Neural Networks as Tools for Posterior Estimation and Inference

Thayer Fisher

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2023

Reading Committee:
Noah Simon, Chair
Frederick Matsen IV, Chair
Ali Shojaie

Program Authorized to Offer Degree:
Biostatistics

University of Washington

Abstract

Neural Networks as Tools for Posterior Estimation and Inference

Thayer Fisher

Co-Chairs of the Supervisory Committee:
Noah Simon
Department of Biostatistics

Frederick Matsen IV
Department of Statistics

In this dissertation, we will discuss three applications for neural networks in the paradigm of Bayesian estimation and inference. In Chapter 2, we describe a likelihood-free method of estimating posterior quantiles using recurrent neural networks. This method is particularly useful for time series data with high-dimensional latent random variables. In Chapter 3, we propose a method for optimizing Bayesian adaptive enrichment design clinical trials using reinforcement learning. Through the simulation of many trials, we use policy gradient descent to train a neural network to optimally incorporate existing patient information into a simple adaptive enrollment criterion. Finally, in Chapter 4, we describe a mechanistically explicit forward model for Somatic Hypermutation (SHM). We estimate the parameters regulating this model with a neural network and manually selected summary statistics.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

Thank you to my advisors, Noah Simon and Frederick Matsen for their guidance and unwavering intellectual curiosity. Thank you to Marco Carone and Alex Luedtke for their work on the methodology in Chapter 2. Thank you to Julia Fukuyama for her collaboration on the work in Chapter 4. Thank you to all the members of the Matsen group and SLAB Lab, and all the students in the Biostatistics department.

DEDICATION

to my family, Mom, Dad, Eva, and Micah, who made this all possible

Chapter 1

INTRODUCTION

In Bayesian data analysis, it is often important to evaluate quantiles of the posterior distribution of a parameter of interest (e.g., to form posterior intervals). In multi-dimensional problems, when non-conjugate priors are used, this is often difficult, generally requiring either an analytic or a sampling-based approximation, such as the Markov chain Monte Carlo, approximate Bayesian computation, or variational inference. In Chapter 2, we discuss a general approach that reframes this as a multi-task learning problem and uses recurrent deep neural networks (RNNs) to approximately evaluate the posterior quantiles. Because RNNs carry information along a sequence, this application is particularly useful in time series. An advantage of the risk-minimization approach is that we do not need to sample from the posterior or calculate the likelihood. We illustrate the proposed approach using several examples.

In randomized clinical trials, subjects often have expected benefits of treatment which depend on their baseline covariate values. In Bayesian adaptive enrichment trial designs, we can alter enrollment criteria at intermediate points in the trial, leveraging the fact that our understanding of how covariates affect outcomes improves with accumulating data. However, existing methods to design an adaptive rule, or trial policy, under which to enroll subjects are generally based on simple heuristics that may not optimally incorporate accumulating information. As in Chapter 2, we can reframe this rule design as an optimization problem, where the function to optimize is the trial policy, and we can use feedforward neural networks as a rich function class. In Chapter 3, we discuss a general approach that develops a trial policy for a Bayesian adaptive enrichment clinical trial using reinforcement learning with deep neural networks. Reinforcement learning allows us to construct a trial policy with

enrollment criteria that are simple functions of baseline covariates, but which are derived from potentially complex functions of existing trial information. We illustrate the proposed method in a simulation study that mimics the simple, but common scenario of evaluating a targeted treatment with a continuous-valued predictive marker.

Somatic hypermutation (SHM) is the enzyme-driven process by which antibodies acquire mutations as part of the adaptive immune response; these mutations enable them to improve binding to a diversity of antigens. Decades of research and tens of thousands of papers have revealed the biochemical basis of SHM, which can be expressed as a consensus model. This model can be expressed as a latent variable probabilistic model encoding a sequence of interacting steps. Even though large volumes of sequence data are available to fit the continuous parameters governing the forward evolution of this system, such parameters are still unknown. Indeed, fitting such a latent variable model is difficult. Similar to the approach used in Chapter 2, we can use a neural network to estimate the posterior mean in a way that does not require likelihood calculation. To do this, we reframe posterior mean estimation as an optimization problem, and train a neural network to accurately estimate the posterior mean. However, due to the high dimension of sequence data, some hand-tuned dimension reduction is necessary. In Chapter 4, we develop an approximate Bayesian computation strategy using neural networks to fit a complex latent variable model of SHM. We are able to estimate most of the parameters of the model to good accuracy, but find that the parameters involving the boundaries of the nucleotide stripping process are difficult to estimate given the type of data available.

Chapter 2

MARGINAL BAYESIAN POSTERIOR INFERENCE USING RECURRENT NEURAL NETWORKS WITH APPLICATION TO SEQUENTIAL MODELS

2.1 Introduction

We consider a common Bayesian scenario in which we have a data set, X, generated from some parametrically specified distribution $(X \,|\theta, \eta) \sim P_{\theta,\eta}$. Here, we assume that $\theta \in \mathbb{R}$ is a one-dimensional parameter of interest. For simplicity, we assume that the nuisance parameter $\eta \in \mathbb{R}^p$ is finite-dimensional, although this is not essential; in our discussion, we explain how to handle an infinite-dimensional η. We further assume that (θ, η) follows a known prior distribution g, that is, $(\theta, \eta) \sim g$. In analyzing our data set X, we would like to evaluate the posterior distribution for θ:

$$dP\left(\theta \,|\, X\right) = \frac{\int_\eta dP_{\theta,\eta}(X) g(\theta,\eta) d\eta}{\int_{\theta,\eta} dP_{\theta,\eta}(X) g(\theta,\eta) d\theta d\eta} \,.$$

The integral in the denominator is often analytically intractable. When the likelihood function, $dP_{\theta,\eta}(X)$, can be calculated efficiently (up to a normalizing constant), one can draw samples from the posterior $dP\left(\theta \,|\, X\right)$ using stochastic simulation techniques, such as Markov chain Monte Carlo (MCMC) and rejection sampling [5]. From these samples, any posterior summary can be computed. However, there are a number of situations in which the likelihood is difficult to calculate, but it is relatively straightforward to sample (η, θ) from g and X from $P_{\theta,\eta}$. Approximate Bayesian computation (ABC) methods have been developed for inference in this regime [e.g., 46]. These methods rely on simulating data from a joint prior-likelihood distribution, and computing the empirical distribution of parameters drawn from data that are "close" to the observed data in some sense. However, because ABC and more classical stochastic simulation techniques often use some form of rejection sampling,

they may have trouble with problems that are complex or high dimensional. Furthermore, defining "close" as it pertains to the sampled data X is nontrivial in many cases. For even a moderately large X, informative summary statistics must be constructed to reduce the rejection rate, otherwise the problem becomes intractable. If the summary statistics are not suitable for the particular setting, the approximation of the posterior will be poor [63].

In this work, we frame the calculation of posterior quantiles as an optimization problem. We consider the posterior quantile functions as risk minimizers with respect to particular losses. We approximate the function that minimizes this risk by restricting our optimization to the class of functions that can be represented by recurrent neural networks (RNNs) [68]. Because we expect the posterior quantile to "update" as more data become available, the recurrent structure is advantageous. Moreover, the recurrent framework allows us to evaluate posterior quantiles for data sets of arbitrary length. Provided that our network architecture is adequate for the specific problem, our function class is sufficiently rich, and the risk minimizer over this class is close to the true posterior quantile function. We discuss how stochastic subgradient optimization can be used to find local optima over this class. In particular, by simulating parameters/data from our prior/likelihood, our neural network can update its parameter estimates to "learn" how to evaluate posterior quantiles. We refer to this as a "statistical meta-learning procedure," because it uses simulated data to learn a Bayesian-optimal statistical rule.

Another obvious advantage of RNNs is their natural compatibility with time series data, where the contribution of each observation to a parameter of interest can depend on previous or later observations. RNNs share information across recurrent steps, making them naturally suited to handling this type of problem. Furthermore, many Bayesian time series are difficult, owing to the intractability of the likelihood (see [24] for some examples). Avoiding an approximation of the entire posterior distribution is favorable in these scenarios when our target is credible intervals, for example.

Other works have proposed using deep learning in similar contexts. The idea of using an RNN as a meta-learner was first introduced by [29]. [93] use deep neural networks to

approximate posterior summaries. In particular, they approximate the posterior mean of a functional in high-dimensional problems. [14] builds on this work by applying this approach to particular econometric models, using feedforward neural networks to estimate the posterior means. [10] use exchangeable neural networks to model population genetic data. They train on the fly, simulating new data each time they update the weights of their neural network. However, they approximate the entire posterior using a parametric approach.

2.2 Reframing as an Optimization Problem

For any $t \in (0,1)$, let $Q^t(X)$ denote the tth quantile of the posterior of θ under a prior g and a likelihood $P_{\theta,\eta}$. Our goal is to learn the quantile functions $\{\Phi(t) := Q^t(\cdot) \text{ for all } t \in \mathcal{T}\}$, where $\mathcal{T} \subseteq (0,1)$ either contains finitely many elements or is equal to $(0,1)$.

The quantity $Q^t(X)$ is the solution in q to

$$P\left(\theta \leq q \mid X\right) = t \ ,$$

where we assume for simplicity that this posterior is continuous.

Our approach hinges on the fact that Q^t can be written as the solution to an optimization problem. In particular,

$$Q^t(X) = \text{argmin}_Q \, \mathrm{E}\left[\rho_t\left(\theta - Q(X)\right)\right], \tag{2.1}$$

where $\rho_t(u) = u\left(t - I\left\{u < 0\right\}\right)$ is the asymmetric L_1 norm (or "pinball loss," as referred to by 40), and the minimization is taken over all functions that are measurable with respect to $\sigma(X)$ (the sigma-algebra generated by the data). Note that when $t = 0.5$, this loss is precisely a scaled L_1 loss.

Solving (2.1) directly is intractable, in general, over the space of all measurable functions. Instead, we restrict the problem to a rich finite-dimensional subclass, namely, those functions that can be represented by deep RNNs, which is known to be a very rich class [16, 6]. In

this case, our optimization problem consists of finding

$$\begin{aligned}\beta^* &= \text{argmin}_\beta \, \mathrm{E}\left[\rho_t\left(\theta - Q_\beta(X)\right)\right] \\ &= \text{argmin}_\beta \int_{\theta,\eta,\mathcal{X}} \rho_t\left(\theta - Q_\beta(X)\right) dP_{\theta,\eta}(X) g(\theta,\eta) d\theta d\eta \ ,\end{aligned} \tag{2.2}$$

where Q_β is a deep RNN that takes as input each of the n observations in X and outputs a single value, and β is a vector of the weight and bias parameters in the neural network. For this network to be an effective estimator, we may need a large number of parameters: the network needs to learn both how to combine inputs to get the output, and what values to store in memory across recurrent steps.

2.2.1 Relation to Quantile Regression

The asymmetric L_1 norm described above is most commonly used in quantile regression. However, in general, a quantile regression is performed in a frequentist context, where the goal is to estimate a quantile or the distribution of a conditional outcome given a set of features. Hence, the outcomes and covariates are not fixed. In fact, the only similarity between our proposed method and a quantile regression is the use of a pinball loss. We are attempting to learn one or many posterior quantiles as functions of data simulated from the prior and the likelihood, $Q^t : \mathcal{X} \to \mathbb{R}$, as a minimizer of (2.1) over the joint distribution of (X, θ), $\int_\eta dP_{\theta,\eta}(X) g(\theta,\eta) d\eta$. We do this without ever engaging with a single, fixed, observed data set, but rather by engaging with many simulated data sets. Reframing the derivation of the posterior quantile in this way allows us to leverage deep learning for the posterior quantile approximation.

2.2.2 Motivation for Recurrent Structure

Time Series and Sequential Models

RNNs are advantageous for posterior inference in sequential models because they pass information sequentially, by design. For illustrative purposes, consider the following second-order

moving-average model described in Section 2.3.5:

$$X_j \sim Z_j + \theta_1 Z_{j-1} + \theta_2 Z_{j-1}.$$

The observed sequence X depends on the parameters of interest, $\{\theta_1, \theta_2\}$, only through an unobserved, latent sequence Z.

For most choices of likelihood over Z and choices of prior over θ, the likelihood $P(X \mid \theta)$ is not tractable. Furthermore, the contribution of X_j to the posterior distribution depends on the values of X_{j-1} and X_{j-2}. Because we cannot sample from the posterior directly for nontrivial choices of the likelihood and prior, and because the data are sequential, an RNN is a good choice for a posterior estimation. Later, in Section 2.3.5, we show that our proposed method outperforms several others in terms of credible interval estimation.

Canonical Exponential Family

To motivate the choice of an RNN over a simpler multilayer perceptron, we consider a d-parameter canonical exponential family indexed by θ with a conjugate prior on θ, namely,

$$p(x|\theta) = h(x) \exp(\theta^\top T(x) - g(\theta))$$
$$p_{\eta_0,\gamma_0}(\theta) = k(\eta_0, \gamma_0) \exp(\eta_0(\gamma_0^\top \theta - g(\theta))),$$

for $\theta \in \mathbb{R}^d$. It follows that, if $X = \{x_1, \cdots, x_n\}$, where $X_1, \cdots, X_n \overset{\text{iid}}{\sim} p(\cdot|\theta)$,

$$p(\theta|X) = p_{\eta', \gamma(X)}(\theta),$$

with $\eta' = n + \eta_0$ and $\gamma(X) = \frac{\sum_{i=1}^n T(x_i)}{n+\eta_0} + \frac{\eta_0}{n+\eta_0}\gamma_0$. By the factorization theorem, in this simple setting, there exists a $(d+1)$-parameter sufficient statistic for $\theta|X$, namely, $[\sum_{i=1}^n T(X_i), n]$. Therefore, we can write Q^t recurrently as

$$Q^t(X) = h\left[x_n, f(x_{n-1}, f(\ldots f(x_1, \beta)\ldots))\right],$$

for some fixed $\beta \in \mathbb{R}^{d+1}$, $h : \mathbb{R}^{d+2} \to \mathbb{R}$, and some $f : \mathbb{R}^{d+2} \to \mathbb{R}^{d+1}$. Here $f(x_i, \sum_{j=1}^{i-1} T(x_j), i-1) = \{\sum_{j=1}^{i} T(x_j), i\}$. We can think of f as a "memory" function that tracks the sufficient

statistic across recurrent steps. In contrast, $h(\sum_{i=1}^{n} T(X_i), n) = F^{-1}(t)$, where F^{-1} is the quantile function of $p_{\eta',\gamma'}$. Thus, h calculates $Q^t(X)$ from the sufficient statistic. Note that increasing d is equivalent to increasing the size of the memory across the recurrent steps.

While it is not true that we can write $Q^t(X)$ in this way for a general likelihood and prior, we hope that for some sufficiently large memory size, there exist an h and f such that

$$Q^t(X) \approx h\left[x_n, f(x_{n-1}, f(\ldots f(x_1, \beta)\ldots))\right].$$

In fact, note that if the size of the memory exceeds the number of observations, the set of order statistics is sufficient.

An advantage of the recurrent structure is that it allows us to approximate a posterior quantile for a data set of arbitrary length. When the data are independent and identically distributed, this is particularly useful, because we are learning how to "update" the posterior as the number of observations grows. Using a feedforward architecture, one can only obtain posterior quantile approximations for data sets of a fixed size that are compatible with the input layer of the network.

This proposed RNN can take in a data set and return an approximation to a single posterior quantile. It is also possible to design a network that learns more than a single quantile, and potentially all quantiles. These extensions are described in what follows.

2.2.3 Discretized Multi-task Learning

We first extend our procedure to simultaneously approximate $\{Q^t\}$ for a discrete collection of t with a single network. Let $\mathcal{T}$ denote that discrete set of quantiles, and write $m_{\mathcal{T}} = |\mathcal{T}|$. We use a multi-task learning network architecture [67] with shared hidden nodes and internal weights, and a set of $m_{\mathcal{T}}$ task-specific output nodes leading from the shared, final hidden layer to approximate the $m_{\mathcal{T}}$ quantiles. Let β_{in} denote the shared internal weights, and let $\beta_{out}(t)$ denote the task-specific output weights used to estimate Q^t. In this case, to find the

optimal weights β^*_{in} and $\beta^*_{out}(t)$, $t \in \mathcal{T}$, we must find

$$\text{argmin}_{\beta_{in},\beta_{out}(t):t\in\mathcal{T}} \sum_{t\in\mathcal{T}} w_t \, \text{E}\left[\rho_t\left(\theta - Q_{\beta_{in},\beta_{out}(t)}(X)\right)\right], \tag{2.3}$$

with $w_t > 0$ denoting predefined costs. For example, one might take $w_t = 1$, for all $t \in \mathcal{T}$. This *hard sharing* multi-task learning framework that shares internal nodes across tasks seems sensible, because intermediate representations of the data that are useful for posterior calculations are likely not quantile-specific. However, one might consider modifying this to allow a "final stage," with several hidden layers specific to each quantile.

2.2.4 Continuous Quantile Learning

We further extend this to construct a single network that approximates all quantiles of the posterior simultaneously. To begin, we consider the optimization problem

$$\mathcal{Q} = \text{argmin}_Q \int_0^1 w(t) \, \text{E}\left[\rho_t\left(\theta - Q(t, X)\right)\right] dt \ , \tag{2.4}$$

where w is any positive integrable function on $(0, 1)$. The solution to (2.4), $\mathcal{Q}$, is a function that takes t and X as inputs, and returns the tth posterior quantile, that is, $\mathcal{Q}(t, X) = Q^t(X)$, following directly from (2.1). By approximating the problem in (2.4) using neural networks, we can build a single network to simultaneously approximate all quantiles.

Without loss of generality, we assume that $\int_0^1 w(t)dt = 1$. Then, for random T drawn from density w, we can rewrite (2.4) as

$$\mathcal{Q} = \text{argmin}_Q \, \text{E}\left[\rho_T\left(\theta - Q(T, X)\right)\right], \tag{2.5}$$

where the expectation is now over T, θ, and X.

As noted above, we approximate the solution to (2.4) using a deep neural network by solving the optimization problem

$$\beta^* = \text{argmin}_\beta \, \text{E}\left[\rho_T\left(\theta - Q_\beta(T, X)\right)\right], \tag{2.6}$$

where Q_β is a network that takes as input a data set and the selected quantile, and returns a single output. The architecture of this network is similar to the single quantile problem

described in Section 2.2, except that T is included as an additional input at every recurrent step.

2.2.5 Optimization

Let ℓ_1, ℓ_M, and ℓ_C denote the losses in criteria (2.2), (2.3), and (2.6), respectively. We can optimize all three using standard stochastic subgradient-based methods [52]. In particular, for the loss (2.2), the sub-differential of our loss, $\nabla \ell_1(\beta)$, consists of

$$\nabla \ell_1(\beta) \ni -\mathrm{E}\left[\dot{\rho}_t\left(\theta - Q_\beta(X)\right) \nabla Q_\beta(X)\right],$$

where $\nabla Q_\beta(X)$ is a subgradient of our network, and $\dot{\rho}$, defined pointwise as $\dot{\rho}_t(u) = tI(u > 0) + (1-t)I(u < 0)$, is a subgradient of the quantile loss function. There are many possible subgradients, depending on how one defines $\dot{\rho}_t(0)$. Stochastic subgradients can be calculated by sampling a (θ, η) pair from g and a data set X from $P_{\theta,\eta}$, and then plugging these into

$$\hat{\nabla} \ell_1(\beta) = \dot{\rho}_t\left(\theta - Q_\beta(X)\right) \nabla Q_\beta(X)\ . \tag{2.7}$$

By linearity, stochastic subgradients can be calculated similarly for the multi-task learning criterion, ℓ_M.

For continuous quantile learning, our expectation-based formulation, ℓ_C, allows us to easily calculate stochastic subgradients. In particular, given a sampled (θ, η) pair, a data set X sampled from $P_{\theta,\eta}$, and T sampled from w, we can calculate

$$\hat{\nabla} \ell_C(\beta) = \dot{\rho}_T\left(\theta - Q_\beta(X)\right) \nabla Q_\beta(X), \tag{2.8}$$

and note that $\mathrm{E}[\hat{\nabla} \ell_C(\beta)]$ is in the sub-differential of ℓ_C at β.

The full approximation procedure for 2.2 is described in Algorithm 1. We repeatedly sample (θ, X) pairs, and then use these pairs to train our neural network. Note that, in general, steps 3 and 4 use back propagation. We can extend this algorithm using linearity for the multi-task learning criterion ℓ_M, and having the network output one prediction for each quantile of interest. For the optimization described by 2.6, we need only sample T from w

before we calculate the gradient in step 3. Although we describe a simple stochastic gradient descent, extensions to more complex stochastic optimization techniques that use adaptive learning rates/momentum [e.e., 38, 77] are usually employed in simulations in practice.

Algorithm 1 Learn $Q^t(\cdot)$

Require: m = batch size, k = learning rate

for $0 \leq i \leq$ total iterations **do**

1. Simulate $(\theta_1, \eta_1), \ldots (\theta_m, \eta_m) \overset{\text{iid}}{\sim} g(\theta, \eta)$
2. Simulate $X_1 \sim P_{\theta_1,\eta_1}, \ldots, X_m \sim P_{\theta_m,\eta_m}$
3. $\hat{\nabla}\ell(\beta) \leftarrow \frac{1}{m}\sum_{j=1}^{m} \dot{\rho}_t\left(\theta_j - Q_\beta(X_j)\right) \nabla Q_\beta(X_j)$
4. $\beta \leftarrow \beta - k\ \hat{\nabla}\ell(\beta)$
5. $i \leftarrow i + 1$

end for

2.3 Examples

In this section, we consider three examples. In the first, we simply estimate the median of i.i.d. Gaussian observations using a Gaussian prior, perform a multi-task estimation of the posterior deciles, and learn the posterior quantiles continuously. Here, the posterior quantiles have a simple closed form, so evaluating performance is straightforward. For the second example, we approximate the posterior median for the maximum component in a mixture of finite mixtures (MFM). Finally, we approximate the posterior median for the basic reproduction number in a Bayesian stochastic SIR model.

For all examples, the optimization is performed using `Adam` [38], as implemented in `TensorFlow` [1], with subgradients calculated using mini-batches. We evaluate the performance in all simulation settings based on the mean loss, or estimated risk, over a held-out test set. The error bars at the conclusion of training represent a 95% confidence interval of the risk of the neural network, based on the sample standard error of a held-out test set. Where appropriate, we compare our neural networks with Stan, another general-purpose

posterior estimation tool, in addition to comparing them with a specialized, problem-specific tool.

2.3.1 Gaussian Example

We consider a simple example with i.i.d. observations $x_1, \ldots, x_n$ drawn from a $N(\theta, 1)$ distribution, with θ drawn from a $N(0, \sigma^2)$ prior. Here, we would like posterior quantiles for θ given our data. In this case, it is well known that

$$\theta \,|\, x_1, \ldots, x_n \;\sim\; N\left(\frac{\bar{x}\sigma^2}{1/n + \sigma^2}, \left(n + \sigma^{-2}\right)^{-1}\right),$$

where $\bar{x} = n^{-1} \sum_i x_i$. Thus,

$$Q^t(X) = \frac{\bar{x}\sigma^2}{1/n + \sigma^2} + \frac{\Phi^{-1}(t)}{\sqrt{n + \sigma^{-2}}} \; .$$

Furthermore, letting $X_j = \{x_1, \ldots, x_j\}$, we have, for $j \geq 2$,

$$Q^t(X_j) = \frac{(\frac{j-1}{j}\bar{x}_{j-1} + x_j)\sigma^2}{1/j + \sigma^2} + \frac{\Phi^{-1}(t)}{\sqrt{j + \sigma^{-2}}} \; .$$

Thus, in this simple case, there exists an exact recurrent update step that uses a two-dimensional sufficient statistic, $\{\bar{x}_j, j\}$. Although this case is quite basic, it is illustrative, and because we have access to the exact value of $Q^t(X)$, it is trivial to evaluate the performance of our approximated posterior quantile.

We consider approximating the posterior median. In this specific case, we can assess the quality of our network by comparing the pinball loss of our network to that of the true posterior median of a held-out test set.

We ran simulations using $n = 100$ observations, where θ has prior variance $\sigma^2 = 1/100$ (equal to the conditional variance of $\bar{x}$). We use the mini-batch `Adam` for the optimization, where each mini-batch contains 100 data sets, with a learning rate of 10^{-2}. Our network has 32 nodes per hidden layer, and four hidden layers with ReLU activation functions. We evaluate the risk on a held-out test set of 500 data sets. The performance is summarized in Figure 2.1.

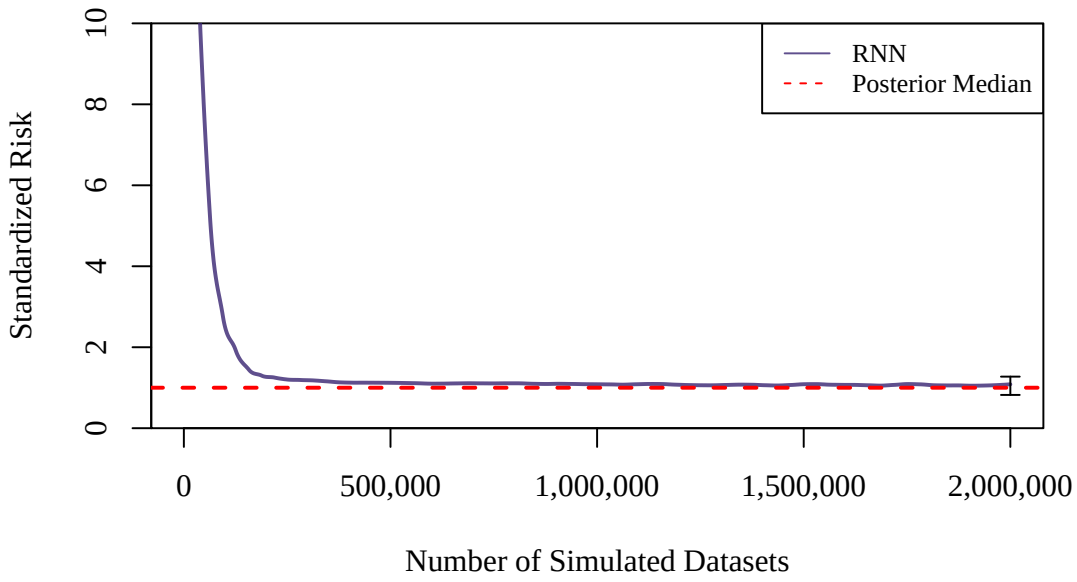

Figure 2.1: Risk curve for the estimation of the posterior median in a Gaussian prior and Gaussian likelihood simulation scenario. The risk of the true minimizer, the posterior median, is standardized to be equal to one (red). Here, the posterior median has a closed form. Our estimator has about 2% excess risk when compared with the Bayes estimator.

We also ran simulations using an identical prior and likelihood, but this time attempting to simultaneously learn all nine posterior deciles, with a uniform weight on the loss over the nine tasks, as described in 2.3. The performance is summarized in Figure 2.2. We compare the results with the risk of the true posterior deciles, and obtain a risk that is close to the true risk of the posterior deciles. We evaluate the performance on a test set of 500 held-out data sets.

