Algorithm 2 is adjusted by no longer simulating outcomes when expanding the set of visited belief states. In practice, we found simulating outcomes could not identify rare events and selecting outcomes independently of the marginal probability produced better results. Additionally, we follow the suggestion of Deisenroth and have an larger initial exploration sample to drive exploitation [34].

**Algorithm 3** Solving sOED using ADP via Kernel Regression and SMC.

**for** $i = 1, \ldots, N$ **do**

**Set parameters:** Select number of experiments $T$, exploration policy $\pi^{\text{explore}}$, number of policy updates $N$, number of importance samples $K$, number of exploration trajectories $M_i^{\text{explore}}$ for $i = 1, \ldots, N$, and number of exploitation trajectories $M_i^{\text{exploit}}$ for $i = 1, \ldots, N$. Denote the total number of experimental trajectories to be added for $i > 0$ $M_i := M_i^{\text{explore}} + M_i^{\text{exploit}}$.

Set SMC approximations $\psi_0(\cdot), \ldots, \psi_{T+1}(\cdot)$. Sample $K$ samples from each.

**for** $i = 1, \ldots, N$ **do**

**State Set Augmentation:** Sample $M_i^{\text{explore}}$ exploration trajectories: Sample $\xi_t$ and $y_t$ from exploration policy $\pi^{\text{explore}}$. Transition $x_{t+1} \leftarrow \mathcal{F}_t(x_t, y_t, \xi_t)$. Repeat for $t = 0, \ldots, T$

Store all posterior belief states visited in $\mathcal{X}_{t,\text{explore}} = \left\{x_t^j\right\}_{j=1}^{M_i^{\text{explore}}}$ for $t = 1, \ldots, T+1$

*Exploitation*— If $i > 1$, simulate $M_i^{\text{exploit}}$ exploitation trajectories: Compute particle approximations $\left\{\psi_t\left(x_t^j\right)\right\}_{j=1}^{M_i^{\text{exploit}}}$ and $\left\{\psi_{t+1}\left(x_t^j\right)\right\}_{j=1}^{M_i^{\text{exploit}}}$, then calculate

$$\xi_t \leftarrow \arg\max_{\xi_t^* \in \chi_t} \int_y p(y_t|\psi_t(x_t), \xi) \left[u_t(\psi_t(x_t), y_t, \xi_t^*) + \tilde{U}_{t+1}^{i-1}(\mathcal{F}_t(\psi_t(x_t), y_t, \xi_t^*))\right] dy_t$$

, and sample $y_t$ from $\pi^{\text{explore}}$. Transition $x_{t+1} \leftarrow \mathcal{F}_t(x_t, y_t, \xi_t)$. Repeat for $t = 0, \ldots, T$

Store all posterior belief states visited in $\mathcal{X}_{t,\text{exploit}}^i = \left\{x_t^j\right\}_{j=1}^{M_i^{\text{exploit}}}$ for $t = 1, \ldots, T+1$

**end for**

**Approximate value iteration:** Construct functions $\tilde{U}_t^i$ via backward induction using all belief states $\mathcal{X}_{t+1}^i := \cup_{l=1}^i \left\{ \mathcal{X}_{t+1,\text{explore}}^l \cup \mathcal{X}_{t+1,\text{exploit}}^l \right\}$, $t = 1, \ldots, T-1$ using the method below

Calculate $\hat{U}_{T+1}^j \leftarrow E\left[x_{T+1}^j\right]$ for $j = 1, \ldots, M$ by applying an MCMC kernel to each $\psi_{T+1}^j\left(x_{T+1}^j\right)$, then store them.

If necessary, apply a variance stabilizing transform $T_{T+1}$ to $\hat{U}_{T+1}^i\left(x_{T+1}^j\right)$

Then construct $\tilde{U}_{T+1}^i$ via kernel regression using data set coupled with inverse transform $T_{T+1}^{-1}$ if applicable $\left\{\psi_{T+1}\left(x_t^j\right), \hat{U}_{T+1}^i\left(x_t^j\right)\right\}_{j=1}^{\sum_{l=1}^i M_l}$ and a kernel as described in Section 3.1.

**for** $t = T, \ldots, 1$ **do**

Compute $\left\{\psi_t\left(x_t^j\right)\right\}_{j=1}^{M_i^{\text{exploit}}}$ and $\left\{\psi_{t+1}\left(x_{t+1}^j\right)\right\}_{j=1}^{M_i^{\text{exploit}}}$

**for** $j = 1, \ldots, \sum_{l=1}^i M_l$ **do**

Compute

$$\hat{U}_t^i\left(x_t^j\right) = \max_{\xi_t^* \in \chi_t} \int_y p\left(y_t | \psi_t\left(x_t\right), \xi_t^*\right) \left[u_t\left(\psi_t\left(x_t^j\right), y_t, \xi_t\right) + \tilde{U}_{t+1}^i\left(\mathcal{F}_t\left(\psi_{t+1}\left(x_t^j\right), y_t, \xi_t^{'}\right)\right)\right] dy_t$$

If necessary, apply a variance stabilizing transform $T_t$ to $\hat{U}_t^i\left(x_t^j\right)$

Then construct $\tilde{U}_t^i$ via kernel regression using data set coupled with inverse transform $T_t^{-1}$ if applicable $\left\{\psi_t\left(x_t^j\right), \hat{U}_t^i\left(x_t^j\right)\right\}_{j=1}^{\sum_{l=1}^i M_l}$ and a kernel as described in Section 3.1.

**end for**

**end for**

**end for**

**Return final policy parameterization:** $\tilde{U}_t^N$, $t = 1, \ldots, T$

### 4.3 Application

Drovandi et al. performed simulation experiments testing myopic Bayesian optimal designs to identify various target stimulus probabilities [52]. Three different likelihoods were tested using both binary and count data. These sequential experi-

ments were replicated 500 times each with up to 100 trials per experiment to examine how close the posterior median came to the target parameters generating the data. The power-logistic model used is defined in Equation 4.6.

$$\log\left(\frac{p}{1-p}\right) = \theta_0 + \theta_1 \frac{\xi - 1}{\lambda} \tag{4.6}$$

Drovandi et al. examined three cases: $\lambda = 1$, $\lambda \to 0$, and $0 < \lambda \leq 1$ with three different utility functions. We implemented the model for the $\lambda = 1$ case for $N = 5, \ldots, 10$ trials and all utility functions, then compared the results to the myopic methods used in their paper. The utility functions used are defined as the KL Divergence, Bayesian D-Optimal, and Precision for a target dose response stimulus. The KL Divergence utility used in this paper is

$$u_T(x_T) = D_{KL}(x_T \parallel x_0) = \int_\Theta p(\theta \mid x_T) \log \frac{p(\theta \mid x_0)}{p(\theta \mid x_T)} d\theta. \tag{4.7}$$

Drovandi et al. actually use $D_{KL}(x_t \parallel x_{t-1})$, but this utility does not translate well to the fully sequential case. Instead, we used the classic KL-Divergence utility shown in Equation 4.7. The Bayesian D-Optimality utility is

$$u_T(x_T) = \left[\det\left(\mathrm{Var}\left[\theta \mid x_T\right]\right)\right]^{-1}. \tag{4.8}$$

Note for a normal posterior this is equivalent to the KL-Divergence utility. The last utility used by Drovandi et al. is the target precision utility:

$$u_T(x_T) = \left[\mathrm{Var}\left[D^* | x_T\right]\right]^{-1} \tag{4.9}$$

where $D^*$ is the stimulus which produces a probability equal to $p^*$ of a positive response. For $\lambda = 1$,

$$D^* = \frac{\text{logit}\,(p^*) - \theta_0}{\theta_1} + 1. \tag{4.10}$$

In practice we found the variance of the Monte Carlo estimate of the target precision utility caused difficulty in modeling it. We implemented a more robust measure of spread for the target precision based on the interquartile range.

$$u_T\,(x_T) = [\text{IQR}\,[D^*|x_T]]^{-1} \tag{4.11}$$

Like Drovandi et al., we consider the case of $p^* = .2$. Drovandi et al. used a prior distribution of $x_0 \sim N(0, 100)$. Their paper considers myopic experiments out to 100 iterations resulting in a final belief state proportional to $N(0, 10)$. We examine a prior of $N\,(0, 50)$ as a middle ground.

### 4.4 Particle Approximations and Distance

A natural question with Monte Carlo methods is how many samples are necessary. For two belief states $x_i$and $x_j$ with design point-outcome pairs $\{(\xi_t^i, y_t^i)\}_{t=1}^T$ and $\{(\xi_t^i, y_t^i)\}_{t=1}^T$, their Hellinger distance is equal to

$$H^2\,(x_i, x_j; \Theta) = 1 - p\,(\theta \mid x_0) \sum_{\theta_k \in \Theta} \sqrt{\prod_{t=1}^{T} p\,(y_t \mid \theta_k, \xi_t) \prod_{t=1}^{T} p\,(y_t \mid \theta_k, \xi_t)}.$$

From as modeling perspective, the worst-case scenario is determining two belief states are similar when they are actually different since this can cause the model

to make inaccurate predictions. This scenario requires poor representation of both belief states and having excess samples in their intersection. Determining two belief states dissimilar when they actually are similar reduces the predictive power of the model, but does not cause inaccurate predictions. This is caused by poor representation in the area of the intersection of the two distributions and more representation elsewhere. Generally, both scenarios tend to happen when the variances of the two belief states are much smaller than the proposal distribution which is more likely to happen the more trials that have passed. This can also occur if the likelihood can cause the modes of the belief states to wander far away from the proposal. For this application, the likelihood of the logistic regression model essentially bisects the plane along a line determined by choice of $\xi$ as shown in Figure 10. This makes it both difficult for the mode to move too far from the proposal distribution and difficult to construct belief states with variance low enough that they cannot be represented well enough with a few thousand samples.

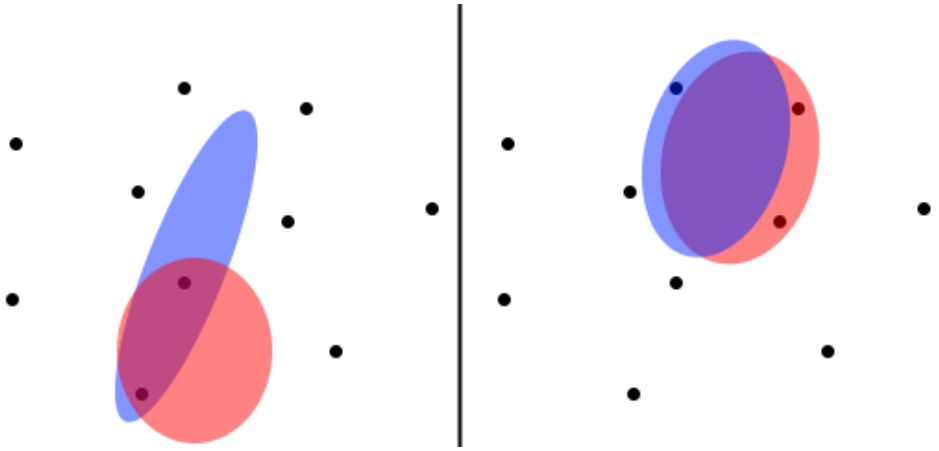

Fig. 9. Example Distance Errors with Particle Approximations

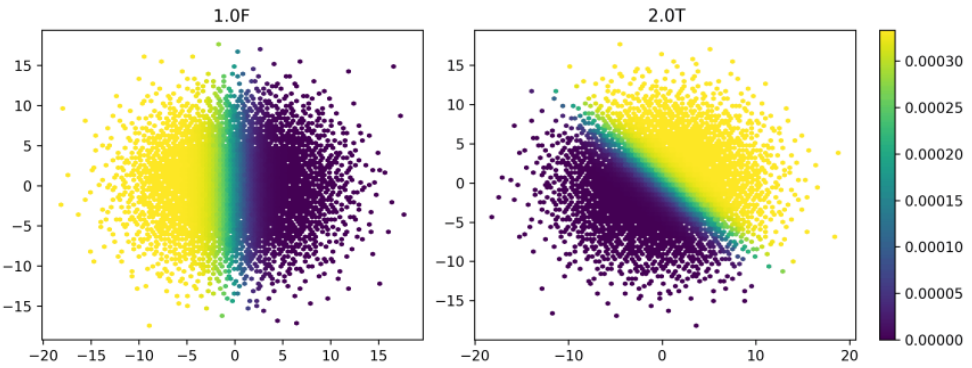


Fig. 10. Particle Approximation to Posteriors for $(\xi, y) = (1, F)$and $(\xi, y) = (2, T)$

### 4.5 Geometry of the Belief State Space

As opposed to Deisenroth's pioneering work and most other applications of Gaussian processes, the geometry of the index space is unintuitive. We denote the space of belief states reachable at trial $t$ as $\mathcal{X}_t$. First, $\mathcal{X}_t = \binom{\Xi \times Y}{t}$ where the binomial coefficient denotes the unordered Cartesian product. To give some intuition, for an outcome space $Y$ of a single element and $\Xi$ a closed interval, $\binom{\Xi \times Y}{t}$ is isomorphic to the interior of a $t$-dimensional simplex. For $\Xi, Y$ discrete sets, $|\mathcal{X}_t| = \binom{|\Xi||Y|+t-1}{t}$. Drovandi et al. performed a grid search of $\xi \in \{0.1, 0.2, \ldots, 2\}$ to find each optimal design point. Given an appropriate particle representation for the current belief state, the myopic utility was calculated for each reachable belief state by reweighting the current approximation according Bayes' theorem. Following the same methodology, solve the full sequential design problem via backward induction would require $\binom{40+t-2}{t-1}$ different particle approximations.

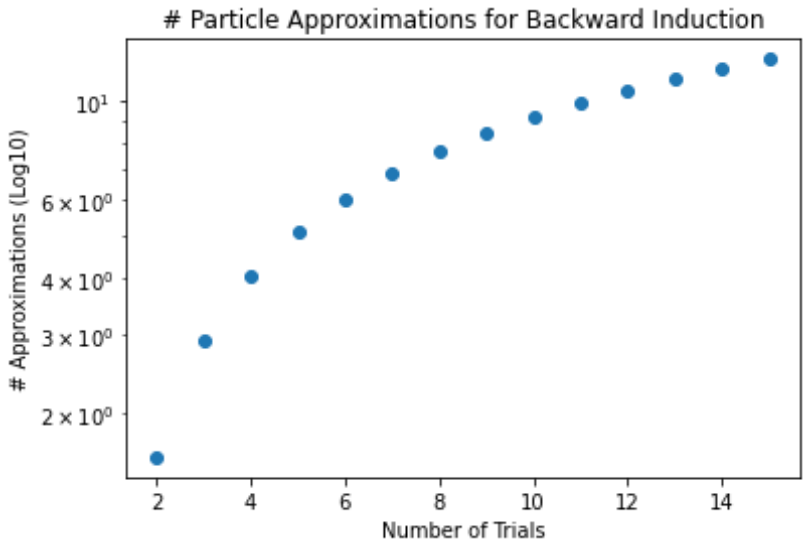


Fig. 11. Number of MCMC Samples Needed for Backward Induction

For $T \leq 4$, an exact solution via Backward Induction may be feasible, but for $T \geq 5$, the computation time is probably too much. For this reason, we consider the proposed algorithm successful if it can find sequential designs which meet or exceed the utility of the myopic design for $T \geq 5$.

We first need to find a set of initial states for each trial to base future learning upon. Unlike Deisenroth's application, determining whether the design region is approximately covered is not intuitive. To do so, we use two tools: the empirical variogram and Ripley's K statistic.

Recall from Equation 3.10 that for an Gaussian process with isotropic variance,

$$\text{Cov}\left(s_i, s_j\right) = \sigma^2 \rho\left(\boldsymbol{\phi}; \|s_i - s_j\|\right) + \tau^2$$

and

$$Var\left(s_i\right) = \tau^2$$

The function of the variance component due to spatial correlation $\rho$ is called the variogram. This can be estimated empirically much like a histogram by breaking the set of observed distances into bins and calculating covariance.

$$\hat{\rho}\left(d;\delta\right) = \frac{1}{2\left|N\left(d;\delta\right)\right|} \sum_{(i,j)\in N(d;\delta)} \left(u_i - u_j\right)^2$$

where $N\left(d;\delta\right) = \{(i,j) : |\|s_i - s_j\| - d| < \delta\}$. $\delta$ denotes the bin width. Consider the variogram in Figure 12 generated from 4430 design point-observation combinations generated via Latin Hypercube sampling using the D-optimality utility.

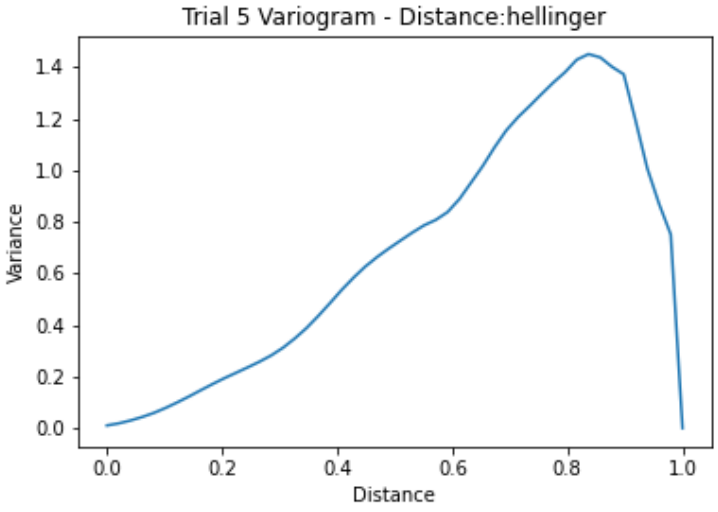


Fig. 12. Example Variogram

We can see that the covariance increases in an approximately linear function of the Hellinger distance. Since the variance increases quickly as a function of Hellinger distance, we need to make sure our initial sample has a sufficient number of observations for short distances. Ripley's K function is defined for a given spatial point process as the expected number of points within a radius $r$ divided by the overall density of the process [53]. While it is generally used to test the homogeneity of spatial point processes, we can use it to roughly gauge if we have sufficient coverage of the state space. Empirically, Ripley's K can be estimated by

$$\hat{K}(r) \propto \frac{1}{n} \sum_{i=1}^{N} w_i^{-1} \left| \{ s : \|s_i - s\| < r \setminus s_i \} \right|$$

where $\{w_i\}_{i=1}^{N}$ are weights used for edge correction on bounded spaces. How the boundaries of the design region and outcome space translate to the state space is hard to intuit, so we did not correct for the edges.

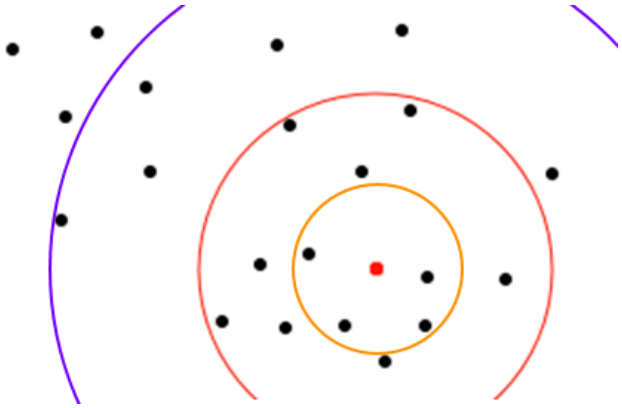

Fig. 13. Radii around a Point in a 2D Spatial Point Process

For the example state-space data set, we have the Ripley's plot in Figure 14 which shows there are on average 10 observations within a Hellinger distance radius of .1 and 100 observations for a radius of .2 which should be sufficient based on the variogram in Figure 12.