Finally, we ran simulations using the same prior and likelihood as above, but where the target of the estimation was a random quantile sampled uniformly on the unit interval, using the loss described in 2.6. We included the target quantile as an input to our neural network at each recurrent step. We evaluated the performance on a test set of 500 held-out data sets and 500 held-out random quantiles. In Figure 2.3, we compare the average loss over this test set with that of the corresponding posterior quantiles, which have a closed form.

2.3.2 MFM Example

Here, we consider a slightly more complex case in which i.i.d. observations $x_1, \ldots, x_n$ are drawn from a $\frac{1}{k}\sum_{i=1}^{k} N(\theta_i, 0.01)$ distribution, with θ drawn from a $N(0, 0.25)$ prior, and k drawn from a Pois(4) prior, shifted to have a minimum of one. The choice of variance for the prior and the likelihood ensure some separation between the components.

Our estimation target is the posterior median of $\theta_{(k)}$, the maximum component. Note that this parameter is sensitive to the number of components, with the distribution of $\theta_{(k)}|X, k$ depending heavily on k. This makes procedures that try to determine the number of clusters, such as the BIC, inaccurate.

We ran simulations using $n = 250$ observations. We used mini-batch `Adam` for optimization, where each mini-batch contains 150 data sets, with a learning rate of 10^{-4}. We used a network with 32 nodes per hidden layer, and four hidden layers.

The reason for the larger mini-batch in this simulation setting is the increased variability in the gradient due to the variable number of components. A larger mini-batch size makes the stochastic optimization more stable. The performance is summarized in Figure 2.4.

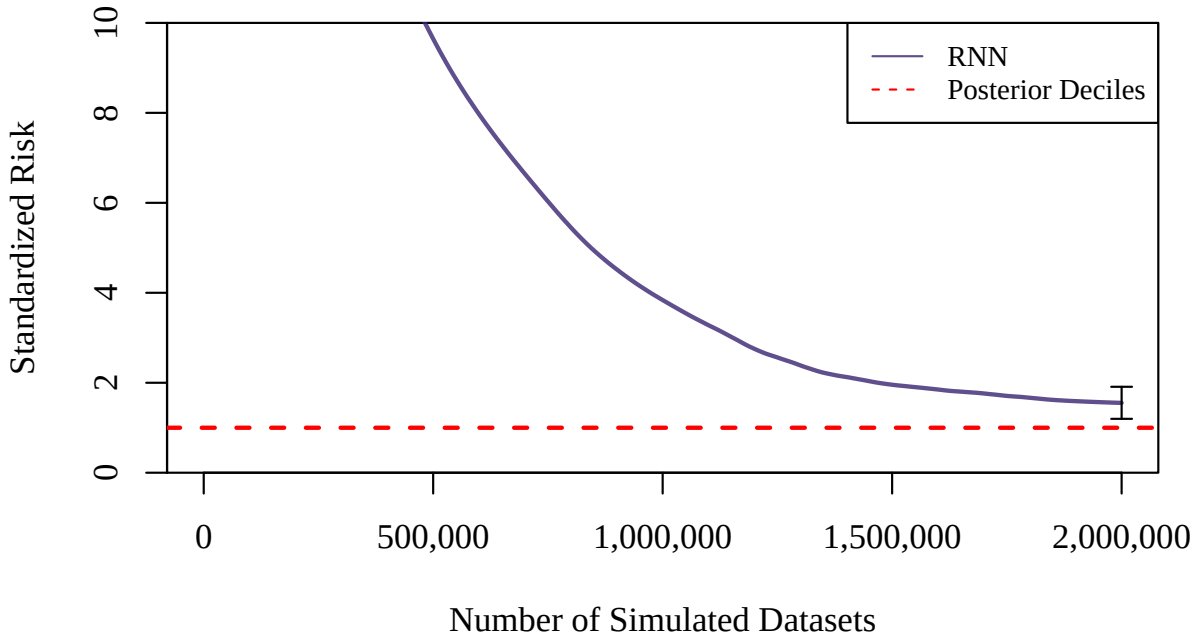

Figure 2.2: Risk curve for the estimation of the posterior deciles in a Gaussian prior and Gaussian likelihood simulation scenario. The risk of the true minimizer, the posterior deciles, is set to one (red). Here, the posterior deciles have a closed form. Our estimator has about 30% excess risk when compared with the Bayes estimator.

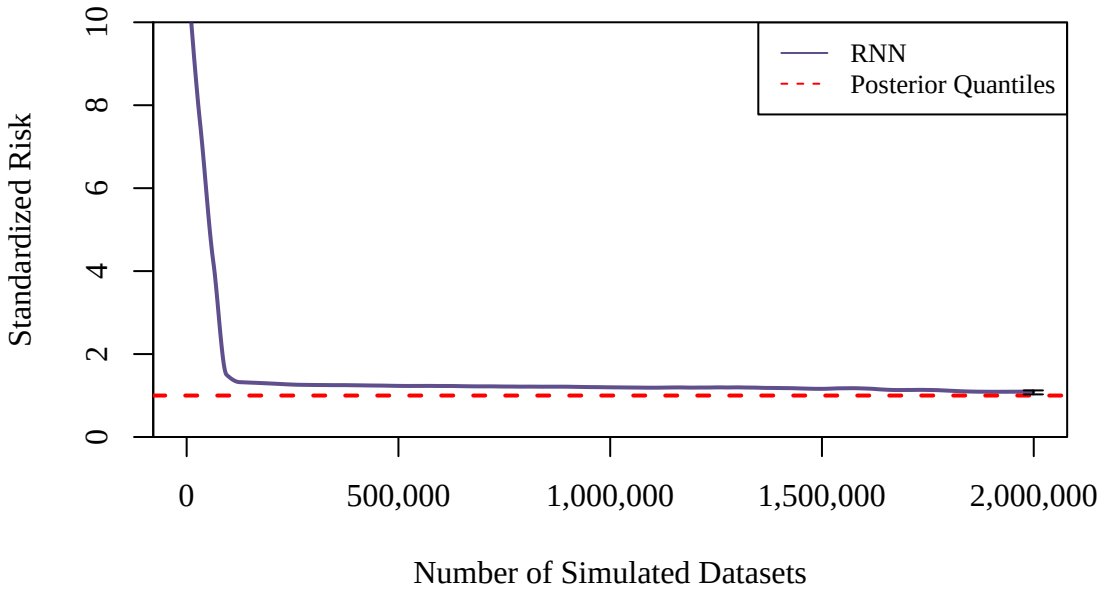

Figure 2.3: Risk curve for the estimation of the posterior quantiles in a Gaussian prior and Gaussian likelihood simulation scenario. The quantile estimated for each of the 1500 simulated data sets is sampled uniformly at random from the unit interval. The risk of the true minimizer, the posterior quantiles, is set to one (red). Here, the posterior quantiles have a closed form. Our estimator has about 5.6% excess risk when compared with the Bayes estimator.

We compare the results with those of a Julia package specialized for obtaining approximate posteriors for this model class [50], and with those of Stan [9], using two strategies to estimate the number of components. The first strategy uses the BIC to estimate the number of clusters, and the second uses the prior mean number of clusters. Stan proceeds with the number of clusters fixed. Because our parameter of interest is sensitive to the number of clusters, the Stan posterior estimation framework performs poorly. Our neural network approaches the risk of the approximate posterior median given by a specialized package within about $15,000$ simulated data sets. We evaluated the performance on a held-out test set of 500 datasets.

2.3.3 Stochastic SIR Example

Here, we consider a stochastic SIR model. In this setting, we observe a disease epidemic, and would like to estimate the posterior median for the basic reproduction number, R_0. A stochastic SIR model is a continuous time stochastic process that models how a disease interacts with a population of size N. At each time t, there are a number of susceptible individuals, $S(t)$, a number of infected individuals, $I(t)$, and a number of recovered individuals, $R(t)$. The disease is modeled according to the following differential equations:

$$\begin{aligned}\frac{\partial S(t)}{\partial t} &= -\frac{\beta S(t)I(t)}{N} + \sqrt{\frac{\beta S(t)I(t)}{N}}\omega_1(\partial t) \\ \frac{\partial I(t)}{\partial t} &= \frac{\beta S(t)I(t)}{N} - \gamma I(t) - \\ &\qquad \sqrt{\frac{\beta S(t)I(t)}{N}}\omega_1(\partial t) + \sqrt{\gamma I(t)}\omega_2(\partial t),\end{aligned}$$

where β is the infection rate, γ is the recovery rate, and ω_1 and ω_2 are standard Wiener processes [2]. The basic reproduction number, R_0, is the number of expected new infections an individual generates over the course of their disease in a fully susceptible population, β/γ. This parameter, and its posterior quantiles, are of interest in this problem [12], because $R_0 > 1$ means the disease is likely to become an epidemic.

We simulate from $t = 0$ to $t = 100$ using finite differences with $\partial t = 0.01$. However, we

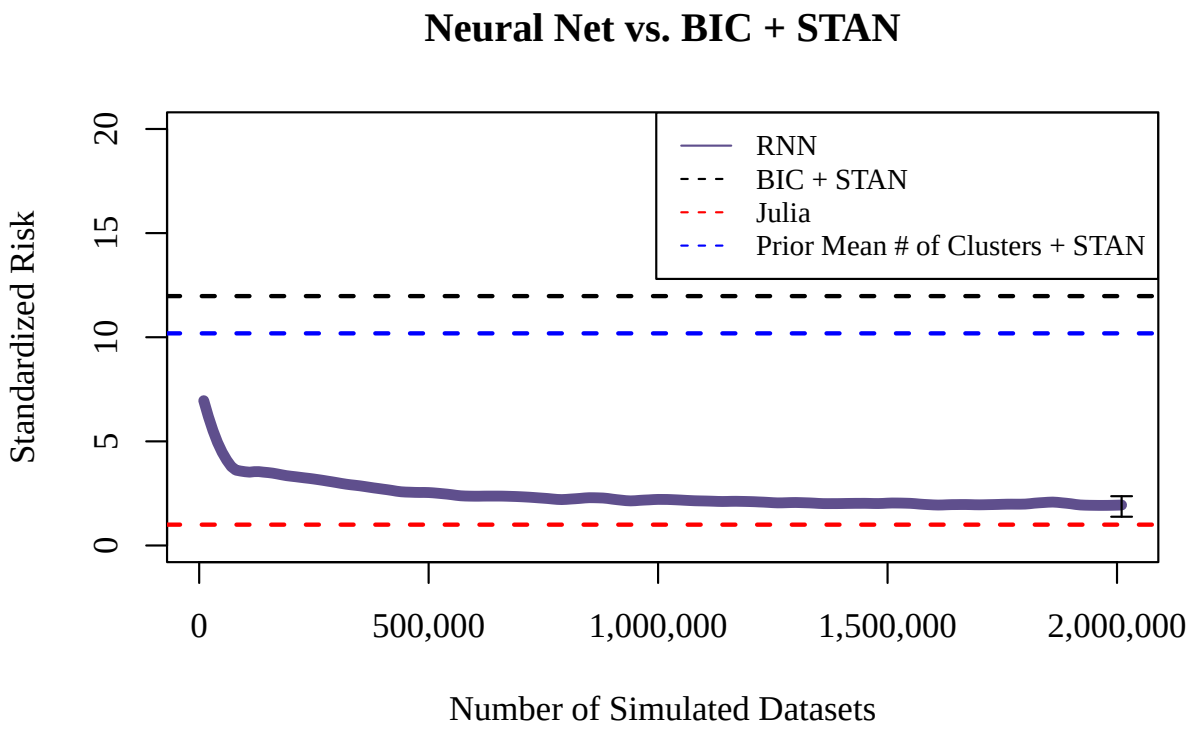

Figure 2.4: Risk curve for the estimation of the posterior median in a Gaussian prior and Gaussian mixture likelihood simulation scenario, with a prior on the number of mixture components. We compare the results with those of Stan, using two strategies to select the number of components, and with those of a specialized Julia package, BayesianMixtures. Our estimator has 87% excess risk when compared with BayesianMixtures, though it is much improved over our Stan implementations. A better architecture and more training may lead to better results.

observe only integer values of t, for a total of 101 observations. In this setting, we observe both $S(t)$ and $I(t)$, though it is simple to reduce the observed data to only $R(t)$, for example. We specify the following priors for β and γ:

$$\beta \sim \Gamma(9, 0.05)$$
$$\gamma \sim \Gamma(3, 0.05).$$

We selected these priors based on [12], tuning the values so that the sample paths express a variety of disease dynamics. We used mini-batch `Adam` for optimization, where each mini-batch contains 100 data sets, with a learning rate of 10^{-4}. We used a network with 32 nodes per hidden layer, and four hidden layers.

The performance is summarized in Figure 2.5. We compare the results with those of Stan, where we have attempted to solve this problem using finite differences. Our neural network approaches the risk of Stan within about $30,000$ simulated data sets, though this problem is slightly more difficult for us. We evaluated the performance on a held-out test set of 2000 data sets.

To determine whether our method improves over Stan in this setting, we consider the same data-generating mechanism, but instead of observing $S(t)$ and $I(t)$ for all integer values of t, we observe only $t \in \{0, 20, 40, 60, 80, 100\}$. In the case of our Stan implementation, we can only use finite differences over the six observations, because we do not marginalize out the drift and the stochastic component over a finer, unobserved grid. Because our observed grid is sparse, this leads to a very poor approximation of the true data-generating mechanism. However, for our RNN, we can simulate over a grid of arbitrary granularity, but only train on the six observed time points. The results are shown in Figure 2.6. We see that, owing to the misspecification of Stan, our method has a lower risk after a small number of observed data sets. This is an example of how stochastic simulation techniques can fail when calculating the likelihood exactly is computationally intractable.

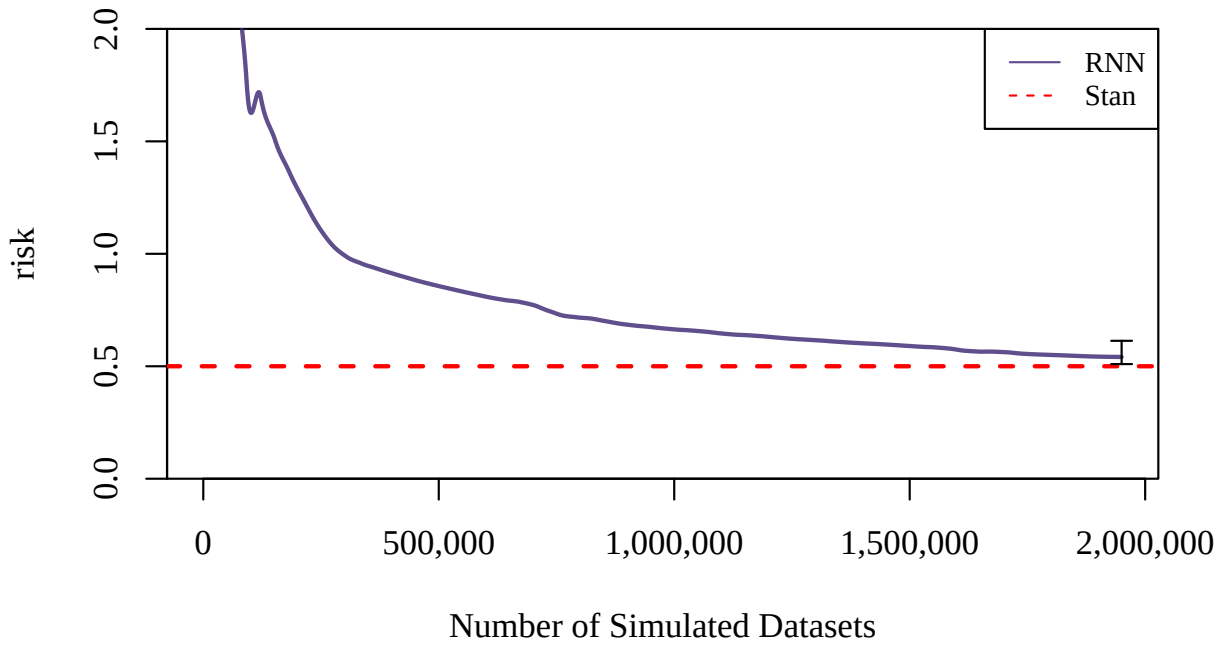

Figure 2.5: Risk curve for estimation of the posterior median of the basic reproduction rate in the stochastic SIR simulation setting. We compare the results with those of Stan, and both methods observe a grid of 101 time points. Our estimator achieves a risk that is about 3% greater than that of the Stan implementation.

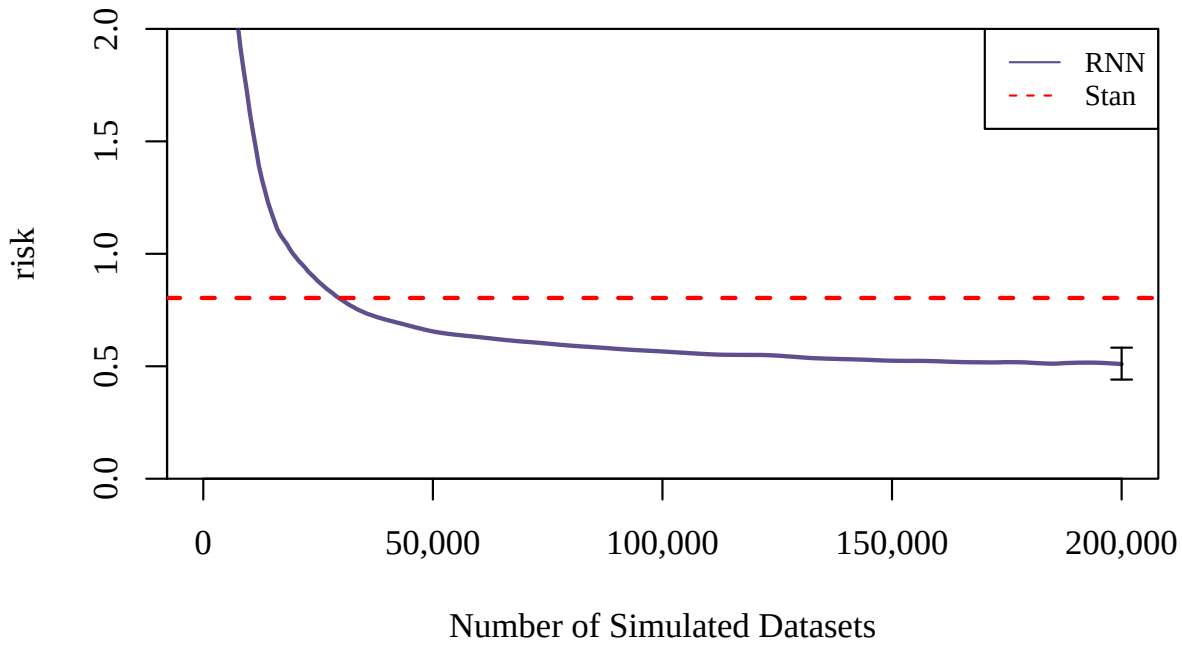

Figure 2.6: Risk curve for the estimation of the posterior median of the basic reproduction rate in the stochastic SIR simulation setting. We compare the results with those of Stan, and both methods observe only six time points. Our estimator had about a 44% reduction in risk over that of the Stan implementation.

2.3.4 HMM Example

In this section, we consider a hidden Markov model (HMM) example. In this setting, we observe a sequence of values, $y_1, \ldots, y_{100}$. These values are emissions from hidden states $x_1, \ldots, x_{100}$, which are not observed. There are three possible states at each x_i, $\{s_1, s_2, s_3\}$. We selected the following prior and likelihood for these simulations:

$$\begin{aligned} \theta &\sim N(0, 1) \\ Z_1, Z_2, Z_3 &\sim N(\theta, 1) \\ X_0 &\sim U(\{s_1, s_2, s_3\}) \\ y_i | X_i = s_j &\sim N(Z_j, 1) \\ P(X_{i+1} = s_j | X_i = s_k) &= \frac{e^{|z_j - z_k|}}{\sum_{j=1}^{3} e^{|z_j - z_k|}}, \end{aligned}$$

Therefore, the hidden states represent unobserved means from which observed Gaussian random variables are drawn. The transition probability between states is proportional to the exponentiated ℓ_1-distance between those states. Our goal is to conduct an inference on the posterior distribution of θ, the center of these unobserved means, $\theta | y_1, \ldots, y_{100}$. This model is a simplified version of the type of continuous-emission HMMs used in speech recognition [3].

We estimated 90% credible intervals on a held-out test set of 2000 sequences. We compared the results with those of a standard importance sampling ABC with Gaussian kernel weights. Because Y is too high-dimensional to be computationally tractable, we instead use the deciles of the Y sequence as a summary statistic. We generated 1000 importance samples per observed sequence, and estimated the 0.05 and 0.95 posterior quantiles from this sample with a bandwidth of one.

The results are shown in Figure 2.7. In this simulation scenario, it is difficult to choose sensible summary statistics. The recurrent network has a marginal coverage of 88.95% and

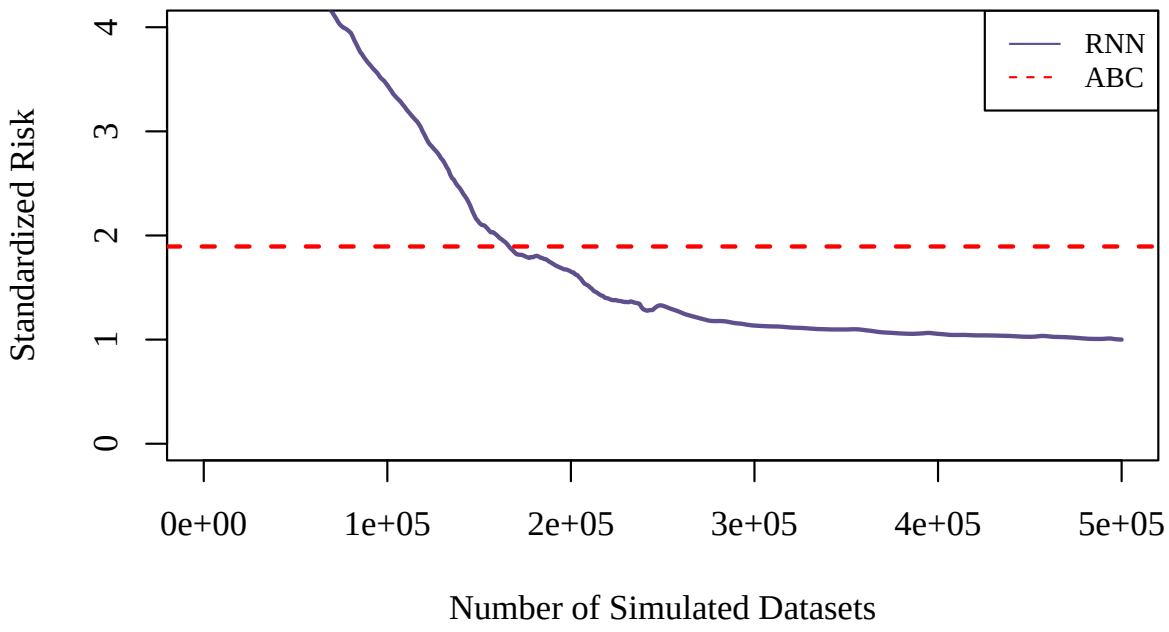

Figure 2.7: Standardized risk of the estimated 90% credible interval for the center of the unobserved hidden state means.

an average interval width of 2.11. The ABC with summary statistics has a marginal coverage of 50.35% and an average interval width of 0.92. Recall that the prior quantiles have an interval width of 3.29 and marginal coverage of 90%. Thus, ABC with these summary statistics is more conservative than the prior, whereas the RNN leverages information to provide narrower intervals. The RNN is ideal in this scenario because of the difficulty of the problem and the sequential nature of the observed data.

2.3.5 Order-Two Moving Average Example

Finally, we consider an order-two moving-average model. Suppose we observe a sequence X, with

$$X_j \sim Z_j + \theta_1 Z_{j-1} + \theta_2 Z_{j-1},$$

where Z_j are latent Gaussian random variables. We would like to obtain a 90% credible interval on θ_2. Note that if Z_j follow a non-Gaussian distribution, the likelihood $P(X \mid \theta)$ is intractable. However, here, we can draw comparisons with the exact posterior distribution.

The likelihood and prior are identical to those of [93]. The Z_i are standard Gaussian, and θ_1 and θ_2 are uniform over a specific triangular region such that they are identifiable.

We compare our method with the semi-automatic ABC procedure described in [20], transforming the input data into a vector of one-gap products, $X_j X_{j+2}$. We chose this transformation because the sample mean of $X_j X_{j+2}$ is a consistent estimator of θ_2. Using a polynomial basis expansion of this transformed X, we then use a linear regression to estimate the posterior mean, and use that estimate as a summary statistic. Fearnhead and Prangle call the method "semi-auto ABC," because although the procedure can be automatic, the choice of the transformation in this problem is not. We also compare our results with those of a method proposed by [93], who use a deep neural network to estimate the posterior mean. Then, they use that estimate as a summary statistic for ABC. In both settings, we use rejection sampling to obtain the approximate posterior. The rejection scheme and the number of samples drawn are calibrated to have approximately the same runtime as our RNN method. The method were tested on a held-out set of 1000 sequences.

The loss results can be seen in Figure 2.8. Because both comparison methods use an estimate of the posterior mean as a summary statistic, it is not surprising that the perform poorly in the tails. Our method directly targets the 90% credible interval, and has approximately correct coverage, while maintaining relatively narrow interval widths.

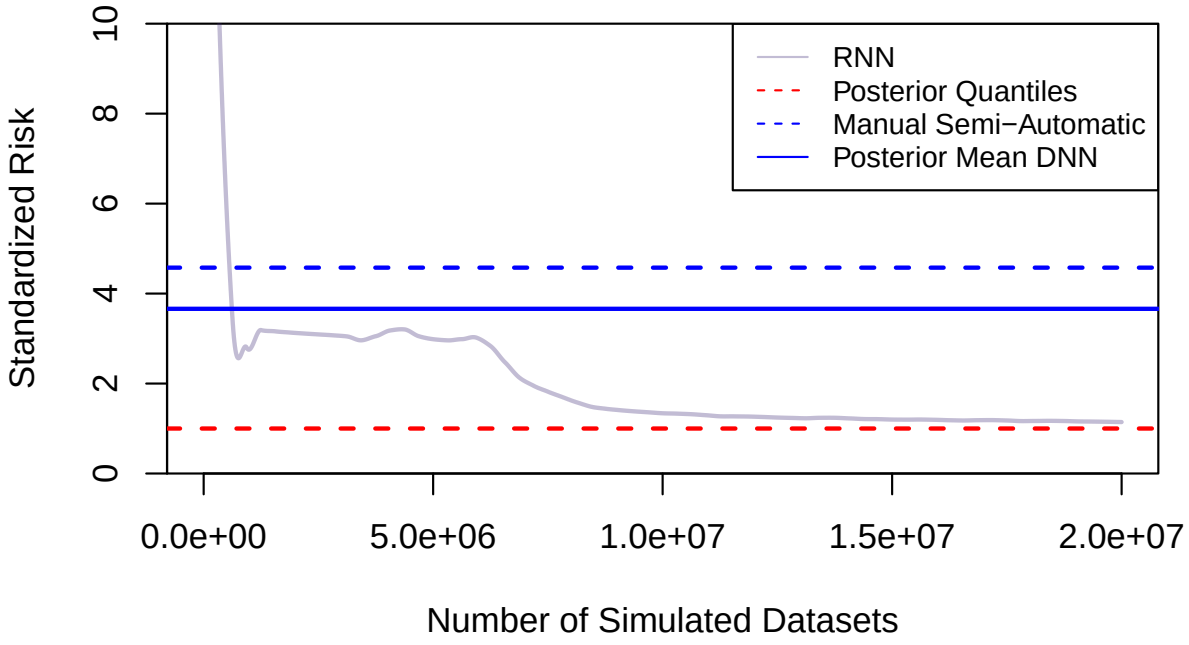

Figure 2.8: Standardized risk of the estimated 90% credible interval for the second-order coefficient in a moving average model.