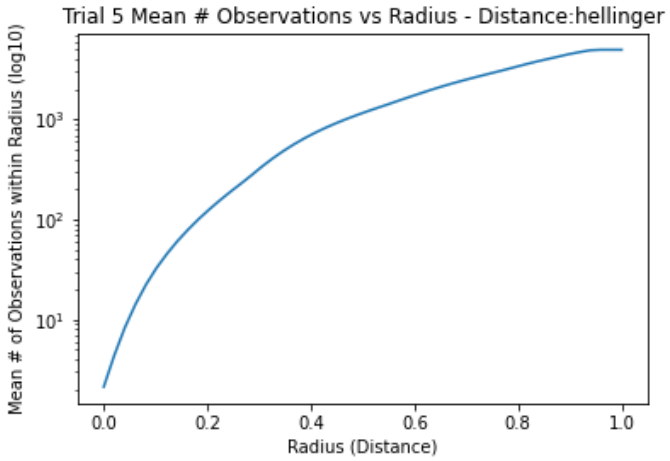


Fig. 14. Ripley's K-Like Plot for the D-Optimality Utility at $T = 5$

### 4.6 Results

The proposed methodology works for $T = 5$ and $T = 6$ with optimal designs found roughly meeting their myopic counter parts for the KL-divergence, D-optimality, and Robust Precision utilities, but fail for the target precision utility. Table 3 summarizes the results. All three utility functions criteria require variance stabilizing transformations for trials closer to the end of the experiment. Note that in addition to these results, in a practical setting, the myopic design can be used as part of the exploration policy. Much like Drovandi et al., we found myopic designs for the D-optimality utility can have high variability due to how flat the utility is

for early trials and subtle changes due to the importance sample. To combat this, the myopic design used as a benchmark is the best of eight calculations of the same design. Representations of all the designs and the expected utilities at each stage of the designs can be found in Appendix A, but an example is shown in Figure 15. In the same vein as Figures 1 and 8, these diagrams represent the optimal design points to be chosen at each trial of the experiment. Upon observing an outcome, the reader proceeds to the cell below and to the left if a failure is observed and below and to the right if a success is observed. For example, at trial 1, we choose $\xi_1 = .9$. The prior predictive has each outcome equally likely, so both cells below have equal width. If we observe a failure, we move to the left with $\xi_2 = 0.1$. Otherwise, we move to the right where we choose $\xi_2 = 2.0$.

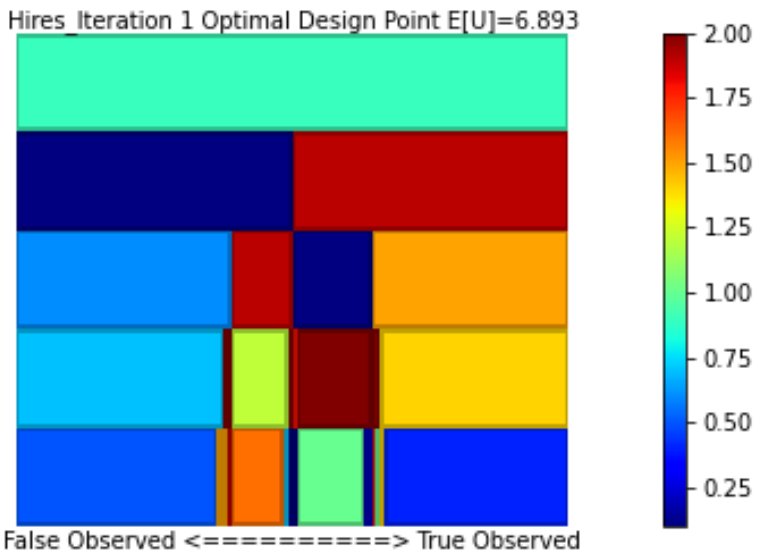


Fig. 15. Myopic Design for the Logistic Regression Model using the KL-Divergence Utility $T = 5$

In a similar fashion, we can visualize the expected utility at each point of the experiment such as in Figure 16. Here, we see all outcomes yield high information, but the outcomes with roughly even successes and failures perform worst. These scenarios narrow the belief state to a concentrated area near the origin since we do not think either $\theta_0$ or $\theta_1$ has much impact on the outcome. While this is new information, it is still roughly normal making it the closest to the prior of all belief states.

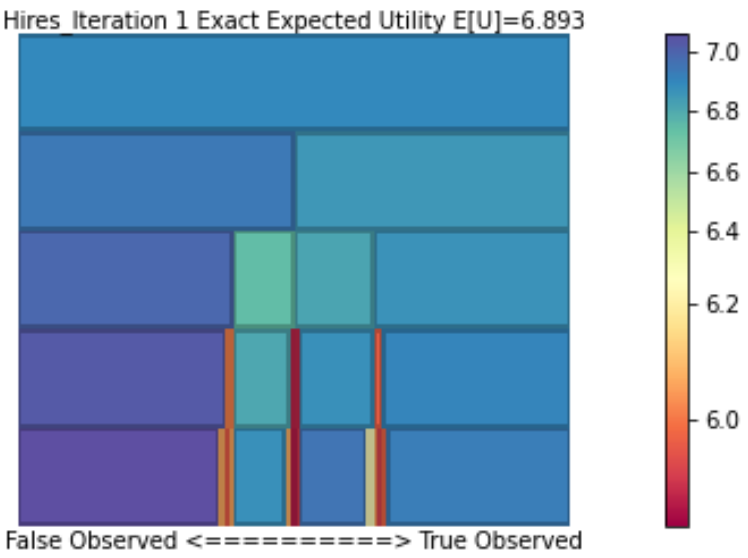


Fig. 16. Expected Utility for Sequential Design for the Logistic Regression Model using the KL-Divergence Utility $T = 5$

The Kernel Ridge Regression surrogate models were fit using 3-fold cross validation with a fixed ridge parameter of $k = 0.01$ for all utilities except D-optimality which was fit with $k = 0.005$. The sizes of $M_0$ and $M_i$ for $i > 0$ used can be found in Table 4. Like the original paper, we calculated optimal choices of $\xi$ via a grid search on $\xi$ a grid search of $\xi \in \{0.1, 0.2, \ldots, 2\}$ to find each optimal design point.

Table 3. Expected Utilities for Approximate BOSDs for $x_0 \sim N(0, 50)$

| Utility | Trials: | 5 | 6 |
|---|---|---|---|
| KL Div. | Myopic | 6.844 | 6.858 |
| KL Div. | Sequential | 6.893 | 6.875 |
| D-Opt. | Myopic | 0.019 | 0.028 |
| D-Opt. | Sequential | 0.019 | 0.027 |
| Precision | Myopic | 0.426 | 0.914 |
| Precision | Sequential | 0.340 | 0.387 |
| Robust Precision | Myopic | 1.830 | 2.140 |
| Robust Precision | Sequential | 1.838 | 2.113 |

Table 4. Initial Sample and Added Sample Sizes by Number of Trials

| Trials: | 5 | 6 |
|---|---|---|
| $M_0$ | 126 | 210 |
| $M_i$ for $i > 0$ | 128 | 256 |

As shown in Figure 17, the KL-divergence, D-optimality, and Robust Target Precision have variance zero for Hellinger distance zero while target precision utility has high variance for distance zero. This indicates the variance stabilizing transformation could not coerce the data set into a form the Hellinger distance can model. The transformations we considered were Yeo-Johnson transforms with a variable center which have an image of $\mathbb{R}$ [54]. When we considered the whole set of Yeo-Johnson transforms, we found cases where surrogate models would predict values outside the

range of the image make them unable to be inverted. Other transforms of the data may exist that allow the target precision to be modeled with Hellinger distance.

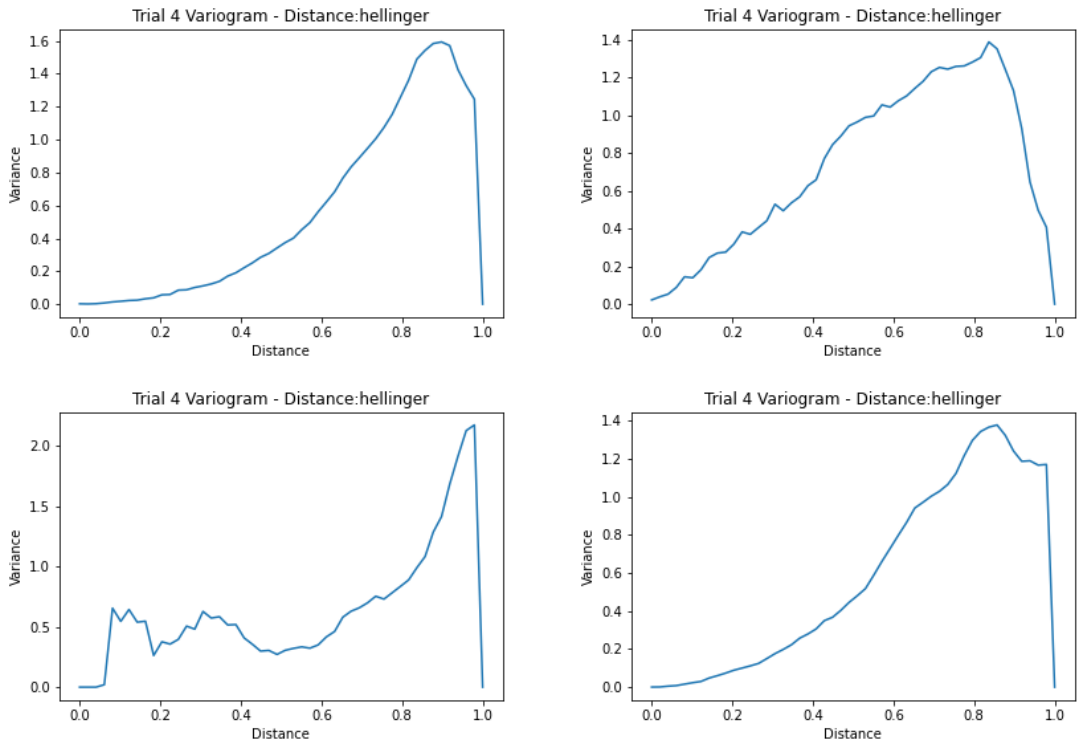


Fig. 17. Comparison of Variograms for $T = 4$ Surrogate Model from the $T = 5$ Logistic Regression Design for KL-Divergence (Top Left), D-Optimality (Top Right), Target Precision (Bottom Left), and Robust Target Precision (Bottom Right)

## 4.7 Conclusions

The proposed usage of a single importance sample to compare multiple belief states works succeeds at allowing the methodology from Section 3 to apply to a wider class of models. In the example shown, the fully sequential designs found by

the proposed algorithm greatly exceed the expected utility of the myopic designs provided the surrogate model fits. Researchers more adept at programming may be able to reduce runtimes further by developing CUDA kernels specific to this application. A potential issue with the proposed methodology is the number of initial belief states needed to start the learning process. With the particle approximation, sampling error may not allow for models to account for very large distance values between belief states. Additionally, covariant metrics cannot compare belief state whose supports do not overlap such as disjoint uniform distributions. Metrics such as Wasserstein distance are expensive computationally, but do not require probability mass to overlap. The representation using transport maps proposed by Huan and Marzouk may translate well to distance-based modeling via Wasserstein distance [46]. Approximations to structural kernels may be a viable option. Muandet et al. give an estimator for the Gassian RBF kernel between two empirical distributions with error $O\left(m^{-\frac{1}{2}}\right)$ where $m$ is sample size though it requires taking an inverse and a determinant [55]. Alternatively, future work could investigate the importance sample of each trial. Given a set of distributions represented as weights on a shared sample, minimizing the maximum KL-Divergence from any weighted sample to a mixture of the weighted samples is convex and can be formulated as a conic program [56]. While we found the noise due to random sampling too hard to rectify when attempting this approach, someone more familiar with stochastic optimization may have success. As mentioned before in Section 4.4, a subgradient approach could also theoretically work though again we had little success.

# CHAPTER 5

# BATCH BAYESIAN OPTIMAL SEQUENTIAL DESIGNS VIA APPROXIMATE DYNAMIC PROGRAMMING

The batch sequential case is an alteration of the BOSD problem where batches of independent trials are chosen in sequence. While the set of reachable belief states is the same between the batch and full sequential case, the marginal probabilities of visiting each belief state differs. Note that full sequential and static designs are special cases of batch sequential designs with batch sizes of 1 and $T$ respectively. While it is possible for batch sizes to vary dynamically within a design, this dissertation focuses on the case of fixed, predetermined batches. For an experiment with $T$ trials and $B$ batches we denote the number of trials in batch $b$ as $T_b$ with $\sum_{b=1}^{B} T_b$. Consider each batch then as a single trial. For batch $b$, we have $\xi_b \coloneqq (\xi_1, \ldots, \xi_{T_b})$ and $y_b \coloneqq (y_1, \ldots, y_{T_b})$ with likelihood defined in Equation 5.1.

$$p\left(y_b|\xi_b, x_b\right) \coloneqq p\left(x_t\right) \prod_{t=1}^{T_b} p\left(y_t|\xi_t, x_t\right) \tag{5.1}$$

Under this construction, we apply can Algorithm 3 to the batch problem by treating it as a regular experiment with $B$ trials.

## 5.1 Application

We revisit the model from Section 4 for the same choices of utility. Optimal designs were calculated for experiments with trials $T = 6$ dividing each up into batches

Table 5. Initial Sample and Added Sample Sizes by Number of Trials

| Iteration | Simulations |
|---|---|
| $i = 0$ | 210 |
| $i > 0$ | 256 |

of size two. Due to expanding to a larger design region, we changed algorithms from a grid search to Nelder-Mead using a soft boundary penalty function for a fixed number of 40 iterations. A fixed number of iterations was used rather than other convergence criteria to make it simpler to implement in parallel for the GPU. Recall that the set of designs for a fixed number of trials is a simplex. With this in mind, we performed the Nelder-Mead algorithm with $T_b + 1$ starts with a starting simplex in each corner of the design space.

### 5.2 Results and Conclusions

The proposed methodology performed well for all cases. In all cases, the optimal batch sequential design roughly met or beat the myopic design in performance. Run times for the batch sequential case were slower though this is expected. The number of evaluations of the GP model increases exponentially with batch size due to the increase in outcomes of each belief state.

The results are summarized in Table 6 and graphical representations of the optimal designs found and their expected utilities can be found in Appendix B.

These diagrams are identical in how they are read to the ones used in 4.6. In Figure 18, notice how the choice of $xi_2$ is the same regardless of the outcome of $xi_1$

Table 6. Expected Utilities for Approximately Optimal Batch Sequential Designs for $x_0 \sim N(0, 50)$

| Utility | Algorithm | $E[U]$ |
|---|---|---|
| KL Div. | Myopic | 6.684 |
| KL Div. | Sequential | 6.652 |
| D-Opt. | Myopic | 0.116 |
| D-Opt. | Sequential | 0.131 |
| Precision | Myopic | 0.067 |
| Precision | Sequential | 0.067 |
| Robust Precision | Myopic | 2.114 |
| Robust Precision | Sequential | 2.124 |

and how the predicted probability of a success versus a failure remains constant for the first two trials. This is due to the independence of trials within a batch.

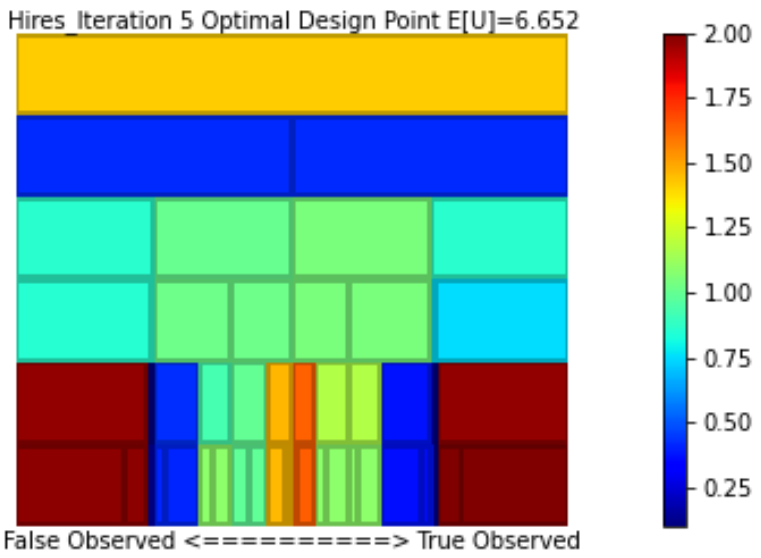


Fig. 18. Batch Sequential Design for the Logistic Regression Model using the KL-Divergence Utility $T = 6$

All future avenues of research discussed in Section 4.7 apply here as well. Additionally, tackling the dynamic batch size case without using separate models for each batch size choice could prove a useful and interesting direction for research.

# CHAPTER 6

# CONCLUSIONS AND FUTURE WORK

In this dissertation, we propose the novel application of Gaussian process models indexed on belief states as surrogate models for approximating Bayesian optimal sequential designs via approximate dynamic programming. We tested this methodology for conjugate models, non-conjugate models, and batch sequential designs for non-conjugate models. The proposed methodology for finding approximate solutions to Bayesian optimal sequential designs performed adequately in all test cases examined. Modeling utility via distances between distributions allows for a general method of finding approximately optimal Bayesian sequential designs. Compared to recent methods which take design points and outcomes as inputs directly, using the distances between the belief states to model the utility allows for more direct comparisons to be made between simulated belief states. Ultimately, the utility of the proposed methodology hinges on the speed of the distance calculation. Conjugate models work very well if they have a parametric formula for a distance, but also allow for simpler execution numerical methods since the distribution is exactly known. In other cases where the distribution must be approximated, Monte Carlo and grid methods which represent all distributions on a unified "basis" will not be exact enough for all cases without careful choice of proposal distributions. For example, complex Markov processes which require more specific filtering schemes may perform poorly using these methods since where reachable distributions are concentrated is

not easy to calculate without brute force simulation. Future avenues of work may consider representing belief states in a kernel mean embedding to solve the distance and approximation problems at the same time [57]. Representing a distribution as the mean of a kernel function over the support allows for computational tricks that allow for easy representation of posterior distributions and several distance metrics to use in comparing distributions [57]. Embedding the mean does require taking the expectation over the kernel either analytically or numerically which brings the same computation problems we tried to avoid. Kernels functions of statistics are certainly much faster though applicability will depend on the model and could require substantial tuning due to differences in scales of each statistic. Variational approximations are mentioned briefly, but could use a more thorough examination to see if there are cases outside the variational approximation to the normal with known distance formulas or fast distance algorithms. Future work on fast distance calculations for conjugate models without a known distance formula is a promising direction for research with many applications for commonly used models as discussed in Section 3.8. As mentioned in Section 4.7, finding a minimax proposal distribution from among the set of mixtures of visited belief states is convex when represented as weights of a single sample. This problem may have other applications in other information theoretic disciplines. Additionally, as noted in Section 5.2, solving the dynamic batch size case of the BOSD problem without separate models for each batch size is potentially useful in many applications.

Algorithmically, there are several improvements that can be made to the algorithm to improve runtimes. When using a grid approximation and a discretization

of the outcome space, the distances from posterior distributions of belief states visited during optimization can be stored in lookup tables which can be stored on disk until needed in the algorithm. While the code used benefits from GPU acceleration via CuPy, no custom CUDA code was used [50]. Custom CUDA kernels would undoubtedly improve runtimes.

Especially in cases with non-terminal rewards such as the number of successful treatments or Markov cost penalties, sequential methods have the potential to greatly outperform their static counterparts. As more work is done in developing algorithms to quickly compute approximate solutions for Bayesian optimal sequential designs, hopefully they will be given more consideration in practical applications. While the curse of dimensionality can never truly be solved, the computing power to tackle these problems is available.

# Appendix A

## OPTIMAL SEQUENTIAL DESIGNS FOR LOGISTIC REGRESSION MODEL

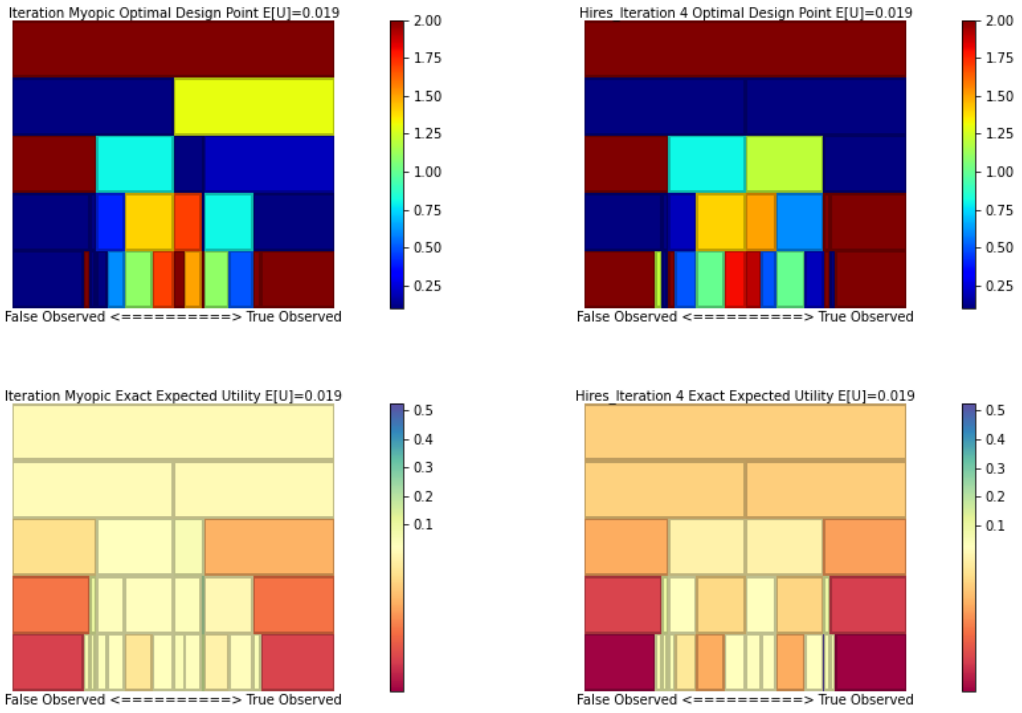


Fig. 19. Myopic and Full Sequential Designs for the Logistic Regression Model in Section 4 for the D-Optimality Utility $T = 5$