2.3.6 Summary of Coverage Results

For all of the simulation settings above, we estimated 90% posterior credible intervals. A comparison of our coverage results with those of various other estimators can be seen in Table 2.1. Our RNN estimators outperform or have comparable performance to STAN in every simulation setting. We maintain approximately correct marginal coverage, while having much narrower intervals than those of the prior.

2.4 Discussion

We have proposed a method for using deep RNNs to approximate posterior quantiles of a univariate parameter of interest from a possibly multivariate Bayesian problem. To fit this neural net, it is only necessary to sample from the prior and likelihood. Posterior samples never need to be drawn, and, as long as we can sample from $P_{\theta,\eta}$, the likelihood itself never needs to be calculated. We have proposed three types of networks: a simple network for estimating a single, prespecificed, conditional quantile; a multi-task network for estimating a finite set of prespecified quantiles; and a slightly more complex network for estimating the entire conditional quantile function. We show that, in increasingly complex settings, an RNN can approximate single posterior quantiles approximately as accurately as specific state-of-the-art methods can that attempt to sample from the posterior directly.

To simplify the exposition, in Section 2.1, we assumed that the nuisance parameter, η, belongs to a finite-dimensional space. In fact, we do not use this assumption. We require only that one can sample parameters from the prior, and subsequently sample data from the corresponding likelihood. Therefore, in nonparametric Bayes problems, where an infinite-dimensional parameter can be sampled from a prior and the data can be sampled from the likelihood, the proposed method can be immediately used to estimate a univariate summary of this infinite-dimensional parameter.

Compared with other generalized tools for posterior approximation, such as Stan, our method is advantageous when the likelihood is not easily calculable. In the second stochastic

Gaussian			
Method	Coverage	Interval Width	Loss
RNN	0.906	0.241	0.0149
Exact Posterior	0.8958	0.233	0.0146
Prior Quantiles	0.903	0.323	0.0203
Mixture of Finite Mixtures			
Method	Coverage	Interval Width	Loss
RNN	0.942	0.339	0.019
Stan (BIC)	0.130	0.283	0.377
BayesianMixtures	0.921	0.147	0.009
Prior Quantiles	0.900	1.225	0.079
Stochastic SIR			
Method	Coverage	Interval Width	Loss
RNN	0.879	4.12	0.321
Stan	0.878	2.773	0.198
Prior Quantiles	0.900	10.17	0.935
Sparse Stochastic SIR			
Method	Coverage	Interval Width	Loss
RNN	0.9105	3.64	0.362
Stan	0.586	2.55	0.926
Prior Quantiles	0.900	10.17	0.935
Hidden Markov Model			
Method	Coverage	Interval Width	Loss
RNN	0.890	2.11	0.137
ABC	0.504	0.92	0.260
Prior Quantiles	0.900	3.29	0.203
Order Two Moving Average			
Method	Coverage	Interval Width	Loss
Exact Posterior	0.90	0.417	0.029
RNN	0.865	0.527	0.035
Semi-Auto ABC	0.440	0.476	0.135
Post-Mean DNN	0.466	0.400	0.108

Table 2.1: Coverage of 90% credible intervals using various posterior estimators in several simulation settings. Estimators exhibiting poor marginal coverage are highlighted in red.

SIR model, the results of which are shown in Figure 2.6, we achieve a lower loss than that of a natural implementation in Stan. We reach a lower loss because our Stan implementation must necessarily misspecify the likelihood in order to get posterior estimates.

It is interesting to compare the proposed method to the ABC methodology in terms of different forms of the smoothing methods used in classical nonparametric problems. Both the ABC methodology and our meta-learning procedure aim to approximate a quantile using many draws from the prior and likelihood. In the ABC methodology, the function mapping from the observed data to a quantile of the posterior distribution is approximated by a local smoother that uses only information from simulation replicates near the originally observed data. This is analogous to kernel smoothing methods in nonparametric regression settings, where a regression function estimate at a single predictor value x relies only on observations with predictors near x. For bounded kernels, a perturbation of distant observed predictor values has no impact on the regression fit at a distant point. On the other hand, our neural network approach uses all simulated data points to learn a rich neural network approximation to the function mapping from the observed data to a posterior quantile. This is analogous to series estimators from nonparametric statistics, where linear models are fitted using a collection of basis functions applied to the observed predictors. Here, a perturbation of distant observed predictor values can have a nontrivial impact on the regression fit at a given point.

Our proposed method is also quite similar to other Bayesian deep learning methodologies, such as the exchangeble network proposed in [10]. However, our networks exhibit good performance in both exchangeable and non-exchangeable cases. One critical advantage of our recurrent networks is that we are able to obtain predictions for sequences or data sets of arbitrary length. This is not possible with an exchangeble network. However, as noted in [86], the memory in a recurrent network is a bottleneck in a number of contexts. Therefore, as the dependence grows between data points that are far apart in our sequences, the size of the memory in our recurrent nodes must grow as well. However, the fixed memory size of the recurrent nodes could also be beneficial in an online learning setting. As new data are

acquired, our RNN estimator quickly calculates new posterior estimates, without the need for retraining. Further investigation of potential applications to online inference should be conducted.

As with other posterior sampling and approximation methodologies, we anticipate that our method will suffer from the curse of dimensionality. For MCMC methods, the curse of dimensionality typically occurs when the parameters are high dimensional. In contrast, we do not expect that having a high-dimensional nuisance parameter η will negatively impact the performance of our method, because we only estimate the posterior of a univariate parameter θ. However, our method is susceptible to the curse of dimensionality in the data structure, because it implicitly smooths across possible data realizations. One approach to improve the performance of our method in the presence of high-dimensional data would be to increase the richness of our network. Because we can simulate an unlimited number of prior-likelihood draws for our stochastic gradient steps, using a very rich network will not lead to overfitting, though the optimization scheme may require more draws to converge. Therefore, when the data are high dimensional, we expect that our method will benefit greatly from recent progress in neural network software that enables users to leverage massive computational power to quickly optimize deep networks.

Chapter 3

OPTIMIZING BAYESIAN ADAPTIVE ENRICHMENT CLINICAL TRIALS WITH DEEP REINFORCEMENT LEARNING

3.1 Introduction

In modern medicine, treatments are often specifically targeted to a sub-population of patients, or catered to the individual patient [13]. Measureable characteristics other than general disease status may affect the efficacy of a treatment of interest. For example, the specific mutations of an individual tumor are used to guide mutation-targeted cancer therapies [56]. Clinical trials that ignore patient heterogeneity can promote drugs which are ineffective or harmful to large portions of the population [71]. The development of predictive signatures, scores indicating the potential benefit a patient might get from a specific treatment, are therefore of interest. An example of one such predictive biomarker is PD1/PD-L1 gene expression in cancer immunotherapy [58]. Based on applications like these, there is interest in statistical methods which can guide the development of policies for personalized treatment [44]. While one could design a trial that enrolls patients irrespective of potential characteristics which guide treatment, doing so often requires a large number of subjects to achieve a desired power. Therefore, [48] and [28], among others, have proposed frameworks that allow the selection of sub-populations during the trial to maximize efficacy. However, parameters controlling enrollment criteria in previous work are derived from simple functions of accumulated trial information. Here we propose a framework that leverages deep neural networks to learn enrollment criteria parameters which are derived from potentially very complicated functions of current trial information through the simulation of many Bayesian clinical trials. Using simulations to optimize and guide clinical trial design and drug development is

extremely common [30].

Bayesian adaptive enrichment designs allow patient enrollment criteria to be changed mid-trial while still maintaining a prespecified type 1 error [76]. While a trial enrollment decision rule is not itself a predictive signature, it can help us develop one more efficiently. Specifically, patients who are preferentially enrolled by the decision rule should be likely to have a higher *expected benefit of treatment* [62]. Some work has been done illustrating the potential benefit of adaptive trial design, as well as how one might select a trial policy. [82] show how using posterior means during an early-phase cancer dose-selection trial can more accurately select the correct dose for each trial subgroup. [75] propose a method for enrolling patients which uses covariate cut-points to maximize power. [66] use simulated annealing to allocate enrollment in a way that minimizes average sample size while maintaining a particular level of power and controlling type 1 error. While such methods can substantially increase power over a static clinical trial, as the true relationship between the outcome and covariates, as well as correlation between baseline covariates, become more complicated, a richer function class for enrollment criteria is desirable. There has been little work on how to develop an enrollment decision rule, or *trial policy*, for a setting where ad-hoc searches over a small space of potential policies are inefficient. In this manuscript, we aim to engage *reinforcement learning* in this process.

Reinforcement learning is a framework that allows us to improve a decision-making model in a setting where direct feedback on individual actions is not available. In reinforcement learning, we seek to maximize some reward with respect to the actions we take throughout a process. By making actions which lead to large rewards more likely, or reinforcing those actions, and making actions that lead to small rewards less likely, we can effectively train a neural network to take actions to maximize reward. A review of reinforcement learning can be found in [4]. In the context of clinical trials, the reward to maximize is a quantitative measure of the success of the trial, typically some function involving the detection of a significant treatment effect. While machine learning techniques have not been widely used in clinical trial design, reinforcement learning has been extremely successful in optimizing

an actor in state-action space, particularly in complete information games, where a large number of simulated games can be easily obtained, e.g. board games like Chess and Go [74].

In this work, we frame Bayesian trial policy development as an optimization problem. This idea is similar to that used in the closed-form solutions obtained for trial policy development by [55]. However, our method can be used in situations where there is no analytic or simple numerical solution for the optimal adaptive enrichment rule, and in situations where enrollment rules derived from simple functions of accumulated trial information perform poorly. To achieve this, we train neural networks through reinforcement learning to identify decisions that will maximize the expectation of a prespecified trial utility, a reward related to the trial outcome. The aforementioned neural network gives us an approximate solution to our optimization problem of interest. If our neural network is rich enough for the optimal solution to be close to the neural network solution, we will have a rule which is very close to optimal in terms of maximizing utility. As we are in a Bayesian setting, we can simulate an arbitrarily large number of trials (our only constraint is computational), and use the results of these trials to train our network. Through this framework, our network is able to "learn" how to efficiently run an adaptive enrichment design clinical trial.

Before moving on, we want to address a critical concern: There is well-founded skepticism about identifying a very complicated indication rule for characterizing who will benefit from treatment. Black box enrollment criteria have no clinical relevance, and are difficult to implement in practice. One would generally rather use a cutpoint applied to a single feature, or a simple linear combination of features to identify who should get treatment in the future. Though the framework discussed in this chapter uses a potentially complex neural network, it actually supports and is predicated upon the development of a simple indication. The neural network is used to specify a *trial policy*: This trial policy indicates how to combine accumulated information up to a given point in the trial to output parameters that can be used in an *indication* (which we will also refer to as an *enrollment rule*). The indication/enrollment rule is a function, with a small number of parameters, that takes, as input, a candidate patient's features, and identifies if they are eligible for the trial. It will

generally be a simple function of the parameters and features. For example, an enrollment rule might dictate that we enroll only patients with $x > c$ for some baseline covariate x, and some cutpoint c. This is quite a simple indication. However, in our framework, the value of c will be determined by a neural network, using previously accumulated trial information. Complexity in this determination does not necessitate complexity in the resulting indication. We discuss this distinction further in Section 3.2.4.

In Section 3.2, we describe the clinical trial setting, and explain how we use deep reinforcement learning to optimize decisions in these trials. Then, in Section 3.3, we apply our method to three simple simulation scenarios, and compare to simpler methods for designing an adaptive clinical trial. Finally, in Section 3.4, we discuss potential improvements to our proposed method, as well as additional applications and extensions.

3.2 Problem Setting

3.2.1 Guiding Example: Binary Adaptive Enrichment Design

We begin with an illustrative example which showcases a simple potential use case for our method. Consider a trial where we would like to investigate the effect of a binary treatment T on a binary outcome Y. Suppose there is also a binary biomarker X which modifies the effect of treatment. A key quantity of interest is the expected benefit of treatment – on a linear scale,

$$\gamma_x = \mathrm{E}(Y|X = x, T = 1) - \mathrm{E}(Y|X = x, T = 0).$$

In a randomized trial setting, we can estimate this quantity empirically,

$$\widehat{\gamma}_x = \bar{y}_{x,1} - \bar{y}_{x,0}$$

where $\bar{y}_{x,1}$ and $\bar{y}_{x,0}$ are the empirical means under treatment and control, respectively, for patients with $X = x$.

Suppose we enroll 100 subjects in our trial. From their randomized treatment assignments, we can estimate γ_0 and γ_1 as described above. Now, we would like to enroll an

additional 100 subjects, but use their X-values to guide enrollment. Specifically, suppose we would like to design an adaptive enrollment criterion which will maximize expected power. In this setting, arguably the most sensible rule is $\mathbf{1}(\widehat{\gamma}_x > \widehat{\gamma}_{1-x})$; though to formally justify this we would need to be more precise on how information would be combined in both blocks of the trial. Intuitively, this rule makes sense as we are enrolling subjects who we suspect, based on evidence from the first 100 subjects, will benefit the most from treatment. However, in more complex scenarios, it becomes difficult to design a good rule. In the following section, we will formalize the results from this illustrative example, and then give a framework under which we can learn a rule from repeated simulation. We also note that we often have multiple goals in a trial, beyond just maximizing power. We likely want to run a trial of minimal duration and may want to include all patients whom we expect to have sufficient positive benefit, among several other potential goals. Our framework can incorporate these goals, though we defer discussion of that to Section 3.4.

3.2.2 Deriving the Trial Policy Gradient

We will now discuss the foundational ideas behind reinforcement learning and policy gradient descent, and how they might be applied in the context of adaptive clinical trial design. Suppose we are running a randomized clinical trial and for each subject enrolled, we observe $\{x, t, y\}$ where x is a vector of covariates, t is the (randomized) treatment, and y is the outcome of interest. Further imagine that once we have enrolled n_1 patients, we get to observe the full trial data (including outcomes) up until that point. We call this the current *state* of the trial, $S_{n_1} = \{x_i, t_i, y_i\}_{i=1}^{n_1}$. At this point, suppose we are allowed to take some action which impacts later observations of the trial. This could be, for example, limiting or increasing the number of future trial participants (sample size re-estimation), early termination of the trial, or restricting the target population we would like to enroll. The set of all permissible actions at this trial state is called the *action space*, denoted by $\mathcal{A}_{S_{n_1}}$. For example, in the context of sample size re-estimation, the action space is the non-negative integers $\mathcal{A}_{S_{n_1}} = \mathbb{Z}^+$, denoting the number of further participants to enroll. Note that we can extend this same framework

over m trial states, and for m action states, instead of just the single pair described above. At the end of the trial, we observe some *utility*, $q(S_n)$, which is a function of all the observations in the trial, denoted by S_n. Examples of such a utility in the context of a clinical trial include a binary variable indicating detection of a significant treatment effect, a z-score for testing a null hypothesis of no treatment effect, or perhaps the estimated average treatment effect itself. Our goal, here, is to maximize the expected value of this utility with respect to the actions that we take in the trial.

A *policy* D_j is a function which takes as input the current state of the trial, and outputs a probability distribution over possible actions.

$$D_j : \mathcal{S}_{n_j} \rightarrow P(\mathcal{A}_{\mathcal{S}_{n_j}}). \tag{3.1}$$

Using the sample size re-estimation example, a trial policy would be a function which takes as input the current observations of the trial, generally after accrual of a specific number of participants, and outputs a non-negative integer, the number of additional participants to enroll in the trial. Let $D = \{D_1, \ldots D_m\}$ be our *trial policy*, a function which gives probabilities of actions at each of the m states in our trial where we take an action. This then yields the following optimization problem for our utility

$$D^* = \text{argmax}_D \, \mathrm{E}_D \left[q(S_n) \right]. \tag{3.2}$$

As it is generally not tractable to solve this optimization problem over all possible trial policies, we instead restrict our optimization to those policies which can be represented by a particular neural network architecture,

$$D^* = \text{argmax}_{D_\beta} \, \mathrm{E}_{D_\beta} \left[q(S_n) \right], \tag{3.3}$$

where D_β is a neural network trial policy with neural network weights β. This policy takes as input, sequentially, each of the actionable states of the trial, $S_{n_1}, \ldots, S_{n_m}$, and outputs probability distributions over $\mathcal{A}_{n_1} \ldots, \mathcal{A}_{n_m}$, respectively. For example, actions might be sample size re-estimation, alteration of enrollment criteria, or early trial termination. While

describing a trial that contains an adaptive decision point is relatively straightforward, it is not clear how to optimally make this decision. If we had some way of constructing an estimate of the gradient of the expected utility with respect to the actions taken by our neural network, we could use stochastic gradient descent to iteratively improve the expected utility.

We will now present a method for iteratively updating β to maximize 3.3. Note that in a Bayesian setting, we can fully simulate a trial and calculate its likelihood. Therefore we can calculate the following, letting $(S_{n_j}, A_{n_j}) = (S_j, A_j)$ for notational simplicity,

$$\mathrm{P}_D(S_1, A_1, \ldots, S_n, A_n) = \mathrm{P}(S_1)\,\mathrm{P}(S_n|S_{k_m}, A_{k_m}) \prod_{j=1}^{m} \mathrm{P}(S_j|S_{j-1}, A_{j-1})\,\mathrm{P}_D(A_j|S_j). \tag{3.4}$$

Where $\mathrm{P}_D(A_j|S_j)$ is the probability of taking action A_j at trial state S_j under trial policy D, and $\mathrm{P}(S_j|S_{j-1}, A_{j-1})$ is the probability of reaching trial state S_j given state S_{j-1} and action A_{j-1}. Note that there is only one term on the right-hand side that depends on the policy, D. Now, letting $S = \{S_1, \ldots, S_n\}, A = \{A_1, \ldots, A_n\}$, starting with 3.3, we have that

$$\begin{aligned} \mathrm{E}_{D_\beta}\left[q(S, A)\right] &= \int \mathrm{P}_\beta(S, A) q(S_n)\,\mathrm{d}(S, A) \\ \implies \nabla_\beta\,\mathrm{E}_{D_\beta}\left[q(S_n)\right] &= \int \mathrm{P}_\beta(S, A) q(S_n) \nabla_\beta \log \mathrm{P}_\beta(S, A)\,\mathrm{d}(S, A) \\ &= \mathrm{E}_{D_\beta}\left[q(S_n)\nabla_\beta \log \mathrm{P}_\beta(S, A)\right]. \end{aligned} \tag{3.5}$$

Where we have now used P_β to denote the probability of a given event under trial policy D_β, a neural network trial policy with weights and biases β. So, we can write the gradient of the expected utility as an expectation. Now we will approximate this stochastically using N simulated trials and use 3.4 to get our final result,

$$\begin{aligned} \mathrm{E}_{D_\beta}\left[q(S_n)\nabla_\beta \log \mathrm{P}_\beta(S, A)\right] &= \frac{1}{N}\sum_{i=1}^{N} q(S_n^i)\nabla_\beta \log \mathrm{P}_\beta(S^i, A^i) + O_p\left(\frac{1}{\sqrt{N}}\right) \\ &= \frac{1}{N}\sum_{i=1}^{N} q(S^i, A^i) \sum_{j=1}^{m} \left(\nabla_\beta \log \mathrm{P}_\beta(A_{k_j}^i|S_{k_j}^i)\right) + O_p\left(\frac{1}{\sqrt{N}}\right), \end{aligned} \tag{3.6}$$

where $A^i_{k_j}$ is the jth action taken in trial i. As β is the weights of a neural network, we can calculate the gradient in 3.6 via backpropagation. Now that we have described how to stochastically approximate the gradient of 3.3, we can train a neural network with stochastic gradient descent. The training procedure is described in 2. It is worth noting that, in practice, we do not use Algorithm 2 exactly, but rather a modified version of it, described in the supplement in Algorithm 3. We avoid using Algorithm 2 exactly because it is computationally difficult to train a neural network based on the entirely of accumulated trial information, S_j. Often, it is much better to use summary statistics of existing trial data as inputs to our neural network instead ($f(S_j)$.

3.2.3 Policy Input Parameterization

In the previous subsection, we described a method for training a neural network to optimize a utility with respect to actions in an adaptive clinical trial. We will now describe what the input of the neural network at each decision point consists of. Often it is either not necessary or not computationally tractable to use the entire trial state S_{k_i} as an input to our network. Rather, we will often use some low-dimensional summary of the trial state. For example, we might use the frequentist maximum likelihood estimate for a parameter of interest, or the estimated treatment effect at the current trial state. Not only does this automatically combine accumulated trial data in a way that we suspect is informative for choosing an adaptive action, it also reduces the complexity of the network architecture. For example, if the input layer is of dimension 10 instead of dimension 100, the number of weights in the first layer of the neural network is decreased tenfold. Further information regarding the specific summary statistics/dimension reduction used can be found in the supplement. In the following section, we present a selection of simulations that detail how our method works as well as highlights its performance.

3.2.4 Trial Policy vs. Indication

In addition to the dimension reduction described in the previous subsection, in practice, we also restrict our neural network to emit trial policies which are simple and clinically relevant. For example, in an adaptive enrichment trial design, our neural network may output a vector θ, and only enroll new individuals for whom $f(\theta, X) > 0$ for some (simple) f. In such a case, the enrollment criterion is a simple function of patient covariates. A high-quality trial policy will select a value of θ which leads to enrollment of individuals for whom there is strong evidence of a larger expected benefit of treatment. The critical distinction that makes the use of neural networks less problematic here than in other biomedical settings is, while the function (f) which determines enrollment is quite simple, the underlying function which determines θ from already accumulated trial information can be quite complex. An illustration of this distinction in the context of our simulations can be found in the supplement.

For example, in an adaptive enrichment design, the set of all possible rules is the set of all functions mapping from the observed portion of the trial, as well as a potential new subject to enroll, to the unit interval (giving the probability of enrollment). However, in practice, we might only consider linear functions of subject covariates. The neural network does not determine this restriction, but rather emits coefficients which denote a particular linear function in this restricted space. Therefore, while we use a black box function (a neural network) to determine the enrollment rule, the rule itself is a simple, linear function. This framework allows for a complex, efficient determination of trial policy, while ensuring that the enrollment rule itself is simple and clinically relevant. While one could also fit a linear model directly to determine enrollment without the use of a neural network, it would not be optimized to maximize utility. Using heuristics or a likelihood-based approach results in poor performance when compared to the use of a neural network. We illustrate this difference in performance in all of our examples in Section 3.3.

Algorithm 2 Learn a Trial Policy

Require: N = batch size, k = learning rate

Set $\beta = \beta_0$

for $0 \leq i \leq$ total iterations **do**

for $0 \leq j \leq$ N **do**

for $1 \leq \ell \leq$ total action states **do**

1. Sample from $\mathrm{P}(S_{k_\ell,j}|S_{k_\ell,j-1}, A_{k_\ell,j-1})$
2. Sample from $\mathrm{P}_{\beta_i}(A_{k_\ell,j}|S_{k_\ell,j})$

end for

1. Calculate $q(S_{n,j})$

end for

1. $\beta_{i+1} \leftarrow \beta_i + \frac{k}{N}\sum_{u=1}^{N} q(S_{n,u})\nabla_\beta \log \mathrm{P}_\beta(S_{\cdot,u}, A_{\cdot,u})$

end for

3.3 Simulations

3.3.1 Two-Block Cutpoint Adaptive Enrichment Design

In this section, we present trial policy reinforcement learning in three simulation settings. In all of them, we consider a single continuous candidate predictive covariate X, a binary treatment T, and an outcome Y. The treatment effect is heterogeneous with respect to X. Specifically, for some unknown *cutpoint*, c, the treatment effect is larger in individuals for whom $X > c$. We conducted and analyzed simulations for binary, continuous, and time to event outcomes.

For each of the three simulations settings, the simulated trials are two-block adaptive enrichment designs. The first block of 100 individuals are enrolled without restriction based on the candidate predictive covariate. These individuals are randomized, 50 are assigned to treatment, and 50 to control. Following this, the adaptive enrichment step takes place. Based on the X, T, and Y values of the initial 100 individuals, we select a value d. For the

second block of 100 individuals, we enroll only patients for whom $X > d$. These 100 subjects are then equally randomized to treatment and control, and we then analyze the trial based on data from all 200 individuals. Our goal is, based on the 100 individuals who we have already observed, to select a value of d which maximizes expected utility. This selection is delegated to a neural network in our framework.

All of our simulations use a feedforward network with 2 hidden layers and 128 nodes per hidden layer. Rather than input the entire trial state for our examples, we instead use a low-dimensional summary of the trial state, for computational efficiency, as well as a way to avoid, with high probability, converging to trivial local maxima (such as unrestricted enrollment in settings where it is not a global maximum). Detailed discussion of dimension-reduction and convergence issues can be found in the supplement.

In order to engage properly with 3.6, our neural-network-based trial policy must have a stochastic component to its output when training. Without this distinction, we cannot iteratively improve performance as there is no notion of a gradient with respect to the actions taken, and we therefore cannot update the weights of our neural network based on the results of each trial. To deal with this issue, we add Gaussian noise to each output indication with $\sigma^2 = 0.02$. The value of the variance was hand-tuned to produce high-quality policies in these simulation settings. When evaluating the network after training, we remove this noise. More information on how the simulations are run and the heuristics we compare to can be found in the Supplement.

Binary

Suppose we have a binary outcome, $\mathcal{Y} = \{0, 1\}$. As [75] note, if we use the test statistic

$$\hat{z} = \frac{1}{\sqrt{n/2}} \sum_{k \leq K} \sqrt{n_k/2} \left(\frac{p_{(\hat{T},k)} - p_{(\hat{C},k)}}{2\sqrt{p_{(\hat{\text{pool}},k)}(1 - p_{(\hat{\text{pool}},k)})/n_k}} \right), \tag{3.7}$$

where $p_{(\hat{T},k)}$ is the mean outcome under treatment in block k, our test statistic will be asymptotically $N(0, 1)$ even if enrollment in the second block depends on covariates X, and

will therefore have the correct prespecified Type 1 error. Consider the following scenario

$$P(Y = 1|X = x, T = t) = p_0 + k\mathbf{1}(x > c, t = 1)$$

where $X \sim U(0, 1)$. So for some unknown cutpoint, c, subjects with $x > c$ receive a benefit from treatment. After enrolling the initial 100 subjects, we adaptively select a cutpoint, $\hat{c}$, above which to enroll 100 subjects in the second block. For this simiulation setting, we have the following prior:

$$c \sim U(0.1, 0.8),\ k \sim U(0.1, 0.4),\ p_0 = 0.5.$$

For our neural network, we input the empirical distribution function for the cutpoint value,$F(c|S_{100})$, as well as the score for the first block.

The utility we would like to optimize is

$$U(z, n) = \mathbf{1}(z > 1.645)\mathbf{1}(n < 1000) - \lambda \log(n) \tag{3.8}$$

where z is the test statistic and n is the number of subjects screened in the second block. So we have a hard cap of 1000 subjects screened, as well as a penalty for the log number of subjects screened. For the purposes of comparison, we compare to a likelihood-based trial policy which determines a trial policy, or $\hat{c}$, based on quantiles of the empirical distribution for c. As we increase the quantile value across simulations, the average $\hat{c}$ value increases, and so does the average number of subjects screened. We trained feedforward neural networks with a variety of penalty values to produce a range of optimal expected numbers of subjects screened. The results of these simulations can be seen in Figure 3.1.

Continuous

Suppose we have a continuous outcome, $\mathcal{Y} = \mathbb{R}$. We can modify our test statistic from Section 3.3.1 to obtain one that is similarly valid for a continuous outcome

$$\hat{z} = \frac{1}{\sqrt{n/2}} \sum_{k \leq K} \sqrt{n_k/2} \left(\frac{y_{(\hat{T},k)} - y_{(\hat{C},k)}}{2\hat{s}_{(\text{pool},k)}} \right), \tag{3.9}$$

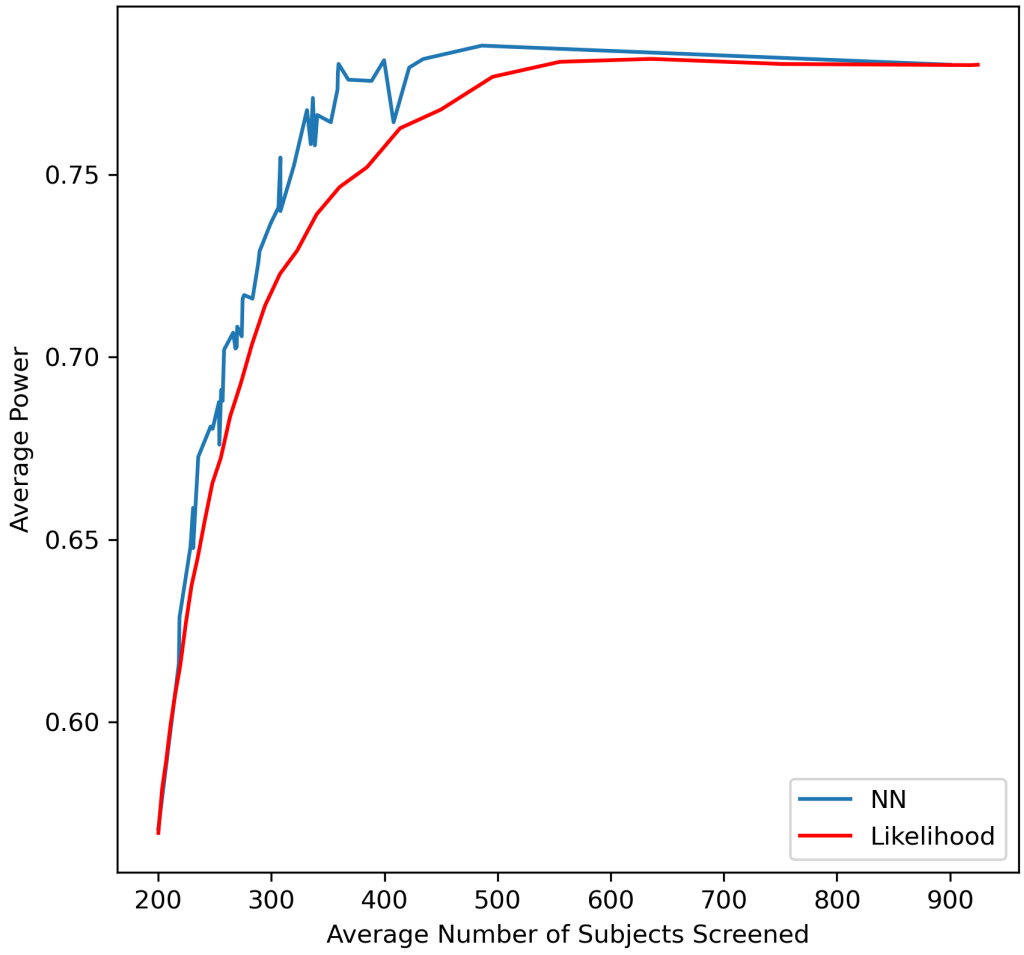

Figure 3.1: Power vs. average number of subjects screened for likelihood-based and neural network-based trial policies. For the same number of subjects screened, the neural network achieves a slightly higher power on average.

where $\hat{s}$ is the sample standard error.

Consider the following scenario

$$Y|X = x, T = t \sim N(\beta_0 + k\mathbf{1}(x > c, t = 1), \sigma^2)$$

where $X \sim U(0, 1)$. So for some unknown cutpoint, c, subjects with $x > c$ receive a benefit from treatment. After enrolling the initial 100 subjects, we adaptively select a cutpoint, $\hat{c}$, above which to enroll subjects in the second block, and then enroll another 100 subjects. For this simulation setting, we have the following prior:

$$c \sim U(0.3, 0.9),\ k \sim U(0.8, 2.3),\ \beta_0 = 0,\ \sigma^2 = \sqrt{2}.$$

As a rule for comparison, we use a likelihood-based rule which uses quantiles of the empirical distribution for c to select the cutpoint for enrollment in the second block. For our neural network, we input the empirical distribution for the cutpoint, as well as the score for the first block. We use the same utility as in 3.8. Results of these simulations can be seen in Figure 3.2.

Time-To-Event

Suppose we have a time-to-event outcome, $\mathcal{Y} = \mathbb{R}^+$.We can modify our test statistic from Section 3.3.1 to obtain one that is similarly valid for a time-to-event outcome

$$\hat{z} = \frac{1}{\sqrt{K}} \sum_{k \leq K} \frac{\ell_k'(0)}{\sqrt{-\ell_k''(0)}}. \tag{3.10}$$

This is simply a weighted sum of log-rank tests, and will be asymptotically $N(0, 1)$ under the null hypothesis. Note that while it is possible to come up with a more efficient weighting scheme, we apply uniform weights to the blocks in this setting.

Consider the following mean model

$$T|X = x, S = s \sim \text{Exp}(\beta_0 + k_0\mathbf{1}(x > c, s = 1))$$
$$P(Y = 1|X = x, S = s) = \beta_1 + k_1\mathbf{1}(x > c, s = 1).$$

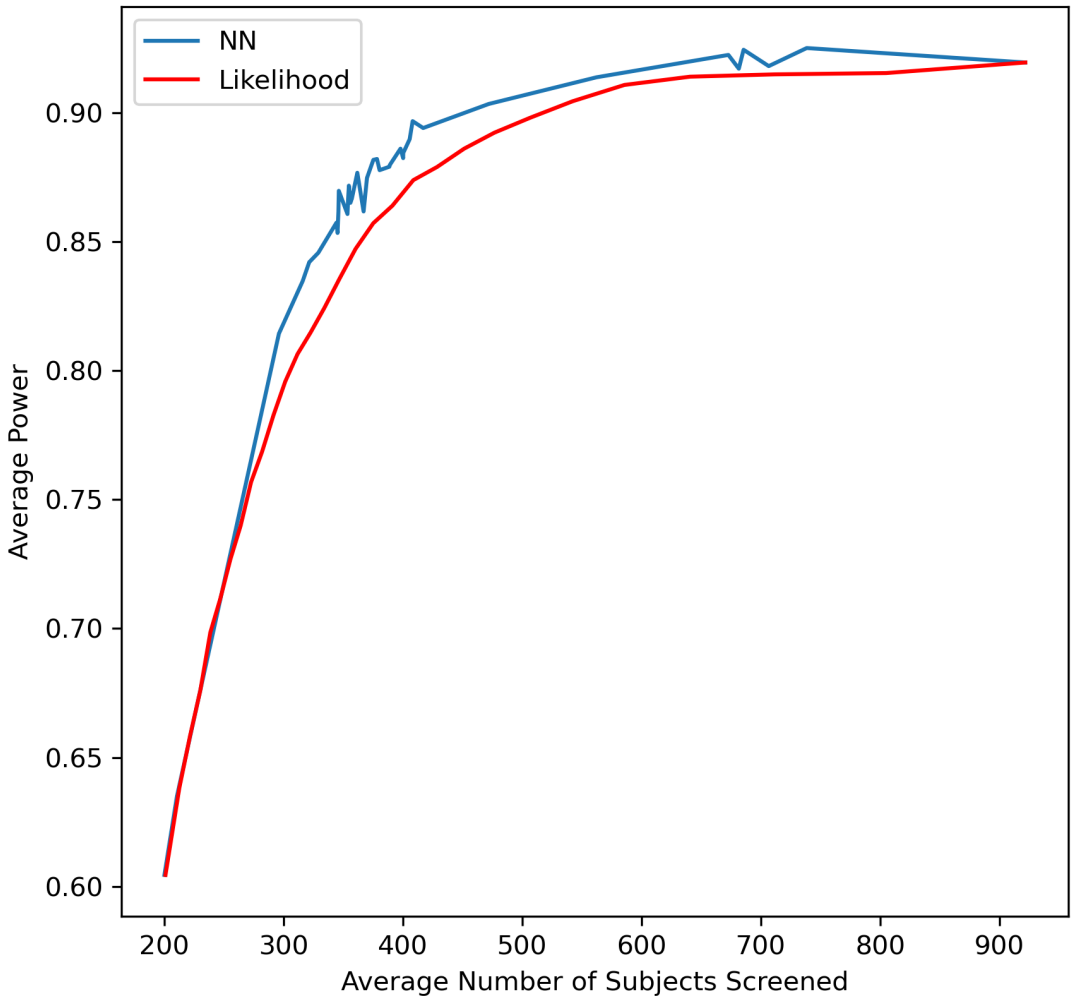

Figure 3.2: Power vs. average number of subjects screened for likelihood-based and neural network-based trial policies. For the same number of subjects screened, the neural network achieves a slightly higher power on average.

This scenario is more complex. We have the time-to-event T, as well as a biomarker Y. The expected values of both are influenced by the unknown cutpoint, c. We have the following prior for this simulation

$$c \sim U(0.2, 0.8),\ k_0 \sim U(0.7, 2.4),\ k_1 \sim U(0.2, 0.8),\ \beta_0 = 2,\ \beta_1 = 0.5.$$

In this simulation setting, we enroll an initial cohort of 100 subjects without any criterion for enrollment as before. At time $t = 2$, we get to see the event times which have occurred, as well as the biomarker y for all subjects. We then select a cutpoint, enroll another 100 subjects, and proceed as in the previous simulations. For simplicity in this simulation, all 100 subjects in the first block are enrolled at time $t = 0$, and all 100 subjects in the second block are enrolled at time $t = 2$. We compare to a likelihood-based approach which selects quantiles of the empirical distribution for c to select the cutpoint for the second block. Our neural network takes as input this distribution function, as well as the score for the first block. We use the same utility as in 3.8. Results from these simulations can be seen in Figure 3.3.

3.4 Discussion

In this manuscript, we have proposed a method for using reinforcement learning to develop a Bayesian adaptive clinical trial policy. Provided we can simulate trials under a particular policy and have a well-defined utility we would like to optimize, we can train a neural network to make decisions which optimize this utility. We illustrated this method in three different settings, and showed that a reinforcement learning-based trial policy outperforms a simple likelihood-based policy in all of these settings.

We only engaged with a single utility function throughout our simulations. It is trivial to include other elements in the utility, or use a different utility function entirely. For example, in multi-site clinical trials, perhaps to cost of recruitment is not equal between sites. This complexity could easily be incorporated into trial simulation, and therefore into the framework discussed here. In short: if the feature we would like to optimize is a function

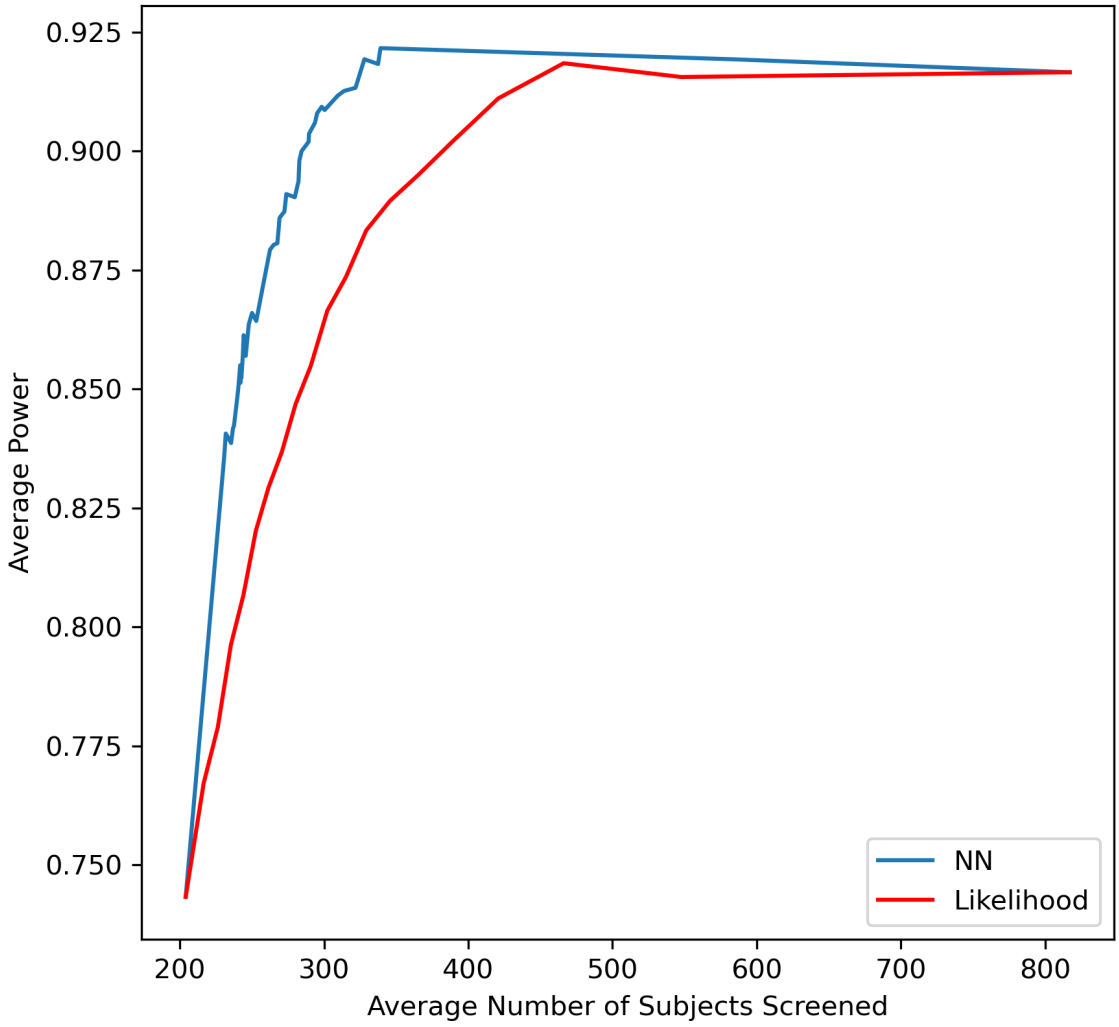

Figure 3.3: Power vs. average number of subjects screened for likelihood-based and neural network-based trial policies. For the same number of subjects screened, the neural network achieves a slightly higher power on average.

of information available at the conclusion of the trial, we can include it in our utility, and the neural network will automatically design a policy which will optimize the modified utility. This flexibility will likely become advantageous as the complexity of the trial design increases.

For simplicity, we considered only adaptive enrichment designs, and trials which only contained a single adaptive decision step, in our simulations in Section 3. However, actual trials are generally more complex in their design, and have a larger number of baseline covariates. One can easily extend the framework described in this work to trials with multiple baseline covariates, multiple adaptive decision steps, and multiple adaptive decision types. One example of such a trial would be a three block adaptive enrichment trial with sample size reestimation after the second block. More work is needed to investigate the efficacy of reinforcement learning in more complex trial settings.

We used an extremely simple reinforcement learning architecture for these trials. Our neural network was a feedforward network which took as input summary statistics of the trial, and output a cutpoint for the second block. Neither the feedforward network nor the use of a single network are optimized for this problem setting. Recent developments in reinforcement learning, such as Actor-Critic reinforcement learning [26], could be leveraged here to improve performance. Moreover, we used summary statistics for the purpose of computational tractability. Using the entire first block (treatment, covariates, and outcome) as an input to a more sophisticated reinforcement learning architecture should be investigated in the future and may improve performance.

3.5 Introduction

Researchers in many biological subfields have developed detailed mechanistic models describing biochemical processes. These models are often qualitative, not quantitative: for example, the experiments underlying the stated biochemical process may have shown that a certain enzyme is necessary for one part the process, however, these models typically do not describe the rates or relative importances of the different pieces. Given appropriate data, one would like to infer rates and relative importances of the processes. Here we investigate somatic hypermutation (SHM), the biochemically-driven mutational process underlying antibody maturation, as an example of such a process.

Biological theories such as the one describing SHM are often given in forms that are easily translated to probabilistic models. Setting up such a model and performing parameter inference can be useful, allowing us to quantify parts of the process that are either not experimentally manipulable or not directly measurable. This can give us insight into the process under normal circumstances and facilitate comparisons of the process under different conditions.

To describe the biological context for this chapter in a little more detail, we will now give a brief introduction to B cell receptors and affinity maturation. B cell receptors (BCRs), also known as immunoglobulins or antibodies in their secreted form, are proteins designed to bind and identify foreign molecules. These BCRs must be incredibly diverse because of the wide diversity of foreign molecules, such as viral proteins, that need to be bound by them. They are generated first by a random process of V(D)J recombination, which allows for high diversity in BCRs with only a small number of genes. If they pass tests against self-reactivity and for functionality, and are stimulated by binding foreign antigens, the corresponding B cells will enter specialized regions of the lymph nodes where they undergo a process called "affinity maturation." In affinity maturation, BCRs diversify according to a Darwinian process of mutation and selection, with the objective of improving binding to antigen. More specifically, BCRs which exhibit high binding affinity undergo positive selection. While

V(D)J recombination increases BCR diversity through combining germline genes randomly, additional diversification through mutating the genes themselves is also necessary to adapt to new antigens.

Our focus is on somatic hypermutation (SHM), the mutation process of affinity maturation, which has a rate in the BCR-coding region of around a million-fold higher than the mutation rate of normal cellular division. This process is driven by specialized biochemical pathways as described below. Accurate estimation of mutation probability during SHM is important for understanding which antibodies are accessible from a certain naïve progenitor cell. For example, it has been shown that specific low-probability mutations are important for broadly-neutralizing HIV antibody maturation [7, 31, 91]. Thus to understand prospects for eliciting such antibodies through vaccination, and how to design vaccine inserts rationally [33], we must estimate mutation probability accurately. Accurate mutation rate estimation is also important for natural selection assessment [18, 27, 83, 94, 96, 47] in which the mutation rate (the introduction rate of nucleotide changes) is compared to the substitution rate (the rate of such changes that survive natural selection).

Current research on SHM can be primarily divided into two approaches: biochemical and statistical. The biochemical approach uses gene knockout experiments in model organisms to elucidate mechanisms of somatic hypermutation. As a prototypical example, [98] use a mouse with a specific DNA substrate spliced in to test the effect of the "Ung" pathway for generating mutations at adenine (A) and thymine (T) bases. They found that when the *Ung* gene was knocked out, the mutation patterns at A/T bases were significantly changed. This experiment clearly demonstrates the importance of the Ung pathway, but does not provide an estimate about the frequency with which it is used.

The statistical approach, on the other hand, works to characterize the mutation process in humans and wild-type animals by developing predictive quantitative models. It typically works to understand the influence of local sequence context, namely the surrounding DNA bases, on mutation probability. For example, if a cytosine base is contained in a "hotspot" such as AGCT, many studies have shown that this is sufficient for a greatly increased muta-

tion rate [97]. "Coldspots" such as CCC have a lower rate of mutation. Early work analyzed these sequence contexts using a hypothesis-testing approach [e.g. 64] while more recent work has performed rate estimation for highly parameterized models [94, 15]. Other work has included a penalized proportional hazards model [22], models that include absolute position in the sequence [79], and deep neural networks on larger tracts of sequence [81]. Such models are useful for predicting the path of affinity maturation [17, 91] and analyzing natural diversification patterns [96, 73, 87]. Furthermore, many of these papers provide information that can clarify mechanistic understanding; for example [95] investigate DNA strand symmetry, while [81] interpret their mutability scores in terms of pathways. Many other papers use the pattern of mutations to describe the qualitative importance of the different processes in SHM [19, 78, 65, 64, 53, 92, 88]. Nevertheless, this work has not resulted in rate estimates for the various repair pathways *in vivo*.

It would seem useful to be able to bring these two approaches to SHM research together by fitting parameters of a probabilistic model of SHM; such a model should in principle be able to be fit with less data by using prior mechanistic knowledge, and should be useful by providing estimates of parameters with direct biological relevance. However, such a project faces several obstacles. First, there are a tremendous number of latent variable patterns that lead to identical mutation locations, and thus calculating a likelihood for a set of mutations given a starting sequence is not possible. Second, the mechanistic model may not be identifiable from the data: there may be many parameter settings leading to identical distributions of sequences of mutations. Third, although it is common to have tens of thousands or more mutated sequences from a given data set, and each sequence is hundreds of bases long, mutations are relatively few and there is a lot of stochasticity in the locations of the mutations. Thus the signal for fitting a complex model may not be as strong as it initially appears.

In this chapter, we take on this challenge by using a method combining neural networks and strategies from Approximate Bayesian Computation (ABC) to infer parameters of a complex latent variable model of SHM. We draw parameter values from a prior distribution,

simulate data given the parameter drawn from the prior, and compute summary statistics from the simulated data. We then use the joint distribution of parameters and summary statistics as a training set for a neural network, which approximates the posterior expectation of the parameters given the summary statistics. This approach is similar to others who have used neural networks to estimate quantiles of the posterior distribution directly [23], but we use summary statistics to generate these values. Unlike some other approaches at using neural networks to improve the performance of ABC, which use the neural network to craft the summary statistics [34], we create the summary statistics ourselves, and then use those statistics as inputs to a neural network. Others have attempted to use deep learning in the context of more classical statistical techniques [35, 72], though generally these works use neural networks to generate informative statistics, rather than generate parameter estimates from pre-selected summary statistics. Biologically, we find that some parameters of the model can be readily identified, however some parameters appear to be more difficult to estimate. Our results also show some deficiencies of the core model of SHM, and indicate what additional components are required to have a more complete description of the process. Our method focuses on the sequence-level mutation distribution, at the expense of accurately quantifying local context mutability. In order to guide further research in the area, we also review alternative procedures we tried which were not successful.

3.6 Methods

3.6.1 Overview of somatic hypermutation

Now we summarize, for the purposes of formalizing our model, current mechanistic understanding of the somatic hypermutation process. The model presented here is necessarily an oversimplification in order to obtain a model that can be fit. Further details on the simplifications employed are in the Discussion section, though others have described the intricacies of the process extensively [61].

The underlying mechanism of SHM involves an initial step of DNA damage by the

Activation-Induced cytidine Deaminase (AID) enzyme, which transforms a cytosine DNA base to a uracil. AID processes along the sequence [59, 11, 45], leading to spatial clustering of AID-induced damage. This defect is resolved by one of several pathways (Fig. 3.4). In the Base Excision Repair (BER) pathway, the uracil is removed by the Uracil-DNA glycosylase (UNG) enzyme, and then special enzymes are used to read through the corresponding abasic site, with additional local mutation from translesion polymerase REV1 [32] and potentially from error-prone DNA polymerases. In the mismatch repair (MMR) pathway, enzymes, EXO1 in particular, strip DNA around a region of the mismatch, which is filled in by error-prone polymerases. This region is referred to as the *exo stripping region*. Both of these processes occur on both strands of DNA. When DNA is replicated, the mutations from one strand are propagated to the other strand.

We have not attempted to model insertion-deletion mutations as part of somatic hypermutation [8, 43]. As such, we can assume that every mutation is a point mutation on a sequence of fixed length. We also do not model certain minor pathways (e.g Polζ [70]) and assume that their effects will be absorbed into our estimates of the two main pathways. In the true process, earlier repairs can have two effects on later repairs that we ignore here. First of all, an earlier repair by the MMR pathway can strip out a lesion, meaning that the repair machinery would not be recruited to that site. Secondly, an earlier repair can either change a C to another base, meaning that the base is not actually available to be deaminated by AID, or an earlier repair can change a non-C base to a C, allowing it to be deaminated by AID. We don't consider either of these possibilities in our model.

We will fit probabilistic parameters to this simplified consensus model.

3.6.2 Probabilistic graphical model formulation of somatic hypermutation

Our mechanistic model translates a standard biochemical SHM model (Fig. 3.4) into a probabilistic graphical model.

To generate samples from this model, we first sample a set of AID lesions, which we will call "lesions" for short, then sample repair types for each lesion. The repair process is

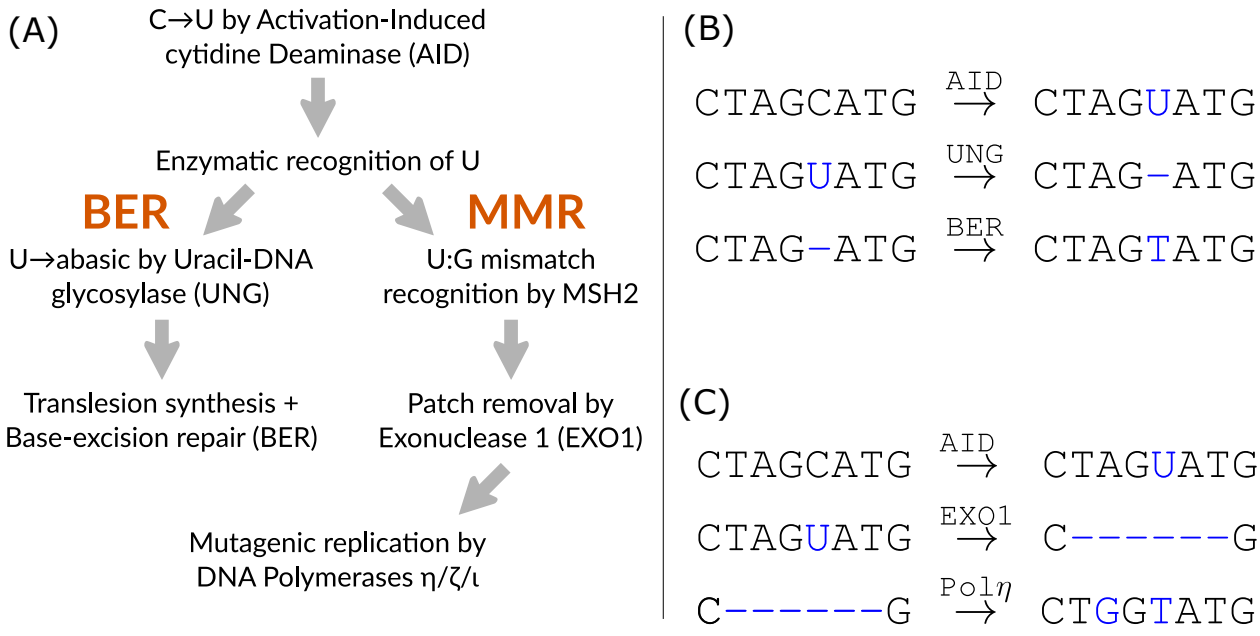

Figure 3.4: The core "consensus model" of the biochemical basis of somatic hypermutation (SHM). (**A**) AID lesion, recognition, and error-prone repair through the non-canonical base excision repair (BER) and mismatch repair (MMR) pathways. (**B**) Error-prone base excision repair (BER) induces a point mutation at a C site with some probability. (**C**) Error-prone mismatch repair (MMR) induces potentially multiple point mutations in a window of stochastic size, or *exo stripping region*, around a C site with some probability.

modeled by sampling a pathway to repair each of the lesions, with random variables governing recruitment probabilities of either the BER or MMR repair machinery to each site. A repaired sequence is then drawn given this choice, with errors introduced by respective error-prone machineries [49].