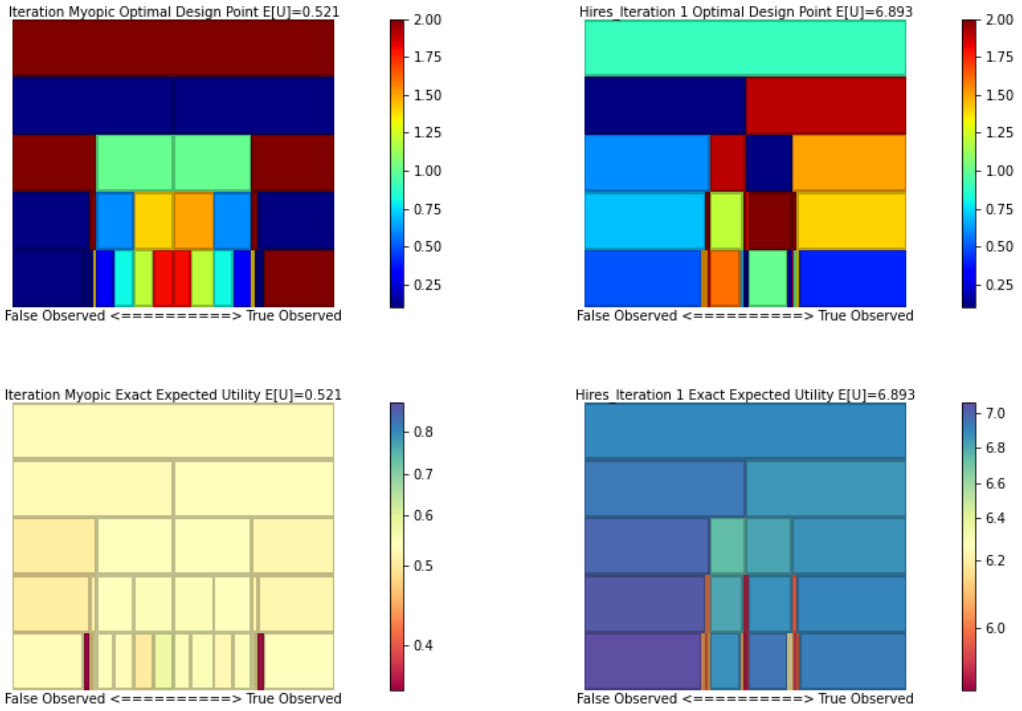


Fig. 20. Myopic and Full Sequential Designs for the Logistic Regression Model in Section 4 for the KL-Divergence Utility $T = 5$

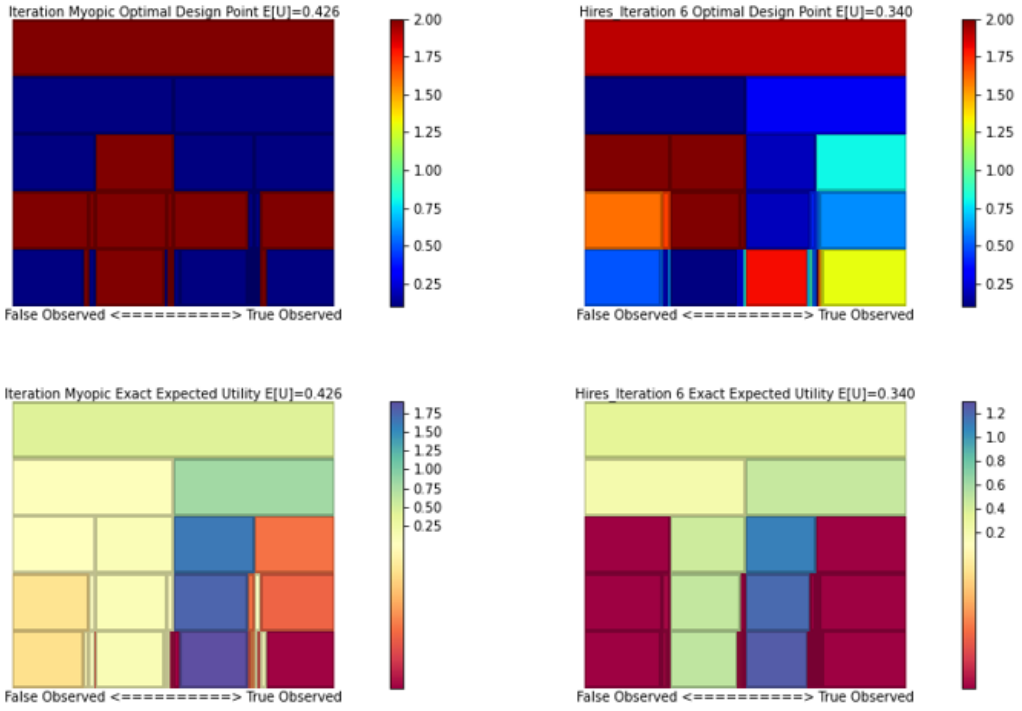


Fig. 21. Myopic and Full Sequential Designs for the Logistic Regression Model in Section 4 for the Target Precision Utility $T = 5$

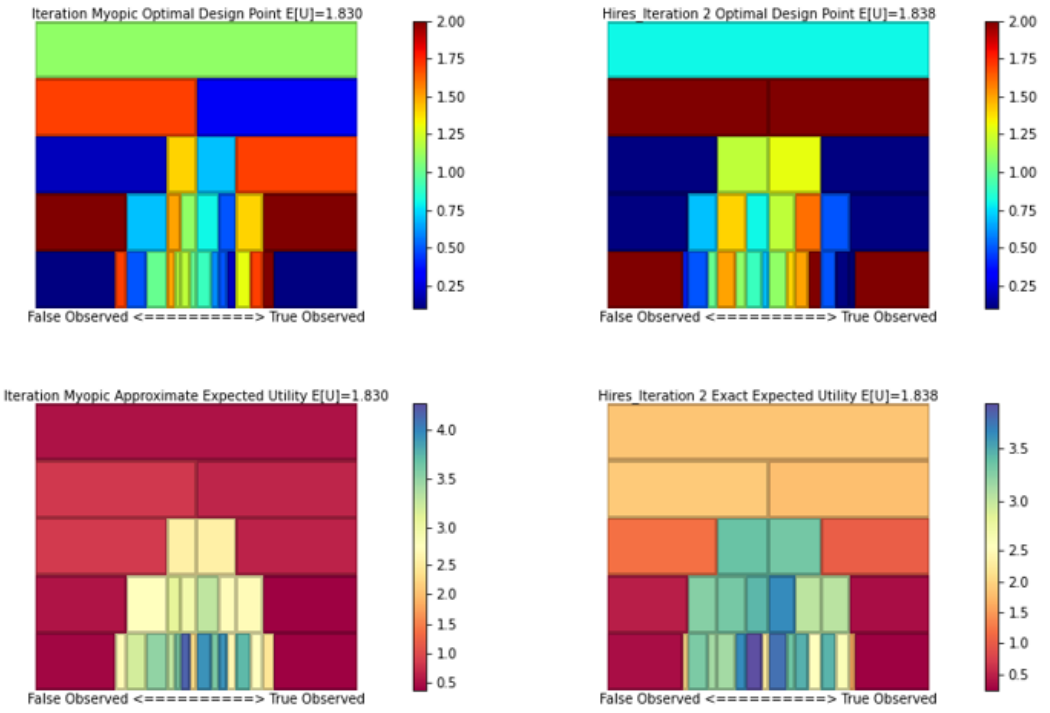


Fig. 22. Myopic and Full Sequential Designs for the Logistic Regression Model in Section 4 for the Robust Target Precision Utility $T = 5$

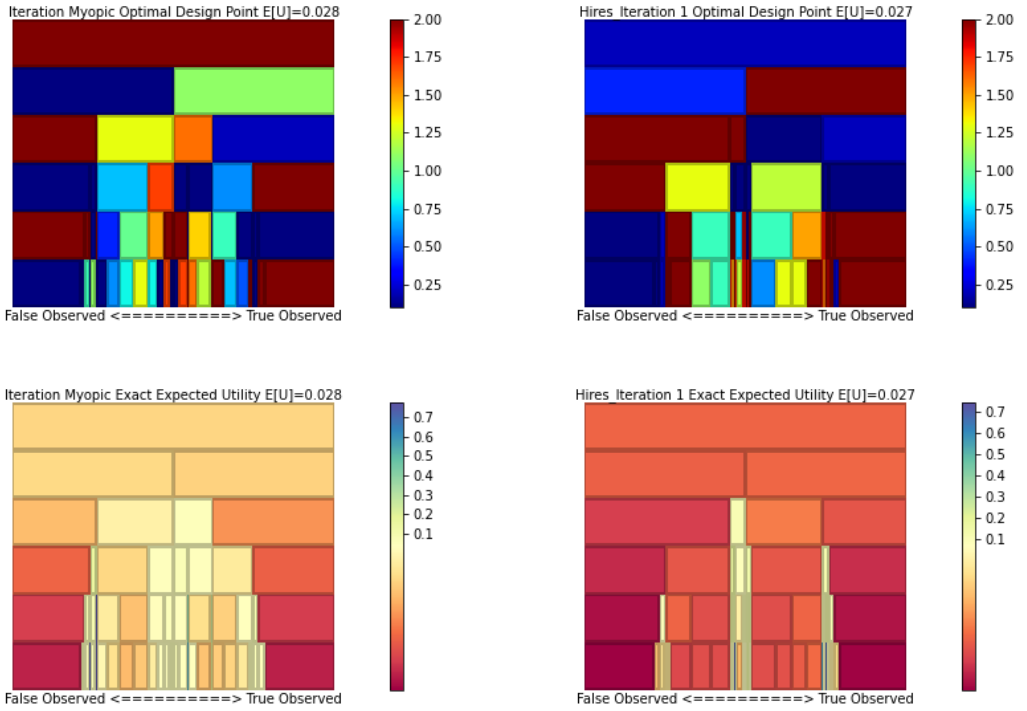


Fig. 23. Myopic and Full Sequential Designs for the Logistic Regression Model in Section 4 for the D-Optimality Utility $T = 6$