We now describe this forward model in detail.

Forward model

To model this process, we start with a sequence of length L. We will call this sequence the "parent" sequence because in phylogenetic terms the new sequence will descend from the parent sequence. Our random variables are:

- $A = \{a_k\}_{k=1}^K$, $a_k \in \{0, \ldots, L-1\} \times \{0, 1\}$.

  We have K AID lesions, and a_k gives us the position along the sequence and strand of the kth lesion.

- MMR_k, $k = 1, \ldots, K$, $\text{MMR}_k \in \{0, 1\}$

  $\text{MMR}_k = 1$ indicates that the MMR repair machinery was recruited to repair the kth lesion.

- E_k^l, E_k^r, for k such that $\text{MMR}_k = 1$.

  E_k^l and E_k^r give the number of bases stripped out to the left and the right of the lesion. When BER is selected, these variables are 0.

- S_i, $i = 0, \ldots, L-1$, $S_i \in \{\text{A}, \text{T}, \text{C}, \text{G}\}^2$.

  S_i represents the nucleotides on the top and bottom strand at position i in the mutated sequence. We assume that these nucleotides satisfy Watson-Crick-Franklin pairing. These S are our observed data.

One important feature of our lesion vector A is that the lesion locations should be spatially correlated. To achieve this, the lesions are the result of thinning what we call *prelesions*. At each site k, there is some prelesion probability p_k. These probabilities are first sampled independently across sites to obtain the prelesion set. Following this, we thin the prelesions according to a Gaussian process with a particular mean, variance, and lengthscale. In particular, we include all prelesions only at sites where $G_k > 0$, where G_k is the value

of the Gaussian process at site k. We choose to do this in order to enable strong spatial correlation due to AID moving in a processive manner along the sequence [59, 11, 45] and mesoscale-sequence effects on AID deamination deriving from local DNA sequence flexibility [89]. Lesions at sites which are close together in a sequence are positively correlated. This distribution for the lesions is called a *Probit Gaussian Cox Process* (PGCP), a modification of the sigmoidal Gaussian Cox process [51]. A more detailed definition of a PGCP can be found in Section C.3. Instead of applying the sigmoid function to the Gaussian process to obtain the thinning probability, as in a sigmoidal Gaussial Cox process, prelesions are thinned if and only if the Gaussian random variable associated with them is negative.

The distributions of all of our random variables are:

$$\begin{aligned} A &\sim \text{PGCP}(\mu, \sigma, \ell) \\ \text{MMR}_k &\sim \text{Bern}\left(p_{\text{MMR}}\right) \\ E_k^l &\sim \text{Geom}(p_l) \\ E_k^r &\sim \text{Geom}(p_r) \\ S_i \mid A, \text{MMR}, E^l, E^r &\sim \text{Categorical}(\pi_i) \end{aligned}$$

The parameters are

- $\mu \in \mathbb{R}$, $\sigma \in \mathbb{R}^+$, $\ell \in \mathbb{R}$: Parameters controlling the overall rate and correlation of the lesion generating Cox process. For the purposes of identifiability, we will assume that μ is fixed at -10.0.

- p_{MMR}: Parameter controlling the relative rate of MMR recruitment to BER recruitment ($p_{\text{MMR}} = 1 - p_{\text{BER}}$).

- p_l, p_r: Parameters controlling the length of the exo stripping region.

- p_{fw}: Parameter controlling the relative rate of transcription of the forward strand.

- π_{Mt}, π_{Mb}, π_{Bt}, π_{Bb}: 4×4 transition probability matrices, assumed known, describing the transition from nucleotide (on the top or bottom strand) after MMR on the top or bottom strand, and BER on the top or bottom strand, respectively. We sample a base at site i depending on which pathway (if any) has been recruited to repair that site.

Our estimand is therefore $\theta = \{\sigma, \ell, p_{\text{MMR}}, p_l, p_r, p_{\text{fw}}\}$.

For our choice of π_B and π_M, we reference work from [42]. They find that Ung, the enzyme largely responsible for BER, induces mutations in a particular pattern. When examining mice who were deficient in Msh2, the enzyme believed to be responsible for mutations on the MMR pathway, they found that 75.4% of Guanine mutations were to Adenine, and 77.9% of Cytosine mutations were to Thymine. They found similar preferential mutation patterns in Ung-deficient mice, which should in our framework have mutations exclusively due to the MMR pathway. These results simplify our model greatly, as they allow the source pathway of an observed mutation to be determined with higher confidence.

The conditional random variable $S_i \mid A, \text{MMR}, E$ is distributed according to static mutation probabilities, conditional on the unmutated base at site i. This means that, conditional on the location and size of exo stripping region, we apply the same transition matrix across the entire region. For the BER pathway, we apply the same transition vector to every affected site. It is important to note that this implies an independent-across-sites assumption for the post-pathway recruitment mutation. Our transition probability matrices tell us about the effect of either repair by MMR or BER on this nucleotide. If there is a lesion that is not repaired before replication, it is treated as a normal base. For the sake of simplicity, we assume that this "no repair" pathway is absorbed into our estimates for the BER transition vector.

3.6.3 Estimating model parameters with Neural Network ABC

One of the problems with estimating parameters in settings with latent variables, such as A, W, and E here, is that we cannot easily calculate $\mathbb{P}(\theta \mid S)$, even via stochastic techniques

(e.g. MCMC) which only require calculating $\mathbb{P}(S \mid \theta)$. The difficulty in calculating $\mathbb{P}(S \mid \theta)$ stems from having to integrate over all latent states which could have generated S, which are too numerous to be computationally tractable.

In order to estimate $\mathbb{E}(\theta \mid S)$, we use a neural network-based estimation procedure which is similar to approximate Bayesian computation (ABC), however in our case the target is posterior mean estimates of the parameters rather than a full approximate posterior. A visual description of the estimation framework can be seen in Figure 3.5. In ABC-based methods, one generates data under a candidate parameter set θ'. The resulting parameters, θ', are then accepted or weighted by some metric ρ of similarity of S' to observed data. Typically, ρ is based not on the high-dimensional observed data S, but on some low-dimensional summary $f(S)$. In our framework, rather than needing to specify ρ to obtain an entire approximate posterior, we instead input $f(S)$ into a neural network, which then outputs an estimate of θ'. Our proposed method differs from ABC in that we are not attempting to estimate/sample from the posterior distribution, but rather trying to estimate the posterior mean. It is similar in that both approaches use summary statistics to deal with high-dimensional data.

It is important to note that the true minimizer of mean-squared error with respect to the joint distribution of $\theta, S(X)$, where $S(X)$ are the summary statistics of sequences X simulated under mechanistic parameters θ, is the posterior mean,

$$\arg\min_{f} \mathbb{E}((f(S(X)) - \theta)^2) = \mathbb{E}(\theta \mid S(X)).$$

Here, the expectation is taken with respect to the joint distribution of $\theta, S(X)$. Therefore, provided our neural network architecture is sufficiently rich, and our summary statistics capture the majority of the relationship between X and θ, we will be able to approximate the posterior mean using this framework. By generating many pairs of parameter sets and corresponding simulated sequences, we can create a large pool of training data, $\{\theta'_i, f(S'_i)\}_{i=1}^{N}$. We can then train a neural network to approximate $\mathbb{E}(\theta \mid f(S))$ using mean-squared error as our loss function. Provided the neural network is sufficiently rich, and our function $f(\cdot)$ captures most of the relevant information about θ present in observed sequences, we will

train a good estimator of $\mathbb{E}(\theta \mid S)$.

This framework allows us to move from a high-dimensional latent-state Bayesian inference setting to a low-dimensional supervised learning setting. Provided we can simulate a large enough set of sequences, we can estimate $\mathbb{E}(\theta \mid S)$ without ever estimating the latent states which define the association between θ and S. Most of the difficulty in estimation with this framework lies in specifying the correct neural network architecture, and constructing informative summary statistics for S.

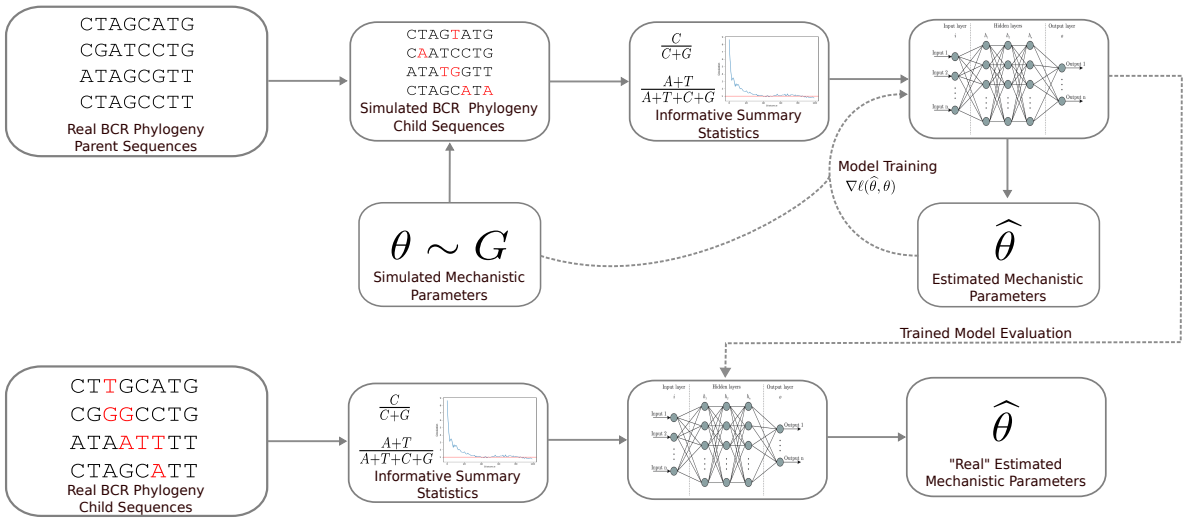

Figure 3.5: Visual description of model estimation framework. The neural network is trained on simulated parameter-sequence pairs. The trained model is then applied to real sequences to make estimates for the natural process.

3.6.4 Summary Statistics

We will now describe the various summary statistics we use and how they aid in inferring key mechanistic parameters.

Spatial Colocalization

One of the key components of our set of summary statistics is the spatial colocalization [79], which describes the observed clustering of mutations in SHM. Unlike local models of mutability such as S5NF [95], our mechanistic framework allows for an elevated frequency of close-together mutations, as seen in real sequences. Local models use several bases on either side of a particular site to model the probability of mutation at that site; the most common is a 5-mer model, which uses two bases on either side. These lack the complexity to model correlation in mutations. A visualization of the shortcomings of a 5-mer model in this respect can be seen in Figure 3.6. Other works have incorporated wider sequence context in probabilistic modeling, but attempts to mimic the bunches of mutations seen in real sequences generally involve a second round of "follow-up" mutations which explicitly occur near existing mutation [e.g. 79]. However, such models are phenomenological rather than explicitly mechanistic, and still render the likelihood intractable.

More formally, spatial colocalization measures the degree to which mutations occur closer together than what one might expect were they spatially independent. The spatial colocalization of Spisak et al. [79], for a pair of sites indexed by i, $i + d$, is defined as

$$\frac{p(i, i + d)}{p(i)p(i + d)} \tag{3.11}$$

where $p(i)$ is the probability of a mutation at site i. So, if the two sites are independent, the co-localization value will be 1. We are interested in the colocalization averaged over all pairs distance d apart. This statistic quantifies the spatial effect on co-mutation over an entire dataset/repertoire.

Estimating this quantity for real sequence data presents some challenges. If all observed sequences were globally aligned, and we had unlimited data, we could estimate this fraction empirically for every site. However, in practice, it is possible to have many different parent sequences from which observed sequences mutate. It is also possible that we have few mutations per site, making the variance of our estimates of the pairwise colocalization high. These issues make estimation of Equation 3.11 infeasible for the data we will be engaging

with in this work.

Instead of the naive version proposed above, we must use an alternative estimate for the spatial colocalization at distance d. If we have N sequences, sequence i has length ℓ, m_i^r is the number of pairs of mutations a distance r apart from each other, and $m_i^+ = \binom{\#\text{ mutations in sequence } i}{2}$ is the total number of pairs of mutations in sequence i, the "SWM" [79] colocalization estimate is

$$\text{coloc}_{\text{SWM}}(r) := \frac{1}{N}\sum_{i=1}^{N} \frac{m_i^r/m_i^+}{(\ell_i - r)/\binom{\ell_i}{2}}. \tag{3.12}$$

This version of spatial colocalization can be thought of as the fraction of pairs of mutations distance r apart, compared to the fraction of total sequence pairs which are distance r apart. Critically, this quantity is estimated per-sequence, and does not require us to estimate the per-site mutation frequency anywhere. The result is that the estimates of the spatial colocalization are more stable in smaller samples, and in samples where there are only a small number of mutated sequences for each parent sequence.

Mutation Frequencies

There are various mutation frequencies that are informative summary statistics. The first is the overall mutation frequency, or the fraction of total bases which differ from the parent sequence. We care about the overall mutation frequency because it tells us about the relative rate of prelesion thinning. The second is fraction of total mutations for which the parent base is A or T. We care about this fraction because A/T mutations can only occur on the MMR pathway. The final mutation fraction that we are interested in is the total number of mutations where the parent base is C, divided by the total number of mutations where the parent base is C or G. This is because the BER pathway can only occur at a site which is C on the strand which is repaired.

Distance Between Mutation-Associated Sites

Finally, we include two distance metrics in our set of summary statistics. The first is the average distance of a mutation at an A/T site from the nearest C/G site. The idea behind this summary statistic is that, for a larger exo stripping region, A/T mutations will be slightly farther away from C/G sites on average. The second distance metric is the pairwise A/T mutation distance. This measures the mean pairwise distance between mutations at A/T sites. We include this summary statistic because, for small exo stripping regions, we expect A/T mutations to cluster closer together.

3.6.5 Parameters of interest

We will now describe parameters of interest in our model, as well as how the aforementioned summary statistics allow us to infer them.

Gaussian Process Variance (σ)

The Gaussian process variance controls the overall prelesion thinning probability by governing the variability of the underlying Gaussian process. We take a uniform prior from 5.0 to 15.0 on the standard deviation of the Gaussian process.

Along with the base rate, the variance of the Gaussian process affects the overall fraction of C sites which are deaminated by AID as follows. Recall that the mean of the Gaussian process is fixed at -10.0. This means that the probability that a prelesion is thinned is proportional to

$$\int_{10/\sigma}^{\infty} e^{\frac{-x^2}{2}}.$$

In terms of summary statistics, a large value of σ, along with a high base rate, contributes to a high overall mutation frequency. Furthermore, a smaller value of σ corresponds to a larger colocalization magnitude (higher colocalization when $d = 1$).

Therefore, we include distance 1 spatial colocalization, as well as the overall mutation frequency, with the idea that it will help estimate the variance of the Gaussian process.

Gaussian Process Lengthscale (ℓ)

The Gaussian process lengthscale controls the spatial scale of correlation in deamination probability. We have a uniform prior from -12.0 to -2.0 for the log of the lengthscale. The lengthscale affects the rate of decrease of the colocalization.

Recall that the covariance of the Gaussian process at two sites is $\sigma^2 e^{\frac{-d^2}{2\ell^2}}$, where d is the distance between the sites, and ℓ is the lengthscale. Therefore, the smaller the lengthscale, the more rapidly the spatial colocalization falls off in terms of the lesions.

To help estimate the lengthscale, we simply include the spatial colocalization vector from distance 1 to distance 100 as described above. The rate at which the colocalization decays to 0 is used by the neural network to estimate the lengthscale.

BER Probability p_{BER}

This parameter describes the probability that the BER repair pathway will be recruited to a lesion. If not repaired by BER, a lesion is repaired by MMR. We have a uniform prior from 0 to 1 for this parameter. Notably, MMR allows for mutations in a neighborhood around the lesion, rather than just at lesion itself. Summary statistics which contain information about this statistic include the fraction of mutations occurring at A/T sites, as well as the average distance of A/T mutations from a C/G site. These are informative for the BER probability parameter because a higher frequency of these mutations, as well as these mutations occurring closer to C/G sites, suggests a lower fraction of lesions which are repaired by BER.

Forward Probability

This parameter controls the probability that the given round of mutation comes from the forward strand. We have a uniform prior from 0 to 1 for this parameter. A higher forward strand probability parameter implies that we see a disproportionate number of sequences originating from the forward strand.

The main summary statistic that informs this parameter is the fraction of C/G mutations

which are at C sites. C site mutations are more likely on the forward strand than on the reverse strand, because C mutation can only occur on the MMR pathway on the reverse strand, whereas they are possible from both pathways on the forward strand.

Exo Left/Right Stripping Means

These two parameters control the mean size of the exo stripping region on either side of the lesion when the MMR pathway is recruited. The parameter is the mean length of the stripping region, which is geometrically distributed. We have a uniform prior from 1 to 20 for the mean stripping region size.

Summary statistics which contain information on these parameters include the average distance to the left, and to the right, of C/G sites from A/T mutations. These parameters are the most difficult to estimate in this model, because it is difficult to determine from the observed data if two A/T mutations exist in the same exo stripping region, or if they originated from different lesions.

3.6.6 Data

In order for our model fit to reflect exclusively the mechanistic properties of somatic hypermutation, it is critical that we train it on data which are free of the impact of natural selection. If we attempted to estimate the mechanistic parameters on sequences which were subject to the full process of affinity maturation, it would be impossible to determine which patterns observed were due to selective pressure, and which were due to SHM mechanisms. Therefore, we engage with out-of-frame sequence data, sequences which are non-functional due to codons being out of frame after the process of VDJ recombination. Specifically, we use the use the out-of-frame clonal family clustering and multiple sequence alignments from [79], who use additional filtering criteria to make sure that the data represents neutral evolution. From these, we filter out all sequences containing insertions or deletions as these are not a part of our mechanistic model. For each of these clonal families, we use IQ-TREE [54] to infer the phylogeny, or evolutionary history of the clonal family. Ancestral sequences are

inferred using the K80 substitution model [37], with known germline V and J genes used as outgroups. Following phylogenetic inference, we consider each edge of the tree to be an observation, with the mechanistic SHM process acting independently on each edge. In total, our dataset contained $84,322$ parent-child pairs.

3.6.7 Training Setup

We trained our neural network on out-of-frame human high-throughput sequence data [79]. To generate each simulated dataset, we sampled 2000 parents at random from phylogenies inferred from the out-of-frame sequence data, as well as a set of mechanistic parameters (θ from its prior distribution above). For each parent sequence, we simulate a single round of the somatic hypermutation process, though our estimation procedure ensures that the mutation frequency is approximately equal to that in real data. Additional complexity to incorporate branch lengths could be included, though it did not appear to influence the accuracy of parameter estimation. After simulating a sequence for each parent according to the forward model, we calculated the summary statistics described above, and input them into our neural network, which then output an estimate for the posterior mean of θ. This is the procedure outlined in Figure 3.5, with the goal of minimizing mean-squared error. The neural network was trained on a set of $10,000$ such simulated datasets, and evaluated on a held-out test set of $1,000$ such datasets. In order to avoid the relative values of the summary statistics adversely affecting our estimation, we standardized each summary statistic to have mean zero and unit variance.

The neural network used for training was a simple feedforward neural network [80] with 5 hidden layers and $64, 32, 16, 8$, and 4 nodes per hidden layer, respectively. Training was done with Adam [39] in PyTorch [57] using mean-squared error as the loss. We used mini-batches of size 200 and trained for $1,000$ iterations with step size 10^{-2}.

Our implementation the forward model, as well as all of the code used for inference, is available at `https://github.com/thayerf/shmpy`.

3.7 Results

We assessed how accurately our neural network recovered the mechanistic parameters by testing it on $1,000$ held-out simulated datasets. Results of these predictions can be seen in Figure 3.7. We were able to explain approximately 75% of the prior variance with our chosen summary statistics and neural network described above. We recover the value of the lengthscale, sigma, the forward probability, and BER probability with high accuracy. We struggle to estimate the size of the exo stripping region to the left and right of the excised base with the same level of accuracy. Thus, it is possible that the mean size of the exo-stripping region cannot be accurately recovered solely from observed sequences.

In order to explore the importance of each summary statistic in predicting each parameter, we performed an experiment in which we shuffled the values of each summary statistic in our test set among simulation replicates. After shuffling one of the summary statistics, we then recorded the MSE for each parameter, and compared it to the MSE of the full model. A summary statistic which, when permuted, greatly increases the MSE for a parameter, is important for estimating that parameter. The mean squared error vs. the summary statistic removed can be seen in Figure 3.8. We found that colocalization is the most informative summary statistic for the lengthscale, the base mutation frequency is most informative for the variance, the fraction of C/G mutations which occur at C sites is most informative for the forward probability, and the A/T mutation fraction is most informative for the BER probability. For the exo window sizes, it appears that all of the summary statistics contribute relatively equally. However, a large part of this reduction in mean squared error is due to strong predictive performance in parameters related to the spatial colocalization of AID lesions.

We also wanted to confirm that summary statistics from real mutated sequences fell in a range which could be credibly generated by our forward model. A comparison of simulated and real summary statistics can be seen in Figure 3.9. All but one of the summary statistics lie in the high-density range of the simulated sequences. The notable exception is the distance

from an A/T mutation to the nearest C/G site. Real A/T mutations are farther away from C/G sites on average than can be explained by our forward model. One possible explanation is the faithful repair of the central deaminated cytosine in the MMR pathway, which we do not account for.

We also wanted to make sure that our estimated mechanistic parameters for the real sequences fell in the prior range. A comparison of the prior distribution to the estimated parameter values for real mutated sequences can be seen in Figure 3.10. We can see that our prior contains feasible values for real sequences.

Finally, we were interested in how our model performed on a per-site mutability level. To investigate this, we wished to compare loss on a logistic regression model which estimates the mutation frequency at a given site using only the 5-mer located there, analogous to the classic S5F model [95]. However, it is difficult to calculate the per-site mutation frequency of our mechanistic model because, unlike regression-based approaches, we do not estimate per-site probabilities directly. In order to obtain per-site mutability estimates, we forward simulated $10,000$ sequences each from $1,000$ parent sequences according to our fit mechanistic parameters. We then averaged over these $10,000$ simulated sequences to get a per-site estimate of mutability for each of the $1,000$ parent sequences. We calculated an average per-site log loss by comparing these estimates to vectors of mutation indicators for the true $1,000$ child sequences. The same per-site calculation was conducted for the 5-mer model. A 5-mer model outperforms our mechanistic model with respect to per-site mutability, with a per-site log loss of 0.315 for the 5-mer model compared to 0.345 for the mechanistic model. This suggests that our mechanistic model, which was designed to capture global mutation patterns, is unable to attain the local accuracy of a 5-mer model.

3.7.1 Cross-colocalization

Both to check on our model's performance on a measure that was not used for training and to get insight into processes our model is missing, we looked at a measure of colocalization that takes base identity into account. We refer to this quantity as the *cross-colocalization*:

the cross-colocalization for bases b_1 and b_2 at a distance r is defined as the ratio between the fraction of pairs of mutations containing bases b_1 and b_2 at distance r at the fraction of pairs of locations on the sequence containing bases b_1 and b_2 at distance r (see Equation (3.12)). This quantity is a generalization of the colocalization defined by Spisak et al. [79], and is described in more detail in Section C.1.2. Measures with a similar goal have been described in other papers [41], but they require more information to compute. Specifically, they require per-site estimates of mutation probability, which would be available to us only if we used a parametric model or restricted ourselves to a subset of the sequences.

We perform a comparison of the cross-colocalizations between the real and simulated data in Figure 3.11. As expected, the cross-colocalizations fall off as a function of r in both the real and the simulated sequences. The cross-colocalizations involving C or G are overall more comparable between the real and simulated sequences than the AA/TT/AT cross-colocalizations. The primary difference between the cross-colocalizations involving C or G occurs at distance 1 for C/G pairs: that cross-colocalization is substantially higher in the real data than in the simulated data. This large difference in cross-colocalization at distance 1 is evidence in favor of a specific process not included in our model that only acts on C/G pairs at distance 1. Two such processes that have been suggested in the literature are overlapping hotspots [90] and tandem mutations [70]. This particular result gives more credence to the overlapping hotspots explanation, as the difference in cross-colocalization occurs for only one pair of bases, and the tandem mutation pathway is not hypothesized to be specific to a certain set of bases.

The other main difference between the cross-colocalizations computed on the real and simulated data is that the AA, TT, and AT cross-colocalizations are higher overall and drop off more slowly in the simulated data than in the real data. This might be due partially to our poor estimation of the exo stripping region. Similar to cross-colocalization results for pairs of bases including C or G, these results show that the colocalization pattern for AA/AT/TT pairs can not be explained without resorting to explanations involving overlapping hotspots or extra pathways not included in the model.

3.8 Discussion

The forward simulation and estimation framework we describe is, to our knowledge, the first work to try to fit mechanistic parameters to the consensus model for somatic hypermutation. While previous work has focused on modeling mutability through local sequence context, or isolating mechanistic pathways via knockout experiments, we present a method which estimates relative rates of known mechanistic pathways. Here we have built a simplified forward model and developed means of estimating its mechanistic parameters using a neural-network-based ABC-like framework.

Other work has employed deep learning in ABC. Jiang et al. [34] use deep neural networks to estimate the posterior mean for parameters in a stochastic genetic oscillator, and then use that posterior mean as a summary statistic for traditional ABC. We cannot efficiently estimate the posterior mean due to the high-dimension and sparsely-distributed information in our sequence data. Fearnhead and Prangle [21] generate candidate summary statistics automatically via regression. Fisher et al. [23] attempt to estimate posterior quantiles directly via neural network estimation and forward simulation. The method proposed here is similar to these previous works, but uses the neural network itself to determine the relationship between user-generated summary statistics and the estimand.

There are several elements of real sequences which are recovered faithfully with our estimation framework. The rate at which spatial colocalization decays to 1, governed primarily by the lengthscale in our model, is estimated with high accuracy. More notably, the size of the exo stripping region estimated here is similar to the range discussed in previous experiments [84], though others have proposed significantly larger windows [36]. Crucially, despite using a non-informative prior, our procedure finds the size of the Exo stripping region to be about 35 bases. This aligns with previous work showing that mutations can occur under SHM in the middle of long stretches of A/T bases [84]. We reproduce that feature in this work. However, it is worth noting that the exo parameters were estimated the least accurately of all our estimands. This is likely due to the large number of possible explanations for

a given mutation pattern, as well as the sparse distribution of information about the MMR pathway throughout our simulated sequences.

Some features estimated in the model differ from previous in vitro or animal studies. Both in vitro [60] and in vivo [61] studies have shown that AID deaminates C's on both DNA strands. The in vitro data study suggests about equal AID targeting to each strand, while the in vivo data suggests about 1.5x more deaminations on the coding strand. We estimate a forward strand bias of about 1.22x, meaning that we do not see as dramatic of a strand bias for AID targeting in our estimation framework. It is possible this is due to differences between human and mouse sequence data, since knockout experiments which isolate individual pathways are not possible for human data.

There are several shortcomings of the model proposed here. While the colocalization falls within the central 90% of values at every distance, the distance 2 colocalization is much higher in real sequences than our model can account for. This difference can possibly be attributed to tandem mutations caused by Polζ [70], or by overlapping hotspots [90]. Our cross colocalization results suggest overlapping hotspots are a more likely explanation. We also found that a simpler 5-mer model had lower log loss than our mechanistic model on real data, presumably due to mutation hotspots not being modeled in our framework. Attempts to incorporate mutability into the prelesion probabilities made little difference, likely due to the relatively high marginal probability that a prelesion is thinned. Further attempts should be made to incorporate local sequence context into the pathways downstream of AID targeting, as well as into lesion probability. Doing so will allow for hotspots which are as highly mutable relative to the base mutation frequency as what we see in real sequence data.

Many recent theories and uncertainties about the mechanisms governing SHM are not incorporated into our model. Recent work has suggested that AID can return to an exo-stripped patch, leading to additional SHM [25]. Other work has suggested that the exo-stripped patch can be longer than any sequence we engage with here, potentially several thousand base-pairs long [36]. Overall, one must carefully balance the seemingly limitless complexity of somatic hypermutation mechanisms with the difficulty of high-dimensional

Bayesian inference. While a model that incorporates every minor element of the pathways governing SHM might seem ideal, the sparse information in the mutated sequences, as well as the exponentially growing number of explanations for the mechanistic history of a sequence, renders such a model intractable. In fact, while estimation in the simple proposed model is quite accurate, it is not clear that a model which incorporates several additional pathways, as well as several downstream context-sensitive transition probabilities, would even be identifiable. The complexity of future mechanistic models should be gradually increased to ensure accurate recovery of the relevant pathways.

In order to guide future work, we will now review other approaches that we tried. We initially attempted to predict parameters directly from the hot-encoded sequences, but it was not computationally feasible. Despite the fact that sequences that have undergone SHM have a relatively high number of mutations, information about the mechanistic parameters is still sparsely distributed throughout the entire set of sequences. It is not clear how to effectively combine information from thousands of hot-encoded sequences into a single parameter estimate. Our initial attempt was to use a convolutional neural network to output a low-dimensional vector for each sequence. Following this, a recurrent neural network would combine these outputs into a single parameter estimate. Not only was this computationally expensive, but performance was poor.

Since it is much easier to find the parameters that maximize the complete-data likelihood than to find parameters that maximize the observed-data likelihood, our next estimation attempt was to use an importance sampling EM procedure. We attempted to sample from an approximation of the conditional distribution of the latent variables given the observations and find the values of the parameters that maximized the complete data log likelihood, using importance weights to correct for the fact that we were sampling from an approximation of the conditional distribution of the latent variables given the data and not the exact conditional distribution. This failed because our approximation did not match the truth closely enough, leading to high variance in the weights and often a single latent variable draw per observation having a weight that dominated all the other draws for that observation.

Our original hypothesis in this work was that by leveraging mechanistic knowledge, we would be able to fit more accurate models. This strategy has worked in other settings, including our recent work on V(D)J recombination [69]. However, in this context, this has not resulted in more accurate mutability predictions. For somatic hypermutation it seems possible that the biochemical process is simply too complex to gain statistical efficiency using this approach. The path forward in this setting may be for more sophisticated mechanism-free models as in [79, 81].

On the other hand, this modeling exercise has revealed some features of the data that support current theories about the mechanisms governing SHM. Some parameters governing a forward model of mechanistic pathways can be recovered faithfully. We show that known pathways which induce mutations are sufficient to produce the level of spatial correlation we see in mutation sites *in vivo*. Although we were not able to provide more accurate predictions than mechanism-free models, we still strongly believe in the merit of mathematically formalizing biochemical models and engaging with parameter estimation from real data. As experiments isolating different pathways are not available for humans, attempting to infer parameters jointly from sequence data is the only way to get parameter estimates for biochemical models; even when isolating those pathways, getting parameter estimates is difficult.

Future work into refinement of the proposed forward model should be considered. While this work showed that our neural network estimation procedure can estimate the parameters governing forward simulation of the two major pathways which drive SHM, there are additional layers of complexity which could be added. However, with the current formulation, such additions would also necessitate additional hand-crafted, informative summary statistics to estimate parameters associated with the new pathways. It is not guaranteed that joint accuracy in estimation, or even identifiability, would be maintained were the model made significantly more complex. Our results suggest that the high dimension and sparsely distributed information contained in BCR sequence data may indicate limits on how much we can learn about biochemical mechanisms from sequencing experiments alone.

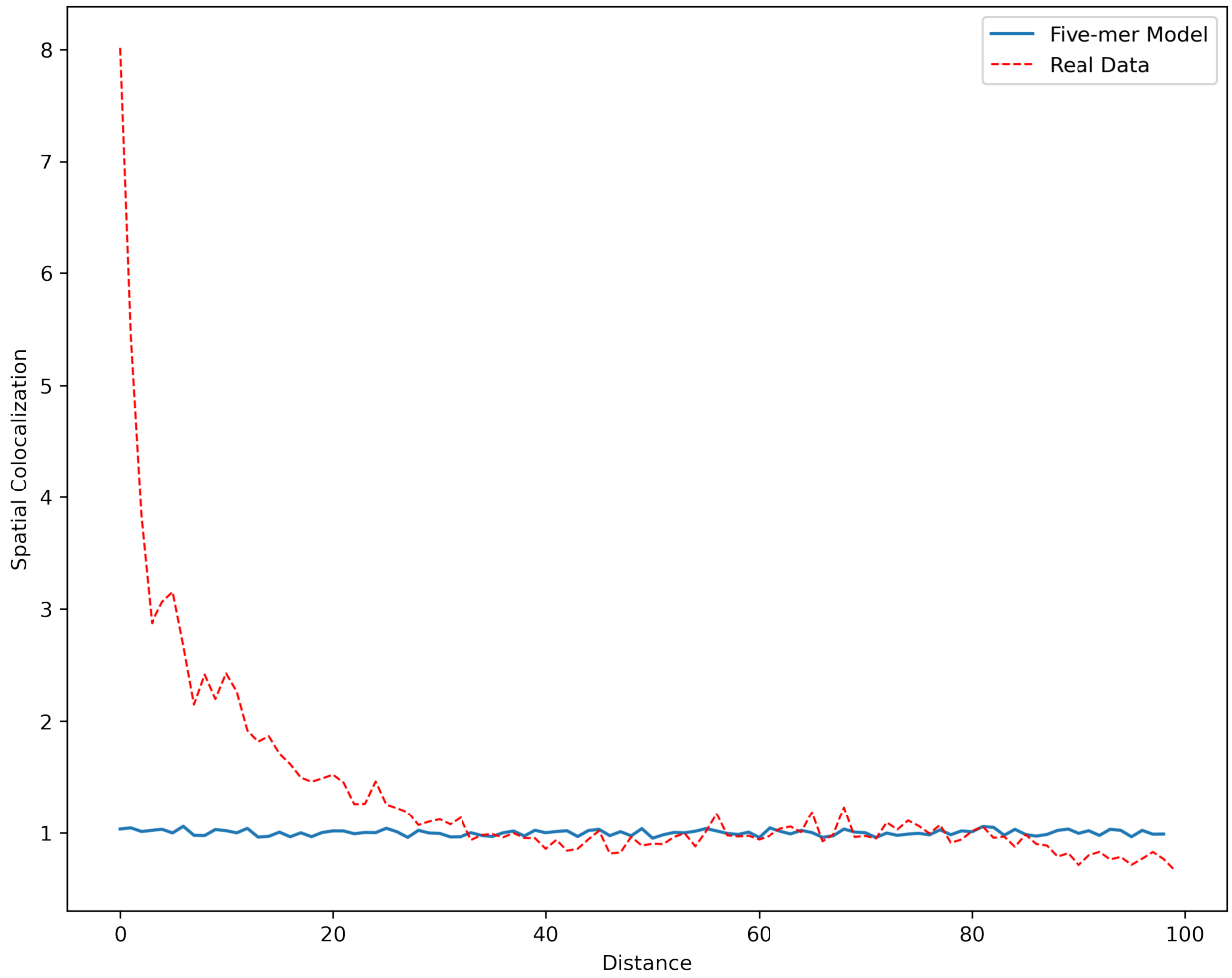

Figure 3.6: True spatial colocalization vs. 5-mer simulated colocalization. Mutations happen much closer together than the local context model accounts for.

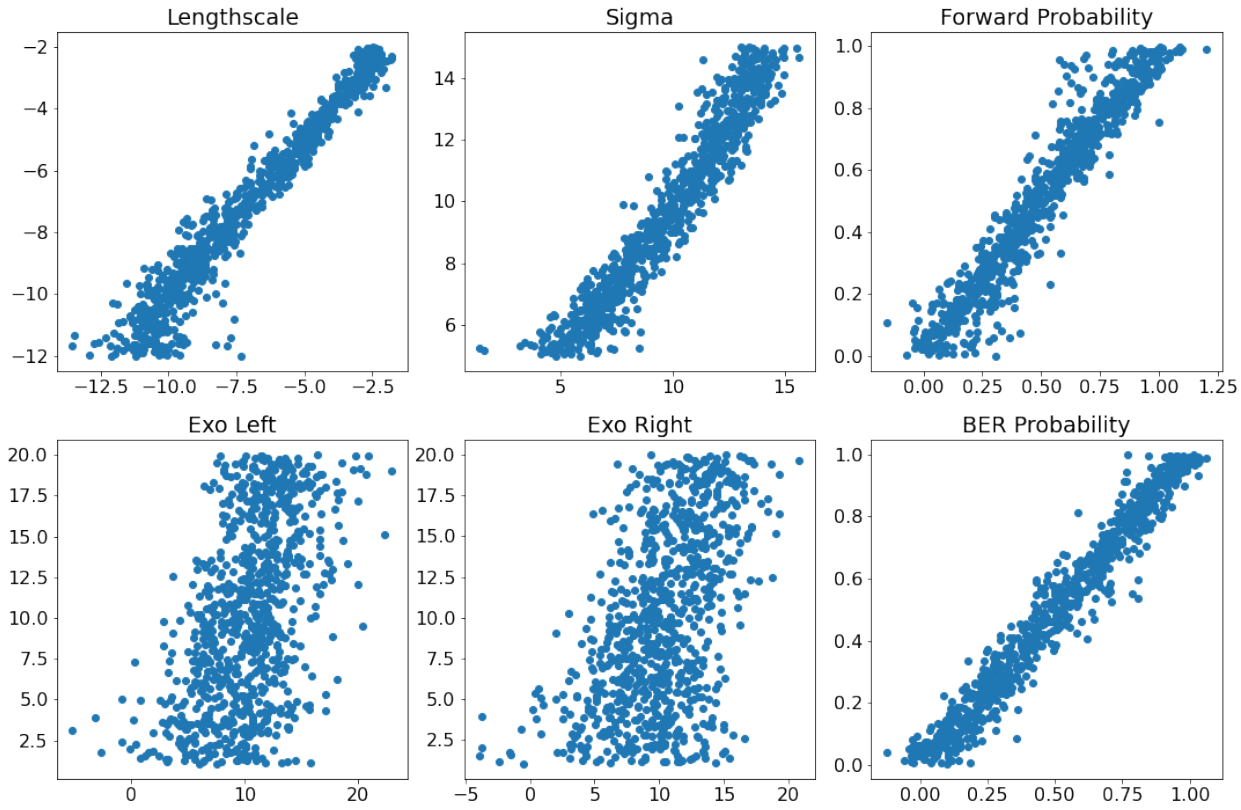

Figure 3.7: True vs. estimated parameters. Despite the inclusion of several summary statistics which ought to inform the exo window size, the approximate posterior variance is quite high. Other parameters are estimated very accurately.

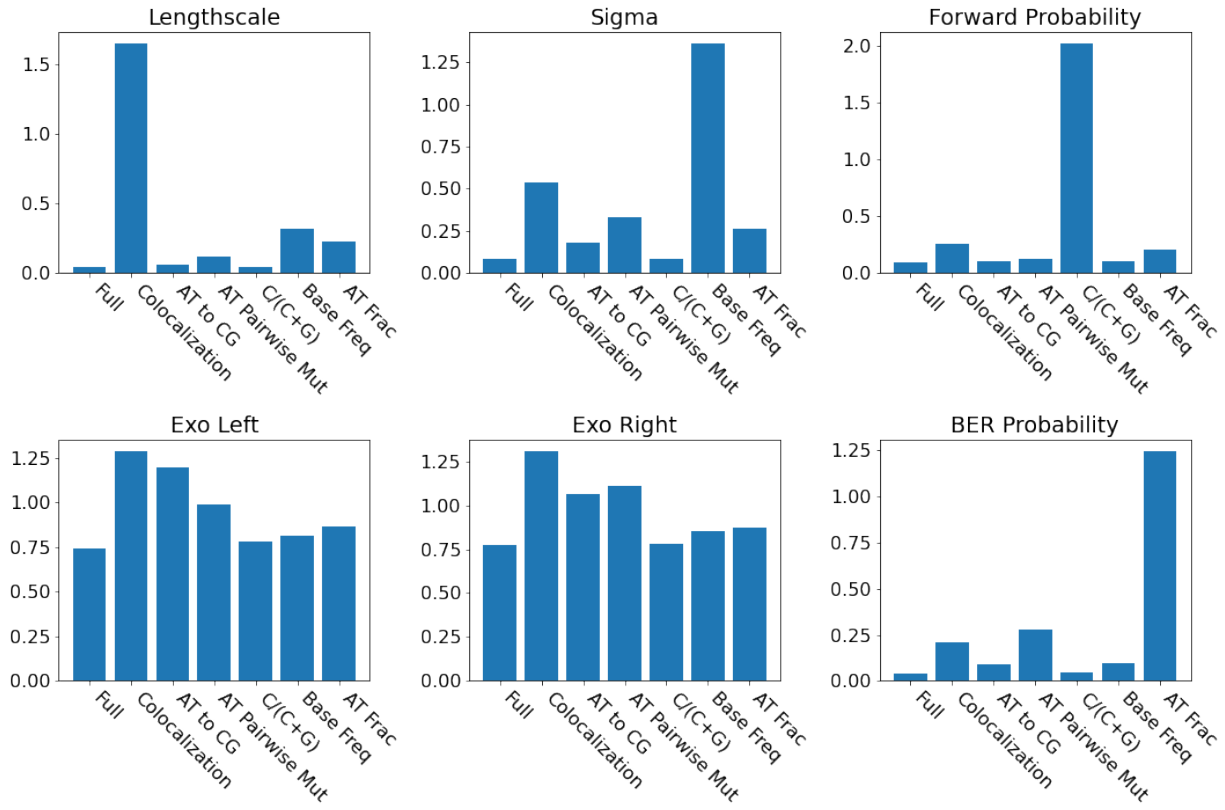

Figure 3.8: Neural network test set mean squared error (MSE) vs. which summary statistic was shuffled, stratified by parameter. A higher MSE relative to the full model implies that a particular summary statistic is informative for that parameter.

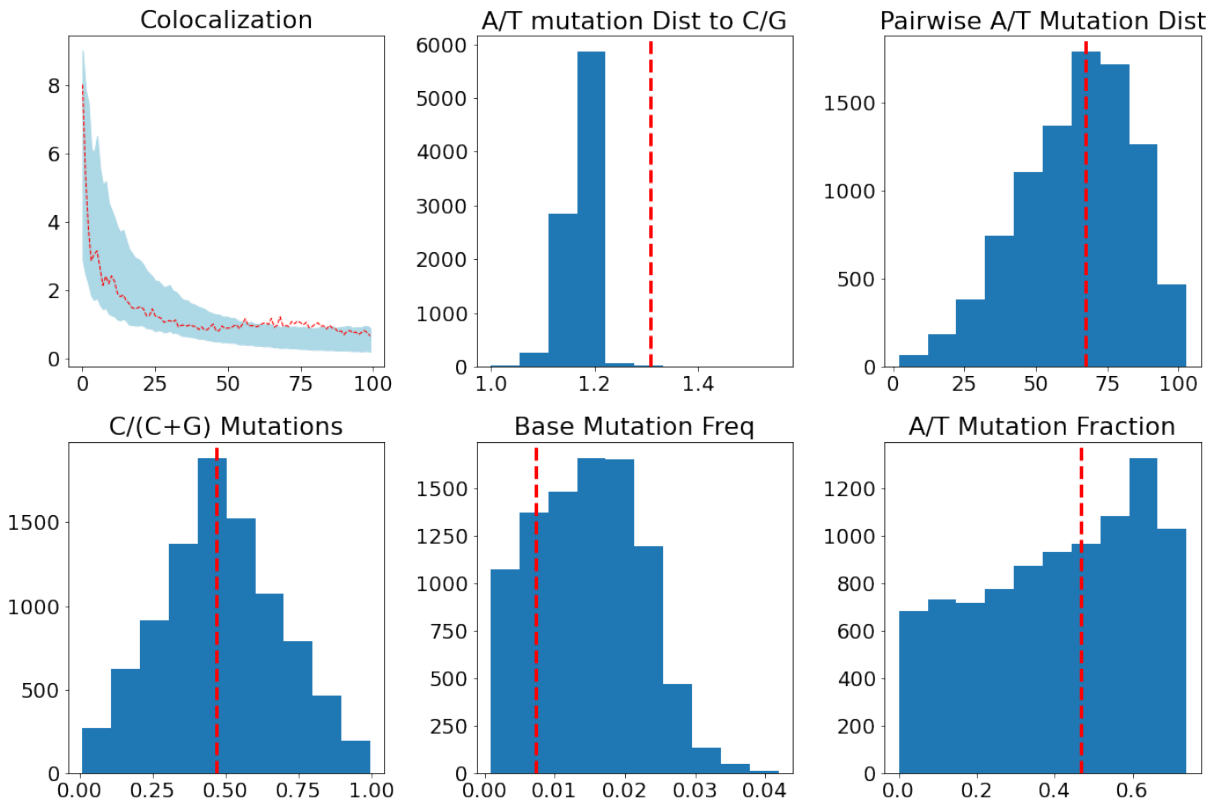

Figure 3.9: Real data summary statistics compared to their simulated data equivalent. The real sequence summary statistics (red dashed line) fall into a range that can be credibly explained by out forward simulation framework (histogram) except for the A/T mutation distance to a C/G site. For the colocalization plot, the shaded region represents the central 90% of colocalization values at each distance.

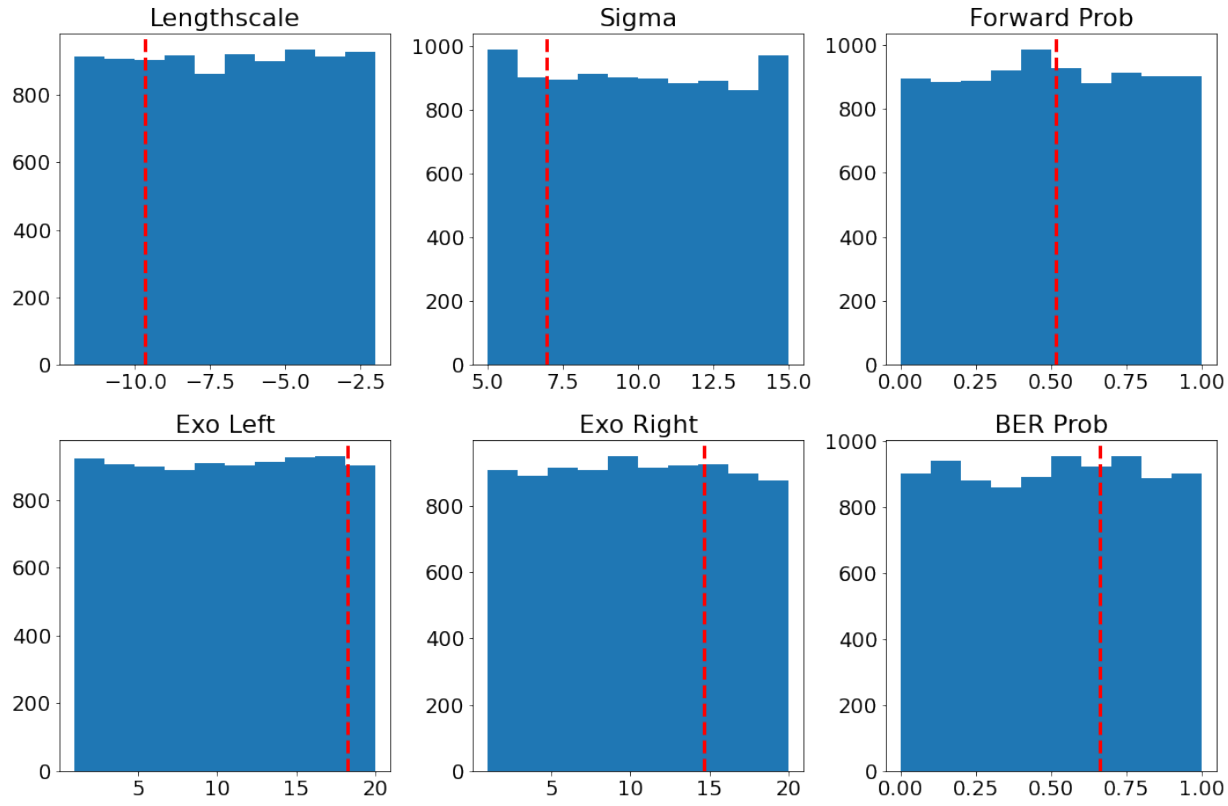

Figure 3.10: Real data estimated parameter values vs. prior range. All of the estimated governing parameters for the real data fall into the domain of our priors, suggesting that our forward model is calibrated to have realistic relative rates of the various pathways.

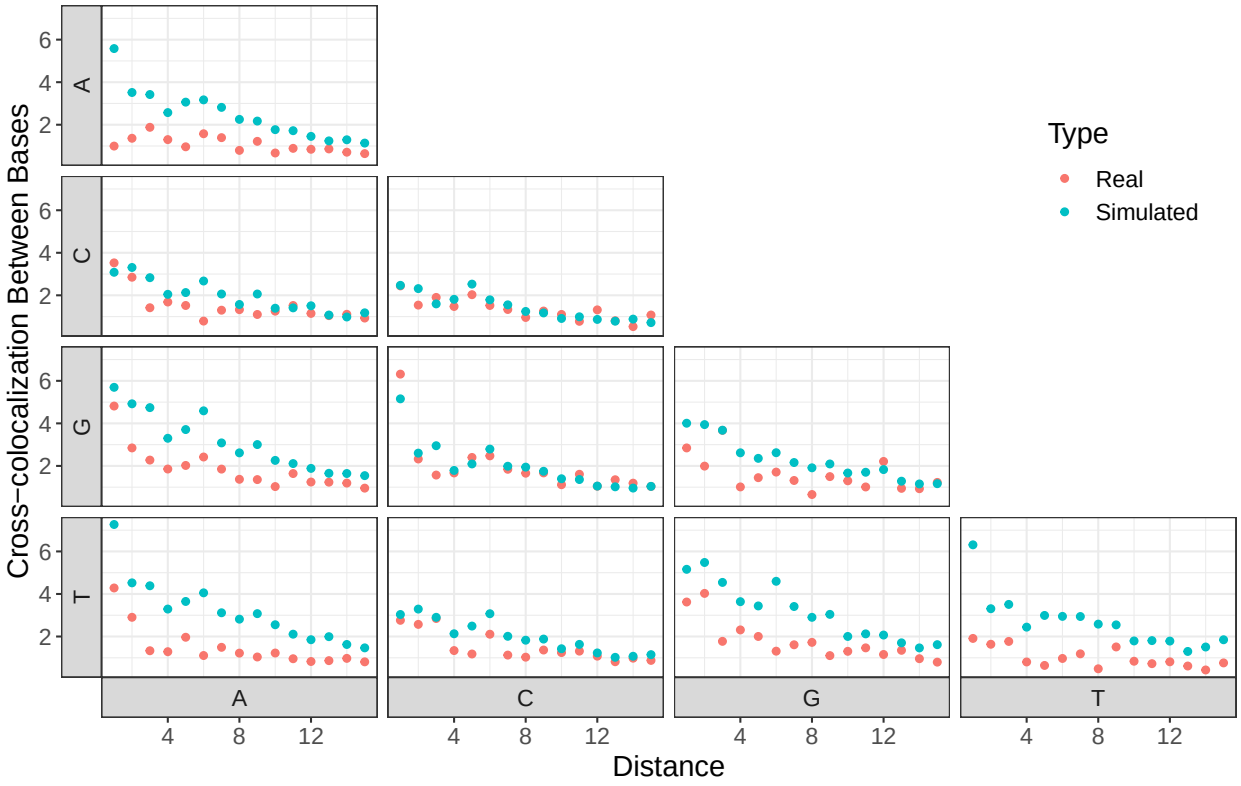

Figure 3.11: Cross-colocalization for data from Spisak et al. [79] and simulated sequences. This quantity is the colocalization between sites with bases as written in the row and column labels.

BIBLIOGRAPHY

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283, 2016.

[2] Linda JS Allen. A primer on stochastic epidemic models: Formulation, numerical simulation, and analysis. *Infectious Disease Modelling*, 2(2):128–142, 2017.

[3] S. Ananthi and P. Dhanalakshmi. Svm and hmm modeling techniques for speech recognition using lpcc and mfcc features. In Suresh Chandra Satapathy, Bhabendra Narayan Biswal, Siba K. Udgata, and J.K. Mandal, editors, *Proceedings of the 3rd International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) 2014*, pages 519–526, Cham, 2015. Springer International Publishing. ISBN 978-3-319-11933-5.

[4] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6): 26–38, 2017. doi: 10.1109/MSP.2017.2743240.