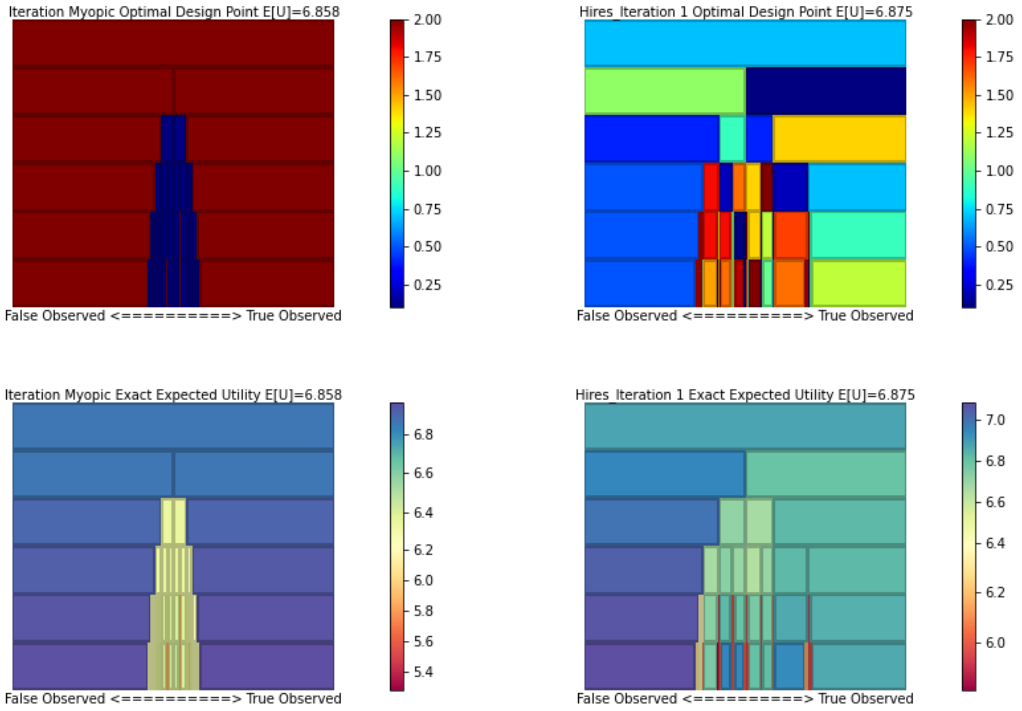


Fig. 24. Myopic and Full Sequential Designs for the Logistic Regression Model in Section 4 for the KL-Divergence Utility $T = 6$

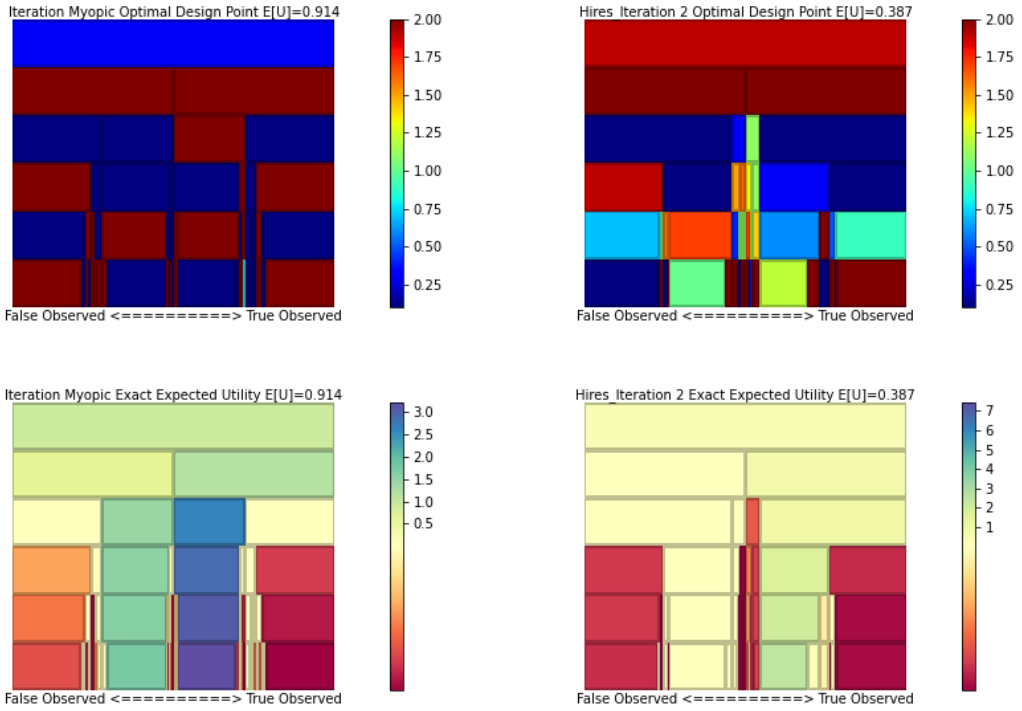


Fig. 25. Myopic and Full Sequential Designs for the Logistic Regression Model in Section 4 for the Target Precision Utility $T = 6$

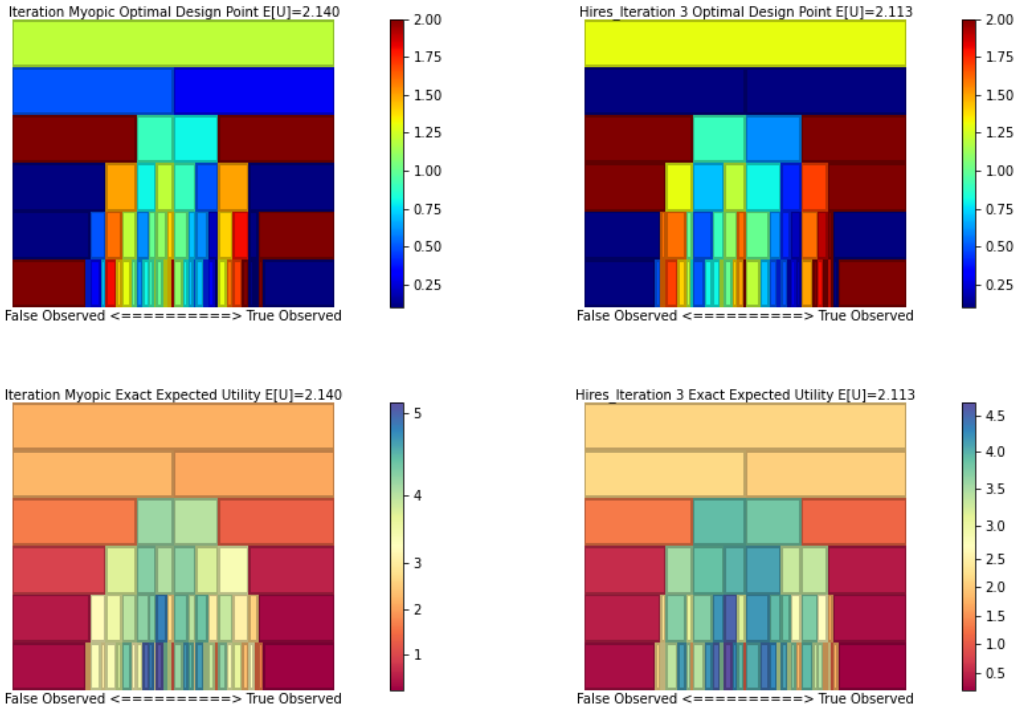


Fig. 26. Myopic and Full Sequential Designs for the Logistic Regression Model in Section 4 for the Robust Target Precision Utility $T = 6$

## Appendix B

# OPTIMAL BATCH SEQUENTIAL DESIGNS FOR LOGISTIC REGRESSION MODEL

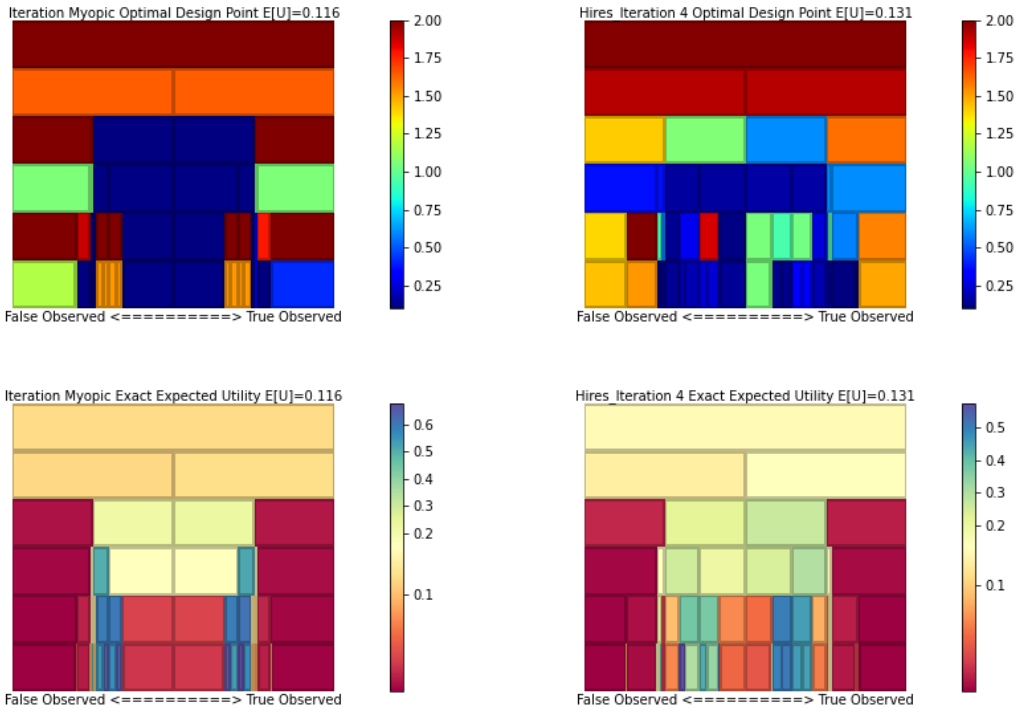


Fig. 27. Myopic and Full Sequential Designs for the Logistic Regression Model in Section 5 for the D-Optimality Utility $T = 6$