[5] Søren Asmussen and Peter W Glynn. *Stochastic simulation: algorithms and analysis*, volume 57. Springer Science & Business Media, 2007.

[6] Francis Bach. Breaking the curse of dimensionality with convex neural networks. *Journal of Machine Learning Research*, 18(19):1–53, 2017.

[7] Mattia Bonsignori, Edward F Kreider, Daniela Fera, R Ryan Meyerhoff, Todd Bradley, Kevin Wiehe, S Munir Alam, Baptiste Aussedat, William E Walkowicz, Kwan-Ki Hwang, Kevin O Saunders, Ruijun Zhang, Morgan A Gladden, Anthony Monroe, Amit

Kumar, Shi-Mao Xia, Melissa Cooper, Mark K Louder, Krisha McKee, Robert T Bailer, Brendan W Pier, Claudia A Jette, Garnett Kelsoe, Wilton B Williams, Lynn Morris, John Kappes, Kshitij Wagh, Gift Kamanga, Myron S Cohen, Peter T Hraber, David C Montefiori, Ashley Trama, Hua-Xin Liao, Thomas B Kepler, M Anthony Moody, Feng Gao, Samuel J Danishefsky, John R Mascola, George M Shaw, Beatrice H Hahn, Stephen C Harrison, Bette T Korber, and Barton F Haynes. Staged induction of HIV-1 glycan-dependent broadly neutralizing antibodies. *Sci. Transl. Med.*, 9(381), March 2017. ISSN 1946-6234, 1946-6242. doi: 10.1126/scitranslmed.aai7514. URL `http://dx.doi.org/10.1126/scitranslmed.aai7514`.

[8] B S Briney, J R Willis, and J E Crowe, Jr. Location and length distribution of somatic hypermutation-associated DNA insertions and deletions reveals regions of antibody structural plasticity. *Genes Immun.*, 13(7):523–529, October 2012. ISSN 1466-4879, 1476-5470. doi: 10.1038/gene.2012.28. URL `http://dx.doi.org/10.1038/gene.2012.28`.

[9] Bob Carpenter, Andrew Gelman, Matt Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Michael A Brubaker, Jiqiang Guo, Peter Li, Allen Riddell, et al. Stan: A probabilistic programming language. *Journal of Statistical Software*, 20(2):1–37, 2016.

[10] Jeffrey Chan, Valerio Perrone, Jeffrey Spence, Paul Jenkins, Sara Mathieson, and Yun Song. A likelihood-free inference framework for population genetic data using exchangeable neural networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL `https://proceedings.neurips.cc/paper/2018/file/2e9f978b222a956ba6bdf427efbd9ab3-Paper.pdf`.

[11] Linda Chelico, Phuong Pham, Peter Calabrese, and Myron F Goodman. APOBEC3G DNA deaminase acts processively $3' \rightarrow 5'$ on single-stranded DNA. *Nat. Struct. Mol.*

Biol., 13(5):392–399, May 2006. ISSN 1545-9993, 1545-9985. doi: 10.1038/nsmb1086. URL http://www.nature.com/nsmb/journal/v13/n5/abs/nsmb1086.html.

[12] Damian Clancy, Philip D O'Neill, et al. Bayesian estimation of the basic reproduction number in stochastic epidemic models. *Bayesian Analysis*, 3(4):737–757, 2008.

[13] Francis S Collins and Harold Varmus. A new initiative on precision medicine. *New England journal of medicine*, 372(9):793–795, 2015.

[14] Michael Creel. Neural nets for indirect inference. *Econometrics and Statistics*, 2:36–49, 2017.

[15] Ang Cui, Roberto Di Niro, Jason A Vander Heiden, Adrian W Briggs, Kris Adams, Tamara Gilbert, Kevin C O'Connor, Francois Vigneault, Mark J Shlomchik, and Steven H Kleinstein. A model of somatic hypermutation targeting in mice based on High-Throughput ig sequencing data. *J. Immunol.*, 197(9):3566–3574, November 2016. ISSN 0022-1767, 1550-6606. doi: 10.4049/jimmunol.1502263. URL http://dx.doi.org/10.4049/jimmunol.1502263.

[16] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

[17] Amrit Dhar, Kristian Davidsen, Frederick A Matsen IV, and Vladimir N Minin. Predicting B cell receptor substitution profiles using public repertoire data. *arXiv; in press at PLOS Computational Biology*, February 2018. URL http://arxiv.org/abs/1802.06406.

[18] D K Dunn-Walters and J Spencer. Strong intrinsic biases towards mutation and conservation of bases in human IgVH genes during somatic hypermutation prevent statistical analysis of antigen selection. *Immunology*, 95(3):339–345, November 1998. ISSN 0019-2805. URL http://www.ncbi.nlm.nih.gov/pubmed/9824495.

[19] D K Dunn-Walters, A Dogan, L Boursier, C M MacDonald, and J Spencer. Base-specific sequences that bias somatic hypermutation deduced by analysis of out-of-frame human IgVH genes. *J. Immunol.*, 160(5):2360–2364, March 1998. ISSN 0022-1767. URL https://www.ncbi.nlm.nih.gov/pubmed/9498777.

[20] Paul Fearnhead and Dennis Prangle. Constructing summary statistics for approximate bayesian computation: Semi-automatic abc, 2011.

[21] Paul Fearnhead and Dennis Prangle. Constructing summary statistics for approximate bayesian computation: semi-automatic approximate bayesian computation. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 74(3):419–474, 2012. doi: https://doi.org/10.1111/j.1467-9868.2011.01010.x. URL https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-9868.2011.01010.x.

[22] Jean Feng, David A Shaw, Vladimir N Minin, Noah Simon, and Frederick A Matsen, IV. Survival analysis of DNA mutation motifs with penalized proportional hazards. *Ann. Appl. Stat.*, 13(2):1268–1294, June 2019. ISSN 1932-6157, 1941-7330. doi: 10.1214/18-AOAS1233. URL https://projecteuclid.org/euclid.aoas/1560758446.

[23] Thayer Fisher, Alex Luedtke, Marco Carone, and Noah Simon. Deep learning for marginal bayesian posterior inference with recurrent neural networks, 2024. URL http://dx.doi.org/10.5705/ss.202020.0348.

[24] Zoubin Ghahramani. An introduction to hidden markov models and bayesian networks. In *Hidden Markov models: applications in computer vision*, pages 9–41. World Scientific, 2001.

[25] Blerta Green, Antoaneta Belcheva, Rajeev M. Nepal, Bryant Boulianne, and Alberto Martin. The mismatch repair pathway functions normally at a non-AID target in germinal center B cells. *Blood*, 118(11):3013–3018, 09 2011. ISSN 0006-4971. doi: 10.1182/blood-2011-03-345991. URL https://doi.org/10.1182/blood-2011-03-345991.

[26] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, 2018. URL https://arxiv.org/abs/1801.01290.

[27] Uri Hershberg, Mohamed Uduman, Mark J Shlomchik, and Steven H Kleinstein. Improved methods for detecting selection by mutation analysis of Ig V region sequences. *Int. Immunol.*, 20(5):683–694, April 2008. ISSN 0953-8178. doi: 10.1093/intimm/dxn026. URL http://dx.doi.org/10.1093/intimm/dxn026.

[28] Tony W Ho, Eric Pearlman, Donald Lewis, Mirja Hämäläinen, Kathryn Connor, David Michelson, Ying Zhang, Christopher Assaid, Lyn Harper Mozley, Nancy Strickler, et al. Efficacy and tolerability of rizatriptan in pediatric migraineurs: results from a randomized, double-blind, placebo-controlled trial using a novel adaptive enrichment design. *Cephalalgia*, 32(10):750–765, 2012.

[29] Sepp Hochreiter, A Steven Younger, and Peter R Conwell. Learning to learn using gradient descent. In *International Conference on Artificial Neural Networks*, pages 87–94. Springer, 2001.

[30] N Holford, S C Ma, and B A Ploeger. Clinical trial simulation: A review. *Clinical Pharmacology & Therapeutics*, 88(2):166–182, 2010. doi: https://doi.org/10.1038/clpt.2010.114. URL https://ascpt.onlinelibrary.wiley.com/doi/abs/10.1038/clpt.2010.114.

[31] Joyce K Hwang, Chong Wang, Zhou Du, Robin M Meyers, Thomas B Kepler, Donna Neuberg, Peter D Kwong, John R Mascola, M Gordon Joyce, Mattia Bonsignori, Barton F Haynes, Leng-Siew Yeap, and Frederick W Alt. Sequence intrinsic somatic mutation mechanisms contribute to affinity maturation of VRC01-class HIV-1 broadly neutralizing antibodies. *Proc. Natl. Acad. Sci. U. S. A.*, July 2017. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.1709203114. URL http://dx.doi.org/10.1073/pnas.1709203114.

[32] Jacob G Jansen, Petra Langerak, Anastasia Tsaalbi-Shtylik, Paul van den Berk, Heinz Jacobs, and Niels de Wind. Strand-biased defect in C/G transversions in hypermutating immunoglobulin genes in rev1-deficient mice. *J. Exp. Med.*, 203(2):319–323, February 2006.

[33] Joseph Jardine, Jean-Philippe Julien, Sergey Menis, Takayuki Ota, Oleksandr Kalyuzhniy, Andrew McGuire, Devin Sok, Po-Ssu Huang, Skye MacPherson, Meaghan Jones, Travis Nieusma, John Mathison, David Baker, Andrew B Ward, Dennis R Burton, Leonidas Stamatatos, David Nemazee, Ian A Wilson, and William R Schief. Rational HIV immunogen design to target specific germline B cell receptors. *Science*, 340(6133): 711–716, May 2013. ISSN 0036-8075, 1095-9203. doi: 10.1126/science.1234150. URL `http://dx.doi.org/10.1126/science.1234150`.

[34] Bai Jiang, Tung-Yu Wu, Charles Zheng, and Wing H. Wong. Learning summary statistic for approximate bayesian computation via deep neural network. *Statistica Sinica*, 27 (4):1595–1618, 2017. ISSN 10170405, 19968507. URL `http://www.jstor.org/stable/26384090`.

[35] Bai Jiang, Tung-yu Wu, Charles Zheng, and Wing H Wong. Learning summary statistic for approximate bayesian computation via deep neural network. *Statistica Sinica*, pages 1595–1618, 2017.

[36] Farid A. Kadyrov, Leonid Dzantiev, Nicoleta Constantin, and Paul Modrich. Endonucleolytic function of mutla in human mismatch repair. *Cell*, 126(2):297–308, 2006. ISSN 0092-8674. doi: https://doi.org/10.1016/j.cell.2006.05.039. URL `https://www.sciencedirect.com/science/article/pii/S0092867406008129`.

[37] Motoo Kimura. A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. *Journal of molecular evolution*, 16:111–120, 1980.

[38] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[39] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[40] Roger Koenker and Kevin F Hallock. Quantile regression. *Journal of economic perspectives*, 15(4):143–156, 2001.

[41] Artem Krantsevich, Catherine Tang, and Thomas MacCarthy. Correlations in somatic hypermutation between sites in ighv genes can be explained by interactions between aid and/or polη hotspots. *Frontiers in immunology*, 11:618409, 2021.

[42] Peter H L Krijger, Petra Langerak, Paul C M van den Berk, and Heinz Jacobs. Dependence of nucleotide substitutions on ung2, msh2, and PCNA-Ub during somatic hypermutation. *J. Exp. Med.*, 206(12):2603–2611, November 2009.

[43] Cosimo Lupo, Natanael Spisak, Aleksandra M Walczak, and Thierry Mora. Learning the statistics and landscape of somatic mutation-induced insertions and deletions in antibodies. *PLoS Comput. Biol.*, 18(6):e1010167, June 2022. ISSN 1553-734X, 1553-7358. doi: 10.1371/journal.pcbi.1010167. URL `http://dx.doi.org/10.1371/journal.pcbi.1010167`.

[44] Junsheng Ma, Brian P. Hobbs, and Francesco C. Stingo. Statistical methods for establishing personalized treatment rules in oncology. *BioMed Research International*, 2015, 2015. ISSN 2314-6133. doi: 10.1155/2015/670691. Publisher Copyright: Copyright © 2015 Junsheng Ma et al.

[45] Chi H Mak, Phuong Pham, Samir A Afif, and Myron F Goodman. A mathematical model for scanning and catalysis on single-stranded DNA, illustrated with activation-induced deoxycytidine deaminase. *J. Biol. Chem.*, 288(41):29786–29795, October 2013.

ISSN 0021-9258, 1083-351X. doi: 10.1074/jbc.M113.506550. URL http://dx.doi.org/10.1074/jbc.M113.506550.

[46] Paul Marjoram, John Molitor, Vincent Plagnol, and Simon Tavaré. Markov chain monte carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 100(26): 15324–15328, 2003.

[47] Connor O McCoy, Trevor Bedford, Vladimir N Minin, Philip Bradley, Harlan Robins, and Frederick A Matsen IV. Quantifying evolutionary constraints on B-cell affinity maturation. *Philos. Trans. R. Soc. Lond. B Biol. Sci.*, 370(1676), September 2015. ISSN 0962-8436, 1471-2970. doi: 10.1098/rstb.2014.0244. URL http://dx.doi.org/10.1098/rstb.2014.0244.

[48] Cyrus Mehta, Ping Gao, Deepak L Bhatt, Robert A Harrington, Simona Skerjanec, and James H Ware. Optimizing trial design: sequential, adaptive, and enrichment strategies. *Circulation*, 119(4):597–605, 2009.

[49] S P Methot and J M Di Noia. Chapter two - molecular mechanisms of somatic hypermutation and class switch recombination. In Frederick W. Alt, editor, *Advances in Immunology*, volume 133, pages 37–87. Academic Press, 2017. URL http://www.sciencedirect.com/science/article/pii/S0065277616300530.

[50] Jeffrey W Miller and Matthew T Harrison. Mixture models with a prior on the number of components. *Journal of the American Statistical Association*, 113(521):340–356, 2018.

[51] Jesper Moller, Anne Randi Syversveen, and Rasmus Plenge Waagepetersen. Log gaussian cox processes. *Scand J Stat*, 25(3):451–482, September 1998.

[52] Angelia Nedic and Dimitri P Bertsekas. Incremental subgradient methods for nondifferentiable optimization. *SIAM Journal on Optimization*, 12(1):109–138, 2001.

[53] Michael S Neuberger, Javier M Di Noia, Rupert C L Beale, Gareth T Williams, Zizhen Yang, and Cristina Rada. Somatic hypermutation at A:T pairs: polymerase error versus

dUTP incorporation. *Nat. Rev. Immunol.*, 5(2):171–178, 2005. ISSN 1474-1733. URL http://www.nature.com/nri/journal/v5/n2/abs/nri1553.html.

[54] Lam-Tung Nguyen, Heiko A. Schmidt, Arndt von Haeseler, and Bui Quang Minh. IQ-TREE: A Fast and Effective Stochastic Algorithm for Estimating Maximum-Likelihood Phylogenies. *Molecular Biology and Evolution*, 32(1):268–274, 11 2014. ISSN 0737-4038. doi: 10.1093/molbev/msu300. URL https://doi.org/10.1093/molbev/msu300.

[55] Thomas Ondra, Sebastian Jobjörnsson, Robert A Beckman, Carl-Fredrik Burman, Franz König, Nigel Stallard, and Martin Posch. Optimized adaptive enrichment designs. *Statistical Methods in Medical Research*, 28(7):2096–2111, 2019. doi: 10.1177/0962280217747312. URL https://doi.org/10.1177/0962280217747312. PMID: 29254436.

[56] Nickolas Papadopoulos, Kenneth W Kinzler, and Bert Vogelstein. The role of companion diagnostics in the development and use of mutation-targeted cancer therapies. *Nature biotechnology*, 24(8):985–995, 2006.

[57] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[58] S. Patel and R. Kurzrock. Pd-l1 expression as a predictive biomarker in cancer immunotherapy. *Molecular Cancer Therapeutics*, 14:847 – 856, 2015.

[59] Phuong Pham, Ronda Bransteitter, John Petruska, and Myron F Goodman. Processive AID-catalysed cytosine deamination on single-stranded DNA simulates somatic hypermutation. *Nature*, 424(6944):103–107, July 2003. ISSN 0028-0836, 1476-4687. doi: 10.1038/nature01760. URL http://dx.doi.org/10.1038/nature01760.

[60] Phuong Pham, Sohail Malik, Chiho Mak, Peter C Calabrese, Robert G Roeder, and Myron F Goodman. AID–RNA polymerase II transcription-dependent deamination of IgV DNA. *Nucleic Acids Research*, 47(20):10815–10829, 09 2019. ISSN 0305-1048. doi: 10.1093/nar/gkz821. URL https://doi.org/10.1093/nar/gkz821.

[61] Bas Pilzecker and Heinz Jacobs. Mutating for good: Dna damage responses during somatic hypermutation. *Frontiers in Immunology*, 10:438, 2019. ISSN 1664-3224. doi: 10.3389/fimmu.2019.00438. URL https://www.frontiersin.org/article/10.3389/fimmu.2019.00438.

[62] Mark J. Pletcher and Michael Pignone. Evaluating the clinical utility of a biomarker. *Circulation*, 123(10):1116–1124, 2011. doi: 10.1161/circulationaha.110.943860.

[63] Dennis Prangle. Summary statistics in approximate bayesian computation, 2015.

[64] Igor B Rogozin and Marilyn Diaz. Cutting edge: DGYW/WRCH is a better predictor of mutability at G:C bases in ig hypermutation than the widely accepted RGYW/WRCY motif and probably reflects a two-step activation-induced cytidine deaminase-triggered process. *J. Immunol.*, 172(6):3382–3384, March 2004. ISSN 0022-1767. URL https://www.ncbi.nlm.nih.gov/pubmed/15004135.

[65] Igor B Rogozin, Youri I Pavlov, Katarzyna Bebenek, Toshiro Matsuda, and Thomas A Kunkel. Somatic mutation hotspots correlate with DNA polymerase η error spectrum. *Nat. Immunol.*, 2(6):530–536, June 2001. ISSN 1529-2908. doi: 10.1038/88732. URL http://dx.doi.org/10.1038/88732.

[66] Michael Rosenblum, Tianchen Qian, Yu Du, Huitong Qiu, and Aaron Fisher. Multiple testing procedures for adaptive enrichment designs: combining group sequential and reallocation approaches. *Biostatistics*, 17(4):650–662, 03 2016. ISSN 1465-4644. doi: 10.1093/biostatistics/kxw014. URL https://doi.org/10.1093/biostatistics/kxw014.

[67] Sebastian Ruder. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017.

[68] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[69] Magdalena L Russell, Noah Simon, Philip Bradley, and Frederick A Matsen. Statistical inference reveals the role of length, breathing, and nucleotide identity in V(D)J nucleotide trimming. *bioRxiv*, page 2022.12.08.519635, December 2022. doi: 10.1101/2022.12.08.519635. URL https://www.biorxiv.org/content/10.1101/2022.12.08.519635v1.

[70] Huseyin Saribasak, Robert W Maul, Zheng Cao, William W Yang, Dominik Schenten, Sven Kracker, and Patricia J Gearhart. Dna polymerase ζ generates tandem mutations in immunoglobulin variable regions. *Journal of Experimental Medicine*, 209(6):1075–1081, 2012.

[71] Nicholas J. Schork. Personalized medicine: Time for one-person trials. *Nature*, 520: 609–611, 2015.

[72] Sara Sheehan and Yun S Song. Deep learning for population genetic inference. *PLOS Comput. Biol.*, 12(3):e1004845, March 2016. ISSN 1553-734X, 1553-7358. doi: 10.1371/journal.pcbi.1004845. URL http://journals.plos.org/ploscompbiol/article/file?id=10.1371/journal.pcbi.1004845&type=printable.

[73] Zizhang Sheng, Chaim A Schramm, Mark Connors, Lynn Morris, John R Mascola, Peter D Kwong, and Lawrence Shapiro. Effects of darwinian selection and mutability on rate of broadly neutralizing antibody evolution during HIV-1 infection. *PLOS Comput. Biol.*, 12(5):e1004940, May 2016. ISSN 1553-734X, 1553-7358. doi: 10.1371/journal.pcbi.1004940. URL http://dx.doi.org/10.1371/journal.pcbi.1004940.

[74] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018. doi: 10.1126/science.aar6404. URL https://www.science.org/doi/abs/10.1126/science.aar6404.

[75] Noah Simon and Richard Simon. Adaptive enrichment designs for clinical trials. *Biostatistics*, 14(4):613–625, 03 2013. ISSN 1465-4644. doi: 10.1093/biostatistics/kxt010. URL https://doi.org/10.1093/biostatistics/kxt010.

[76] Richard Simon and Noah Robin Simon. Using randomization tests to preserve type i error with response adaptive and covariate adaptive randomization. *Statistics and Probability Letters*, 81(7):767–772, 2011. ISSN 0167-7152. doi: https://doi.org/10.1016/j.spl.2010.12.018. URL https://www.sciencedirect.com/science/article/pii/S0167715210003688. Statistics in Biological and Medical Sciences.

[77] Leslie N. Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates, 2017.

[78] J Spencer, M Dunn, and D K Dunn-Walters. Characteristics of sequences around individual nucleotide substitutions in IgVH genes suggest different GC and AT mutators. *J. Immunol.*, 162(11):6596–6601, June 1999. ISSN 0022-1767. URL https://www.ncbi.nlm.nih.gov/pubmed/10352276.

[79] Natanael Spisak, Aleksandra M Walczak, and Thierry Mora. Learning the heterogeneous hypermutation landscape of immunoglobulins from high-throughput repertoire data. *Nucleic Acids Research*, 48(19):10702–10712, 10 2020. ISSN 0305-1048. doi: 10.1093/nar/gkaa825. URL https://doi.org/10.1093/nar/gkaa825.

[80] Daniel Svozil, Vladimir Kvasnicka, and Jiri Pospichal. Introduction to multi-layer feed-

forward neural networks. *Chemometrics and intelligent laboratory systems*, 39(1):43–62, 1997.

[81] Catherine Tang, Artem Krantsevich, and Thomas MacCarthy. Deep learning model of somatic hypermutation reveals importance of sequence context beyond hotspot targeting. *iScience*, 25(1):103668, January 2022. ISSN 2589-0042. doi: 10.1016/j.isci.2021.103668. URL http://dx.doi.org/10.1016/j.isci.2021.103668.

[82] Peter F. Thall. Bayesian utility-based designs for subgroup-specific treatment comparison and early-phase dose optimization in oncology clinical trials. *JCO Precision Oncology*, 3:1–7, 2019. doi: 10.1200/PO.18.00379. URL https://doi.org/10.1200/PO.18.00379.

[83] Mohamed Uduman, Gur Yaari, Uri Hershberg, Jacob A Stern, Mark J Shlomchik, and Steven H Kleinstein. Detecting selection in immunoglobulin sequences. *Nucleic Acids Res.*, 39(Web Server issue):W499–504, June 2011. ISSN 0305-1048. doi: 10.1093/nar/gkr413. URL http://dx.doi.org/10.1093/nar/gkr413.

[84] Shyam Unniraman and David G. Schatz. Strand-biased spreading of mutations during somatic hypermutation. *Science*, 317(5842):1227–1230, 2007. doi: 10.1126/science.1145065. URL https://www.science.org/doi/abs/10.1126/science.1145065.

[85] M. Unser, A. Aldroubi, and M. Eden. B-spline signal processing. i. theory. *IEEE Transactions on Signal Processing*, 41(2):821–833, 1993. doi: 10.1109/78.193220.

[86] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[87] Marcos C Vieira, Daniel Zinder, and Sarah Cobey. Selection and neutral mutations drive pervasive mutability losses in Long-Lived Anti-HIV B-Cell lineages. *Mol. Biol. Evol.*, 35(5):1135–1146, May 2018. ISSN 0737-4038, 1537-1719. doi: 10.1093/molbev/msy024. URL http://dx.doi.org/10.1093/molbev/msy024.

[88] Meng Wang, Cristina Rada, and Michael S Neuberger. Altering the spectrum of immunoglobulin V gene somatic hypermutation by modifying the active site of AID. *J. Exp. Med.*, 207(1):141–153, January 2010. ISSN 0022-1007, 1540-9538. doi: 10.1084/jem.20092238. URL http://dx.doi.org/10.1084/jem.20092238.

[89] Yanyan Wang, Senxin Zhang, Xinrui Yang, Joyce K Hwang, Chuanzong Zhan, Chaoyang Lian, Chong Wang, Tuantuan Gui, Binbin Wang, Xia Xie, Pengfei Dai, Lu Zhang, Ying Tian, Huizhi Zhang, Chong Han, Yanni Cai, Qian Hao, Xiaofei Ye, Xiaojing Liu, Jiaquan Liu, Zhiwei Cao, Shaohui Huang, Jie Song, Qiang Pan-Hammarström, Yaofeng Zhao, Frederick W Alt, Xiaoqi Zheng, Lin-Tai Da, Leng-Siew Yeap, and Fei-Long Meng. Mesoscale DNA feature in antibody-coding sequence facilitates somatic hypermutation. *Cell*, April 2023. ISSN 0092-8674, 1097-4172. doi: 10.1016/j.cell.2023.03.030. URL http://dx.doi.org/10.1016/j.cell.2023.03.030.

[90] Lirong Wei, Richard Chahwan, Shanzhi Wang, Xiaohua Wang, Phuong T Pham, Myron F Goodman, Aviv Bergman, Matthew D Scharff, and Thomas MacCarthy. Overlapping hotspots in CDRs are critical sites for V region diversification. *Proc. Natl. Acad. Sci. U. S. A.*, 112(7):E728–37, February 2015. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.1500788112. URL http://dx.doi.org/10.1073/pnas.1500788112.

[91] Kevin Wiehe, Todd Bradley, R Ryan Meyerhoff, Connor Hart, Wilton B Williams, David Easterhoff, William J Faison, Thomas B Kepler, Kevin O Saunders, S Munir Alam, Mattia Bonsignori, and Barton F Haynes. Functional relevance of improbable antibody mutations for HIV broadly neutralizing antibody development. *Cell Host Microbe*, 0(0), May 2018. ISSN 1931-3128. doi: 10.1016/j.chom.2018.04.018. URL http://www.cell.com/article/S1931312818302191/abstract.

[92] Teresa M Wilson, Alexandra Vaisman, Stella A Martomo, Patsa Sullivan, Li Lan, Fumio Hanaoka, Akira Yasui, Roger Woodgate, and Patricia J Gearhart. MSH2–MSH6 stimulates DNA polymerase η, suggesting a role for A:T mutations in anti-

body genes. *J. Exp. Med.*, 201(4):637–645, 2005. ISSN 0022-1007. URL http://jem.rupress.org/content/201/4/637.abstract.

[93] Wing Wong, Bai Jiang, Tung-yu Wu, and Charles Zheng. Learning summary statistic for approximate bayesian computation via deep neural network. *Statistica Sinica*, 2018. ISSN 1017-0405. doi: 10.5705/ss.202015.0340. URL http://dx.doi.org/10.5705/ss.202015.0340.

[94] Gur Yaari, Mohamed Uduman, and Steven H Kleinstein. Quantifying selection in high-throughput immunoglobulin sequencing data sets. *Nucleic Acids Res.*, 40(17):e134, May 2012. ISSN 0305-1048. doi: 10.1093/nar/gks457. URL http://dx.doi.org/10.1093/nar/gks457.