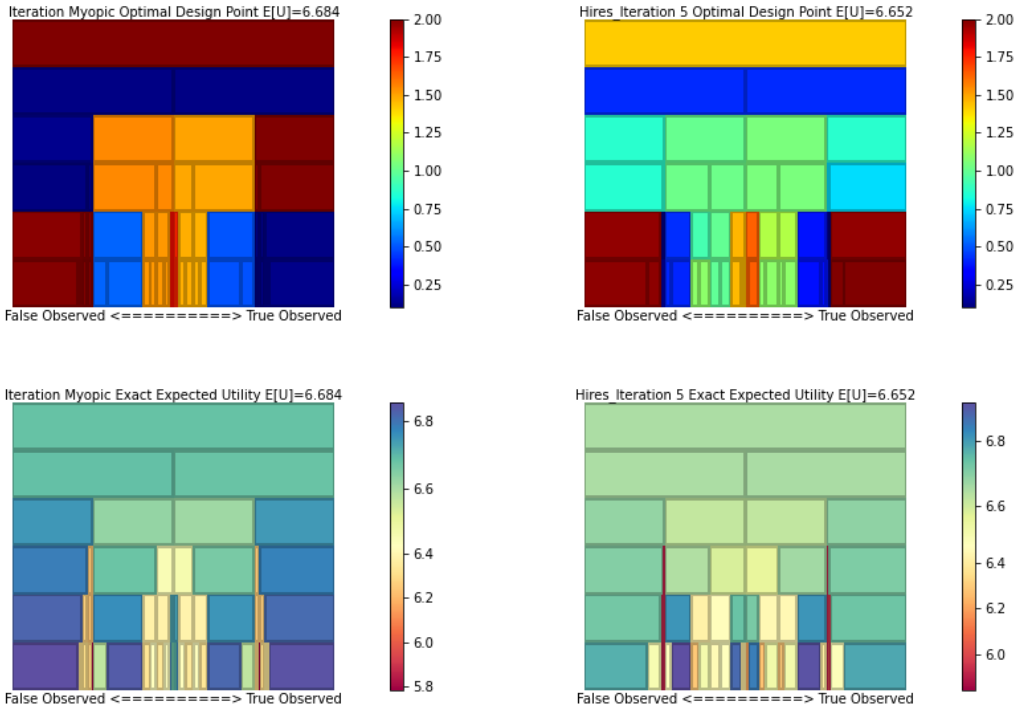


Fig. 28. Myopic and Full Sequential Designs for the Logistic Regression Model in Section 5 for the KL-Divergence Utility $T = 6$

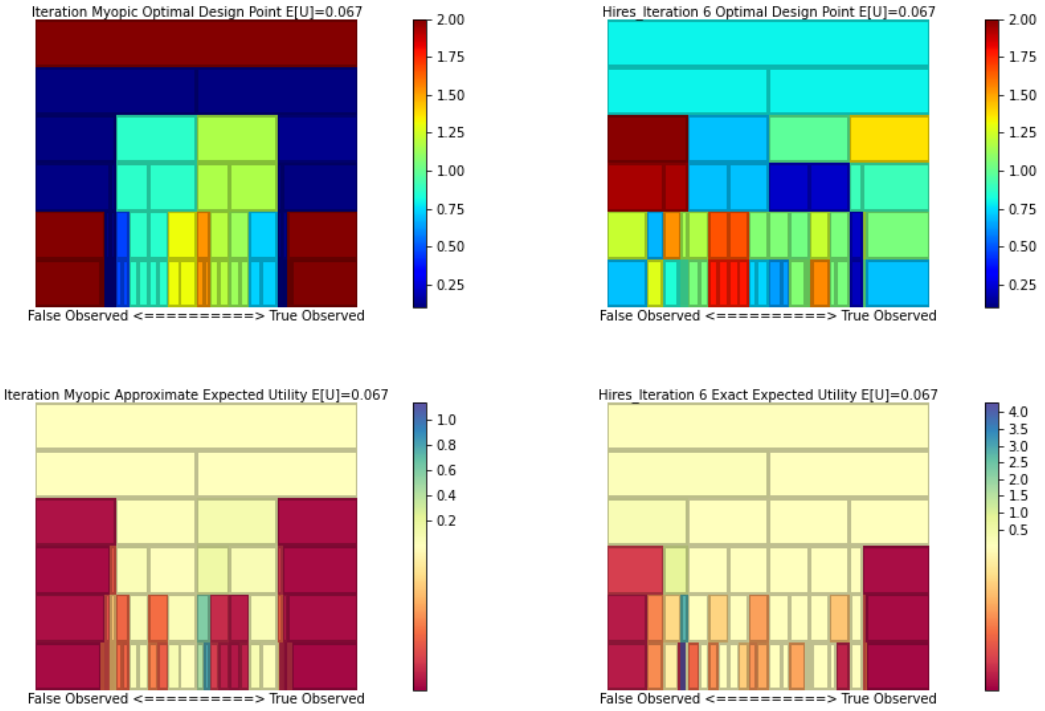


Fig. 29. Myopic and Full Sequential Designs for the Logistic Regression Model in Section 5 for the Target Precision Utility $T = 6$

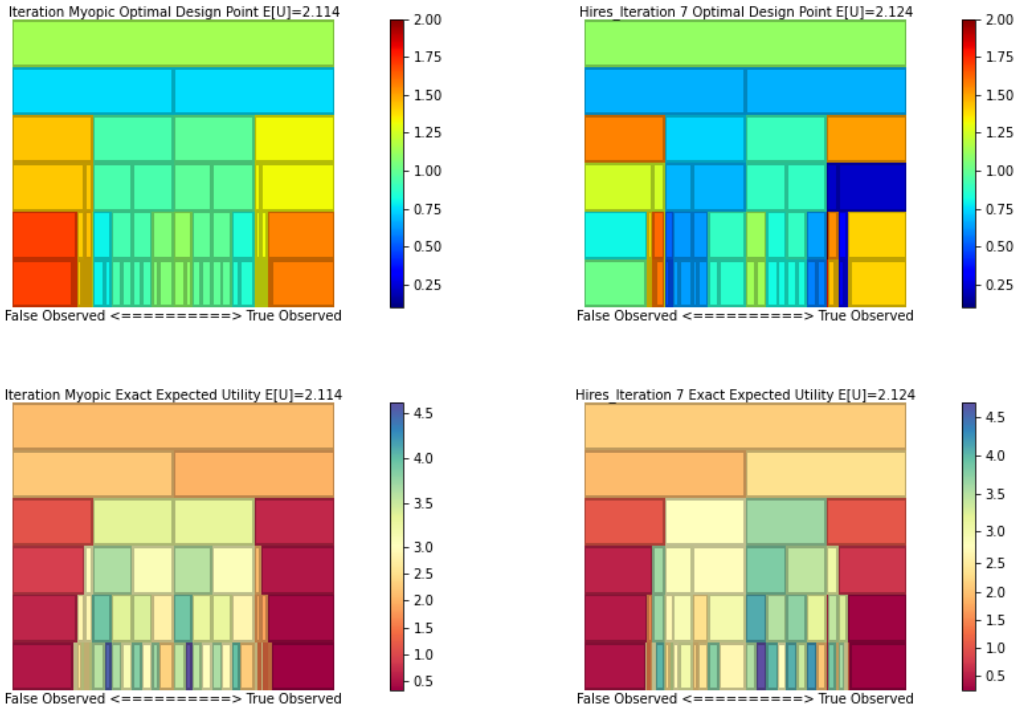


Fig. 30. Myopic and Full Sequential Designs for the Logistic Regression Model in Section 5 for the Robust Target Precision Utility $T = 6$

# REFERENCES


[1] Kathryn Chaloner and Isabella Verdinelli. "Bayesian experimental design: A review". In: *Statistical Science* (1995), pp. 273–304.

[2] Shein-Chung Chow and Mark Chang. "Adaptive design methods in clinical trials – a review". In: *Orphanet Journal of Rare Diseases* 3.1 (May 2008), p. 11. ISSN: 1750-1172. DOI: 10.1186/1750-1172-3-11. URL: https://doi.org/10.1186/1750-1172-3-11.

[3] Elizabeth G. Ryan et al. "A Review of Modern Computational Algorithms for Bayesian Optimal Design". In: *International Statistical Review* 84.1 (2016), pp. 128–154. ISSN: 1751-5823. DOI: 10.1111/insr.12107. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/insr.12107 (visited on 03/19/2019).

[4] Peter Whittle. "Multi-armed bandits and the Gittins index". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 42.2 (1980), pp. 143–149.

[5] Sofía S. Villar, Jack Bowden, and James Wason. "Multi-armed Bandit Models for the Optimal Design of Clinical Trials: Benefits and Challenges." eng. In: *Statistical science : a review journal of the Institute of Mathematical Statistics* 30.2 (2015), pp. 199–215. ISSN: 0883-4237 2168-8745. DOI: 10.1214/14-STS504.

[6] Warren Buckler Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Hoboken, UNITED STATES: John Wiley & Sons, Incorporated, 2011. ISBN: 978-1-118-02915-2. URL: http://ebookcentral.proquest.com/lib/vcu/detail.action?docID=697550 (visited on 12/07/2019).

[7] Xun Huan and Youssef M. Marzouk. "Sequential Bayesian optimal experimental design via approximate dynamic programming". In: *arXiv preprint arXiv:1604.08320* (2016).

[8] Wanggang Shen and Xun Huan. *Bayesian Sequential Optimal Experimental Design for Nonlinear Models Using Policy Gradient Reinforcement Learning*. arXiv:2110.15335 [cs, stat]. Mar. 2022. DOI: 10.48550/arXiv.2110.15335. URL: http://arxiv.org/abs/2110.15335 (visited on 08/07/2022).

[9] Adam Foster et al. "Deep adaptive design: Amortizing sequential bayesian experimental design". In: *International Conference on Machine Learning*. PMLR. 2021, pp. 3384–3395.

[10] Tom Blau et al. "Optimizing Sequential Experimental Design with Deep Reinforcement Learning". In: *International Conference on Machine Learning*. PMLR. 2022, pp. 2107–2128.

[11] Richard Bellman. "The theory of dynamic programming". In: *Bulletin of the American Mathematical Society* 60.6 (1954), pp. 503–515.

[12] A. P. Dempster. "The Dempster–Shafer calculus for statisticians". en. In: *International Journal of Approximate Reasoning*. In Memory of Philippe Smets (1938–2005) 48.2 (June 2008), pp. 365–377. ISSN: 0888-613X. DOI: 10.1016/

j.ijar.2007.03.004. URL: https://www.sciencedirect.com/science/article/pii/S0888613X07000278 (visited on 08/07/2022).

[13] Olivier Sigaud and Olivier Buffet. *Markov decision processes in artificial intelligence*. John Wiley & Sons, 2013.

[14] Kevin P Murphy. "A survey of POMDP solution techniques". In: *environment* 2.10 (2000).

[15] Sebastian Thrun. "Monte carlo pomdps". In: *Advances in neural information processing systems* 12 (1999).

[16] Anthony E Brockwell and Joseph B Kadane. "A gridding method for Bayesian sequential decision problems". In: *Journal of Computational and Graphical Statistics* 12.3 (2003), pp. 566–584.

[17] Peter Müller et al. "A Bayesian decision-theoretic dose-finding trial". In: *Decision analysis* 3.4 (2006), pp. 197–207.

[18] Amir Ali Nasrollahzadeh and Amin Khademi. "Optimal stopping of adaptive dose-finding trials". In: *Service Science* 12.2-3 (2020), pp. 80–99.

[19] Mario J. Miranda and Paul L. Fackler. *Applied Computational Economics and Finance*. Cambridge, UNITED STATES: MIT Press, 2002. ISBN: 978-0-262-27992-5. URL: http://ebookcentral.proquest.com/lib/vcu/detail.action?docID=3338831 (visited on 12/10/2019).

[20] Fatemeh Nargesian et al. "Learning Feature Engineering for Classification." In: *Ijcai*. 2017, pp. 2529–2535.

[21] Peter Flach. *Machine learning: the art and science of algorithms that make sense of data*. Cambridge University Press, 2012.

[22] Daniel G. Krige. "A statistical approach to some basic mine valuation problems on the Witwatersrand". In: *Journal of the Southern African Institute of Mining and Metallurgy* 52.6 (1951), pp. 119–139.

[23] Zoltán Szabó et al. "Learning theory for distribution regression". In: *The Journal of Machine Learning Research* 17.1 (2016), pp. 5272–5311.

[24] Matthias Hein and Olivier Bousquet. "Hilbertian Metrics and Positive Definite Kernels on Probability Measures". In: (), p. 8.

[25] Isaac J. Schoenberg. "Metric spaces and positive definite functions". In: *Transactions of the American Mathematical Society* 44.3 (1938), pp. 522–536.

[26] Rajendra Bhatia. *Positive Definite Matrices*. Princeton, UNITED STATES: Princeton University Press, 2007. ISBN: 978-1-4008-2778-7. URL: `http://ebookcentral.proquest.com/lib/vcu/detail.action?docID=445446` (visited on 12/09/2019).

[27] Sudipto Banerjee, Bradley P. Carlin, and Alan E. Gelfand. *Hierarchical modeling and analysis for spatial data*. Chapman and Hall/CRC, 2014.

[28] Leandro Pardo. *Statistical inference based on divergence measures*. Chapman and Hall/CRC, 2018.

[29] U. Kamps. "Distance measures in a one-parameter class of density functions". In: *Communications in Statistics - Theory and Methods* 17.6 (Jan. 1988),

pp. 2013–2019. ISSN: 0361-0926. DOI: 10.1080/03610928808829729. URL: https://doi.org/10.1080/03610928808829729 (visited on 11/02/2019).

[30] Nachman Aronszajn. "Theory of reproducing kernels". In: *Transactions of the American mathematical society* 68.3 (1950), pp. 337–404.

[31] Grace Wahba. *Spline models for observational data*. SIAM, 1990.

[32] Carl Boettiger, Marc Mangel, and Stephan Munch. "Avoiding tipping points in fisheries management through Gaussian process dynamic programming". In: *Proceedings of the Royal Society B: Biological Sciences* 282.1801 (2015), p. 20141631.

[33] Julie M. Walker, Allison M. Okamura, and Mykel J. Kochenderfer. "Gaussian Process Dynamic Programming for Optimizing Ungrounded Haptic Guidance". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Oct. 2018, pp. 8758–8764. DOI: 10.1109/IROS.2018.8594395.

[34] Marc Peter Deisenroth, Carl Edward Rasmussen, and Jan Peters. "Gaussian process dynamic programming". In: *Neurocomputing* 72.7 (Mar. 2009), pp. 1508–1524. ISSN: 09252312. DOI: 10.1016/j.neucom.2008.12.019. URL: https://linkinghub.elsevier.com/retrieve/pii/S0925231209000162 (visited on 09/14/2019).

[35] Joaquim Barris and Pilar Garcia Almirall. *A density function of the appraisal value. Calculation and evaluation of the empirical density function of the appraisal value based on comparison method, spatial correlation techniques, re-*

*sampling methods, compliant with the Spanish legal framework*. Tech. rep. European Real Estate Society (ERES), 2011.

[36] Sabine Bellstedt et al. "The SLUGGS survey: using extended stellar kinematics to disentangle the formation histories of low-mass S0 galaxies". In: *Monthly Notices of the Royal Astronomical Society* 467.4 (2017), pp. 4540–4557.

[37] Hanefi Bayraktar and F. Sezer Turalioglu. "A Kriging-based approach for locating a sampling site—in the assessment of air quality". In: *Stochastic Environmental Research and Risk Assessment* 19.4 (2005), pp. 301–305.

[38] Jay D Martin. "Computational improvements to estimating kriging metamodel parameters". In: *Journal of Mechanical Design* 131.8 (2009).

[39] Yuchen Zhang, John Duchi, and Martin Wainwright. "Divide and conquer kernel ridge regression: A distributed algorithm with minimax optimal rates". In: *The Journal of Machine Learning Research* 16.1 (2015), pp. 3299–3340.

[40] Ahmed El Alaoui and Michael W Mahoney. "Fast randomized kernel methods with statistical guarantees". In: *arXiv preprint arXiv:1411.0306* (2014).

[41] Warren B. Powell and Jun Ma. "A review of stochastic algorithms with continuous value function approximation and some new approximate policy iteration algorithms for multidimensional continuous applications". In: *Journal of Control Theory and Applications* 9.3 (2011), pp. 336–352.

[42] Pauli Virtanen et al. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17 (2020), pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

[43] James O Berger. *Statistical decision theory and Bayesian analysis*. eng. 2nd ed.. Springer series in statistics. New York: Springer-Verlag, 1985. ISBN: 0-387-96098-8.

[44] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. "Variational Inference: A Review for Statisticians". In: *Journal of the American Statistical Association* 112.518 (Apr. 2017), pp. 859–877. ISSN: 0162-1459. DOI: 10.1080/01621459.2017.1285773. URL: https://amstat.tandfonline.com/doi/full/10.1080/01621459.2017.1285773 (visited on 06/05/2019).

[45] Francois Bachoc et al. "Gaussian processes with multidimensional distribution inputs via optimal transport and Hilbertian embedding". In: *arXiv:1805.00753 [stat]* (May 2018). URL: http://arxiv.org/abs/1805.00753 (visited on 09/08/2019).

[46] Xun Huan and Youssef Marzouk. "Sequential Optimal Experimental Design using Transport Maps." In: (2017). URL: https://www.osti.gov/biblio/1463964.

[47] Arnaud Doucet and Adam Johansen. "A Tutorial on Particle Filtering and Smoothing: Fifteen Years Later". In: *Handbook of Nonlinear Filtering* 12 (Jan. 2009).

[48] Hera Y He and Art B Owen. "Optimal mixture weights in multiple importance sampling". In: *arXiv preprint arXiv:1411.3954* (2014).

[49] Stephen Boyd and Almir Mutapcic. "Stochastic subgradient methods". In: *Lecture Notes for EE364b, Stanford University* (2008).

[50] ROYUD Nishino and Shohei Hido Crissman Loomis. "Cupy: A numpy-compatible library for nvidia gpu calculations". In: *31st confernce on neural information processing systems* 151.7 (2017).

[51] Charles R. Harris et al. "Array programming with NumPy". en. In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/s41586-020-2649-2. URL: https://www.nature.com/articles/s41586-020-2649-2 (visited on 07/24/2022).

[52] Christopher C. Drovandi, James M. McGree, and Anthony N. Pettitt. "Sequential Monte Carlo for Bayesian sequentially designed experiments for discrete data". In: *Computational Statistics & Data Analysis* 57.1 (2013), pp. 320–335.

[53] Philip M Dixon. *Ripley's K Function*. eng. 2014.

[54] In-Kwon Yeo and Richard A Johnson. "A new family of power transformations to improve normality or symmetry". In: *Biometrika* 87.4 (2000), pp. 954–959.

[55] Krikamol Muandet et al. "Learning from distributions via support measure machines". In: *Advances in neural information processing systems* 25 (2012).

[56] Mosek ApS. "Mosek optimization toolbox for matlab". In: *User's Guide and Reference Manual, Version* 4 (2019).

[57] Krikamol Muandet et al. "Kernel mean embedding of distributions: A review and beyond". In: *Foundations and Trends® in Machine Learning* 10.1-2 (2017), pp. 1–141.

**Vita**

2014 B.S. in Mathematics with Concentration in Statistics and Minor in Computer Science, University of Maryland, College Park

2016-2019 Research Assistant, NASA Langley

2017 M.S. in Mathematics, Virginia Commonwealth University

2022 Ph.D. in Statistical Sciences and Operations Research, Virginia Commonwealth University

Publications

- J. Burris, S. R. Wilson, D. J. Edwards, and K. M. Ballard, "Modeling Uncertainty in Time and Fuel Benefit Estimation for TASAR Operational Evaluation," NASA/TM-2019-220420, Nov. 2019. Accessed: Jul. 26, 2022. [Online]. Available: https://ntrs.nasa.gov/citations/20190033461

- J. Burris, K. Ballard, S. R. Wilson, and D. J. Edwards, "Visualization and comparison of aircraft trajectories using Gaussian-Smoothed heatmaps," Quality Engineering, Sep. 2021, Accessed: Jul. 26, 2022. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/08982112.2021.1976795