[95] Gur Yaari, Jason A Vander Heiden, Mohamed Uduman, Daniel Gadala-Maria, Namita Gupta, Joel N H Stern, Kevin C O'Connor, David A Hafler, Uri Laserson, Francois Vigneault, and Steven H Kleinstein. Models of somatic hypermutation targeting and substitution based on synonymous mutations from high-throughput immunoglobulin sequencing data. *Front. Immunol.*, 4:358, November 2013. ISSN 1664-3224. doi: 10.3389/fimmu.2013.00358. URL http://dx.doi.org/10.3389/fimmu.2013.00358.

[96] Gur Yaari, Jennifer I C Benichou, Jason A Vander Heiden, Steven H Kleinstein, and Yoram Louzoun. The mutation patterns in B-cell immunoglobulin receptors reflect the influence of selection acting at multiple time-scales. *Philos. Trans. R. Soc. Lond. B Biol. Sci.*, 370(1676), September 2015. ISSN 0962-8436, 1471-2970. doi: 10.1098/rstb.2014.0242. URL http://dx.doi.org/10.1098/rstb.2014.0242.

[97] Leng-Siew Yeap, Joyce K Hwang, Zhou Du, Robin M Meyers, Fei-Long Meng, Agne Jakubauskaite, Mengyuan Liu, Vinidhra Mani, Donna Neuberg, Thomas B Kepler, Jing H Wang, and Frederick W Alt. Sequence-Intrinsic mechanisms that target AID mutational outcomes on antibody genes. *Cell*, 163(5):1124–1137, November 2015. ISSN

0092-8674, 1097-4172. doi: 10.1016/j.cell.2015.10.042. URL http://dx.doi.org/10.1016/j.cell.2015.10.042.

[98] Marija Zivojnovic, Frédéric Delbos, Giulia Girelli Zubani, Amélie Julé, Alexandre Alcais, Jean-Claude Weill, Claude-Agnès Reynaud, and Sébastien Storck. Somatic hypermutation at A/T-rich oligonucleotide substrates shows different strand polarities in ung-deficient or -proficient backgrounds. *Mol. Cell. Biol.*, 34(12):2176–2187, June 2014. ISSN 0270-7306, 1098-5549. doi: 10.1128/MCB.01452-13. URL http://dx.doi.org/10.1128/MCB.01452-13.

Appendix A

MARGINAL BAYESIAN POSTERIOR INFERENCE USING RECURRENT NEURAL NETWORKS WITH APPLICATION TO SEQUENTIAL MODELS

A.1 Feedforward Comparison

As an illustrative example of how an RNN outperforms a Feedforward network in posterior quantile estimation, we performed simulations which make a direct comparison. In Figure A.1, we show that a feedforward network with an identical internal structure (same number of hidden nodes per layer and number of layers) is outperformed by a recurrent counterpart in estimating a 90% credible interval for the maximum component of a mixture of finite mixtures. Furthermore, the majority of the cases where our method is best-suited involve series data where the likelihood is not easily calculable, making the RNN an even more preferable option.

A.2 Rate of Convergence

According to the Bernstein-von Mises theorem, any posterior quantile should concentrate around the true parameter at a rate of $O(1/\sqrt{n})$. For the Gaussian conjugate prior simulation setting, we trained several networks with varying sample sizes to predict the 0.05 quantile. Averaging the distance from these estimates to the true values over a large test set give us the approximate convergence rate. In Figure A.2, we see a log-log plot of the 0.05 quantile distance from the true value vs. sample size. The slope of the imposed line of best fit is -0.48. Therefore, we see that the 0.05 quantile concentrates around the true parameter at approximately a rate of $O(1/\sqrt{n})$, though we admit this relationship is not perfectly linear. While ideally we would also show this convergence in more complicated simulation

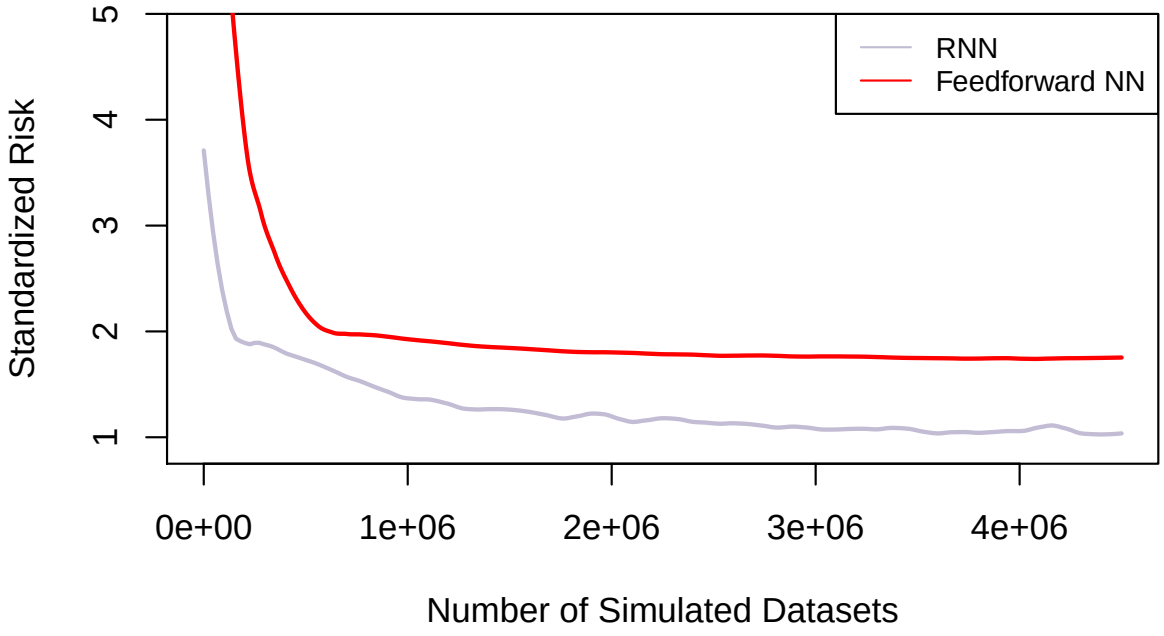

Figure A.1: Standardized risk of a feedforward neural network vs. a recurrent neural network. Targets of estimation are the 0.05 and 0.95 posterior quantiles of the maximum component in a mixture of finite mixtures with a prior on the number of components. Both networks have 3 hidden layers and 10 nodes per hidden layer.

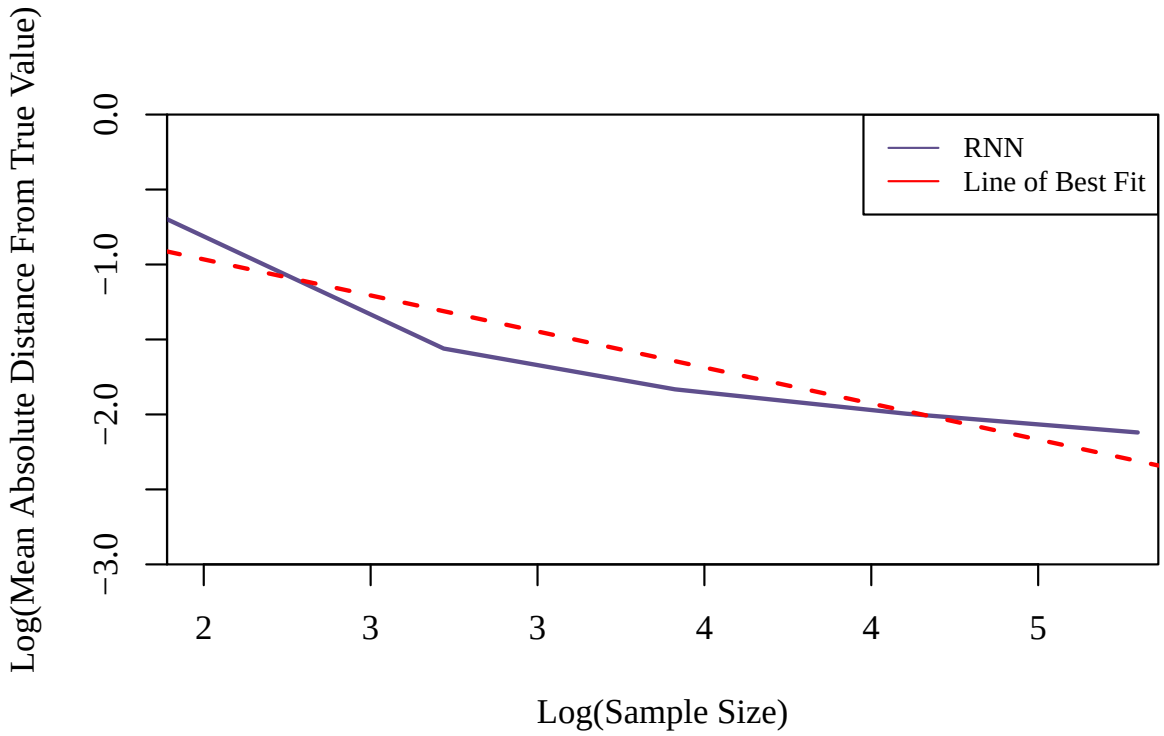

Figure A.2: Log(Mean absolute difference) between estimated 0.05 quantile and true value. The line of best fit is superimposed.

settings, doing so requires training a separate neural network for each value of n, which is computationally expensive for more complex examples. Note also that as n increases, the optimal architecture and training regime changes. However, all of the networks use to produce this convergence plot were identical in their architecture and training regimen. This is one reason why the convergence rate is not exact.

A.3 Distribution Function Analysis

In the continuous estimation setting, where the quantile to be estimated is an imput to the RNN, it is possible to obtain an estimate for the entire distribution function by inputting a grid of quantiles. In the Gaussian conjugate prior scenario, it is also possible to compare this estimated distribution function with the true posterior distribution, since it has closed-form. One example of this can be seen in Figure A.3. While the distribution function is easier to estimate than the density function, one could imagine binning predicted quantile values to achieve a density estimate.

A.4 Conditional Coverage Analysis

Risk is not an ideal measure of performance, but it is the best option in some cases. Other measures worth considering, such as marginal coverage and interval width, are reported in Table 2.1. However, conditional coverage is still of interest. In the Gaussian conjugate prior setting, where we have direct access to the posterior distribution, we can stratify by the sufficient statistic, the sample mean, to obtain a measure of conditional coverage, seen in Figure A.4. We see that, apart from extreme values of the sufficient statistic, we obtain approximately correct conditional coverage as well.

A.5 Permutation Invariance

In the first two simulation settings presented, the posterior distribution of our parameter of interest is invariant to permutations of the input data. In the first of these examples, the Gaussian conjugate prior scenario, we analyzed the performance of our model by averaging predictions from permutations of the input data. Two examples of this can be seen in Figure A.5.

Another potential solution to exchangeability is sorting the input data. When we tested coverage in Table 2.1, we sorted the input data for the Mixture of Finite Mixture exam-

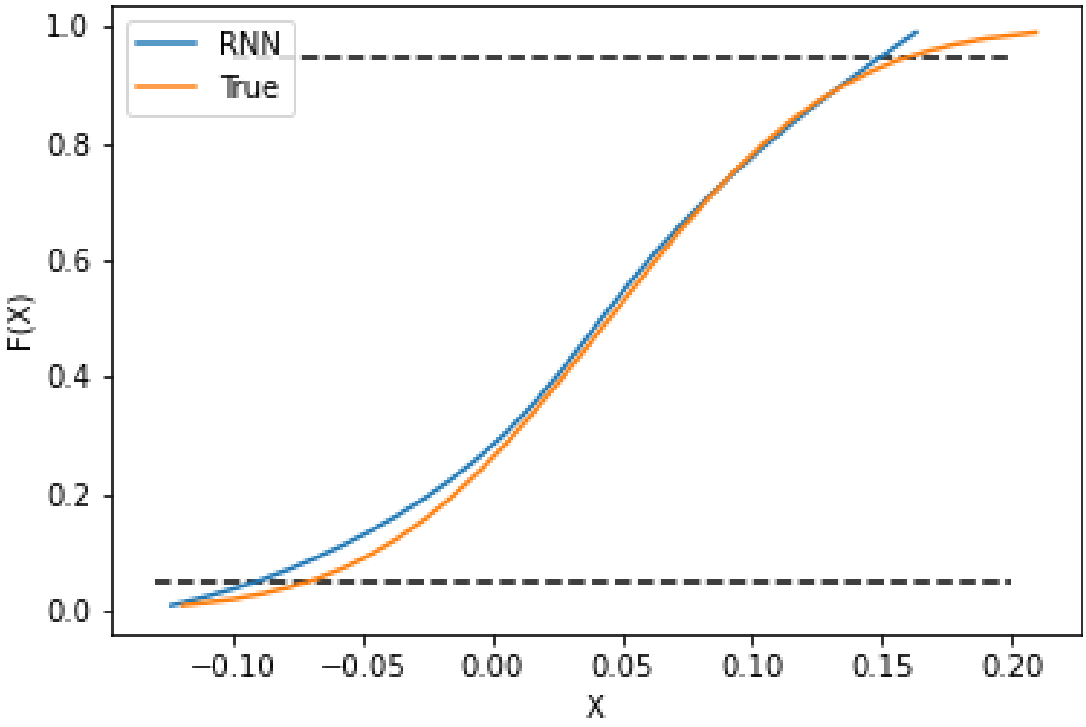

Figure A.3: True posterior distribution functions vs. RNN estimated distribution function. Horizontal lines represent the 0.05 and 0.95 quantiles.

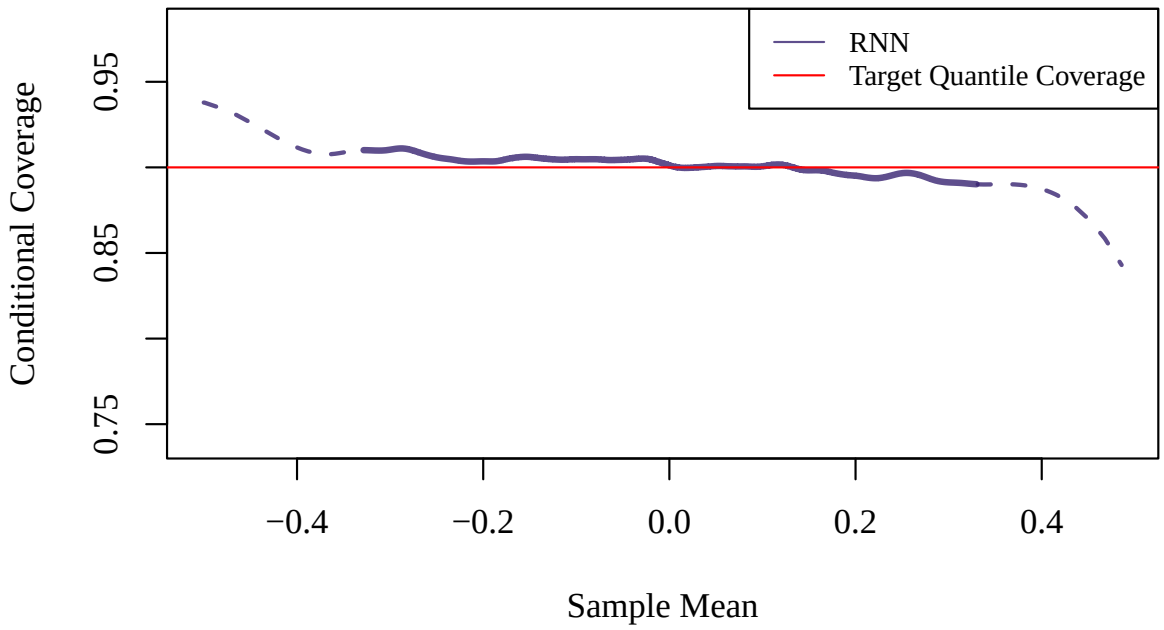

Figure A.4: Conditional coverage of RNN posterior quantile estimates, stratified by sample mean. Dotted lines represent values in the lowest or highest 1% of our test set.

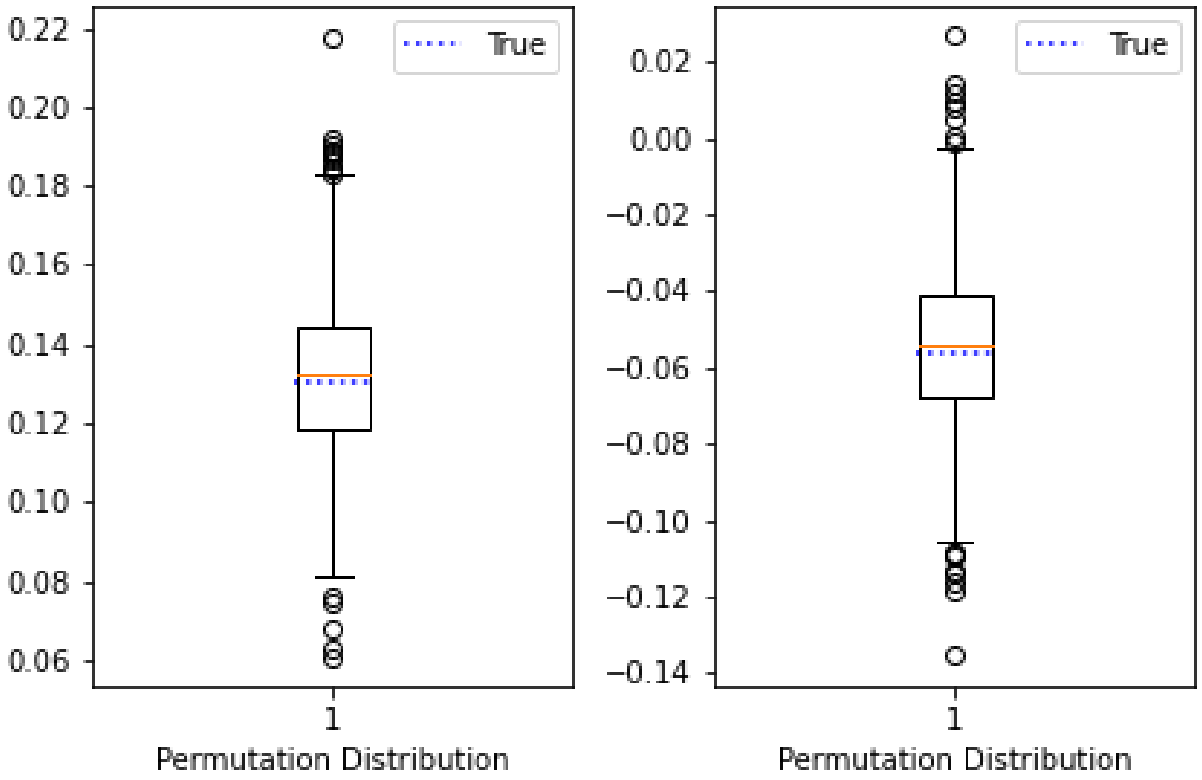

Figure A.5: Boxplot of predicted values under permutation with a superimposed true posterior value for two example datasets.

ple, which improved performance. In this specific setting, because we are interested in the posterior distribution of the maximum component, sorting is an intuitive solution.

Appendix B

OPTIMIZING BAYESIAN ADAPTIVE ENRICHMENT CLINICAL TRIALS WITH DEEP REINFORCEMENT LEARNING

B.1 Training Setup

As mentioned in the main text, we do not use Algorithm 1 exactly to obtain the results in our simulation section. Instead of a set of action states, we have a single point (for the two block trial). Our neural network also does not take as an input the entirety of the trial, but rather an estimate of the empirical distribution function for the unknown cutpoint, c, as well as the score from the first block of the trail. This leads to the algorithm outlined below. Note that in our time-to-event example, Y is two-dimensional, containing both the intermediate indication of successful treatment, as well as, if it is observed, the event time.

Additionally, due to random initialization in training, it is possible for the trial policy to get stuck in a local-minimum. To combat this, for each utility function, we initialized 5 separate neural networks, and selected the one which, at the conclusion of training, had the highest average utility on a test set. The addition of several random starts was particularly useful in the time-to-event example, where a rule which always enrolls participants regardless of their X values is a particularly strong local minimum.

B.2 Model Comparison

In all of our simulation settings, we compared to a likelihood-based trial policy. As we were already calculating $\widehat{\mathrm{F}}_c$ for input into the neural network, a natural comparison was to use this function to produce a trial policy which is rooted in the likelihood of c. To achieve this, we used as our policy $\widehat{\mathrm{F}_c^{-1}(p)}$, an estimate of the p^{th} quantile of c. For lower values of p, this

Algorithm 3 Learn a Two-Block Cutpoint Trial Policy

Require: N = batch size, k = learning rate

Set $\beta = \beta_0$

for $0 \leq i \leq$ total iterations **do**

for $0 \leq j \leq$ N **do**

1. Sample from $\mathrm{P}(\theta, c)$
2. Sample $X_1, \ldots, X_n \sim \mathrm{P}(X)$
3. Sample $Y_1, \ldots, Y_n \sim \mathrm{P}(Y|X_i, T_i, \theta, c)$
4. Calculate $\mathrm{P}(c|X, Y)$ for a grid of 20 candidate cutpoint values spanning the prior domain. Use these to approximately construct a conditional distribution function for $c, \widehat{\mathrm{F}}_c$
5. Sample from $\mathrm{P}_{\beta_i}(\widehat{\mathrm{F}}_c)$
6. Enroll and complete trial based on policy output by the neural network.
7. Calculate $q(S_{n,j})$

end for

1. $\beta_{i+1} \leftarrow \beta_i + \frac{k}{N} \sum_{u=1}^{N} q(S_{n,u}) \nabla_\beta \log \mathrm{P}_\beta(S_{\cdot,u}, A_{\cdot,u})$

end for

policy will output a trial policy cutpoint which is close to 0, meaning it enrolls nearly every subject screened. As we increase the value of p, we increase the average power of the trial, while also increasing the average number of subjects screened. There is a parallel between different values of p, and different values of the penalty parameter λ in our neural network utility function, q. Therefore, we are able to compare the two, and visualize the increase in power that the reinforcement learning based approach has when compared to a naive, likelihood-based approach.

An illustration of the parallels between the neural network and likelihood based models can be seen in the figures below. There is also a figure illustrating how the neural network based indication is used in a simple enrollment rule throughout the trial.

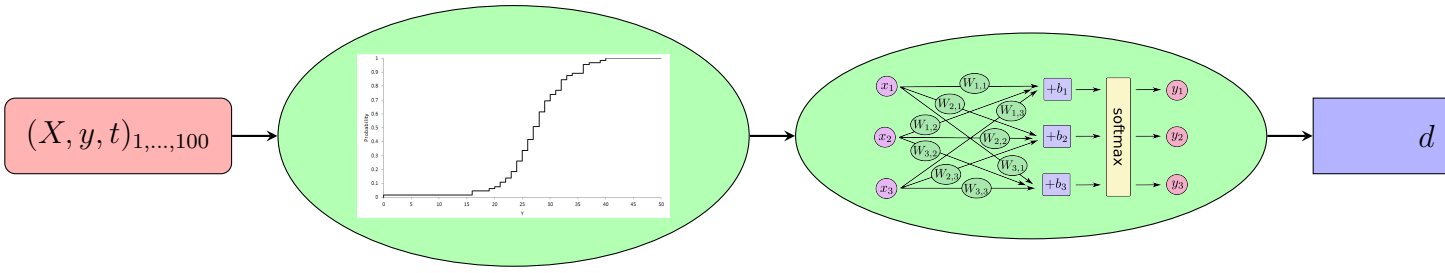

Figure B.1: Visual workflow for a neural network trial policy. Summary statistics, such as the empirical distribution for an unobserved covariate threshold which impacts the effect of treatment, are input into a neural network which determines an enrollment cutpoint.

B.3 Extensions

In the simulations described in this work, the underlying relationship between baseline covariates and treatment effect is a simple step function. However, this simplicity is not necessary. One could imagine a more complicated conditional mean, and corresponding expected benefit of treatment. Consider the general case $E(Y|X, T = 1) - E(Y|X, T = 1) = f(X, T)$. Here a simple cutpoint trial policy space is insufficient. However, we can imagine training a

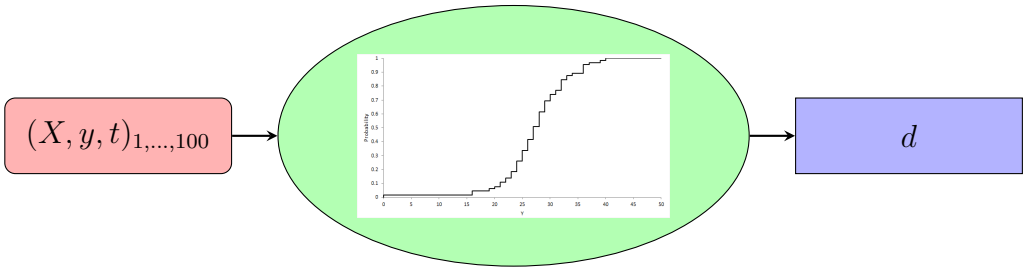

Figure B.2: Visual workflow for a heuristic-based trial policy. The enrollment cutpoint is determined by a quantile of the empirical distribution for the unobserved covariate threshold which impacts the effect of treatment.

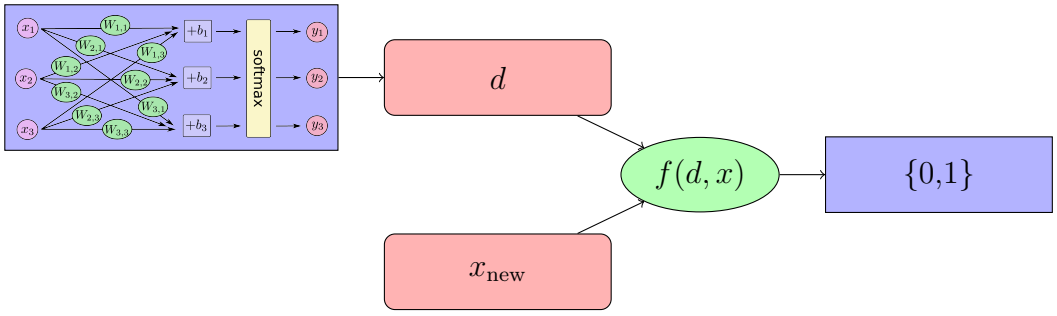

Figure B.3: Visual workflow describing how a neural network generated indication governs enrollment. The neural network outputs an indication d, after which a simple function of d is used to determine enrollment for (potentially) the remainder of the trial.

neural network which takes as input all of the current patient covariates, treatment assignments, and outcomes, $\{X, Y, T\}$, and outputs a vector θ. The p-dimensional vector θ is a parameterization of a B-spline basis [85]. The enrollment rule can then be derived from this more complicated function. Further work should be done to investigate the performance of reinforcement learning in this more complicated trial setting.

Appendix C

INFERRING MECHANISTIC PARAMETERS OF SOMATIC HYPERMUTATION USING NEURAL NETWORKS AND APPROXIMATE BAYESIAN COMPUTATION

C.1 Additional Sequence Information

C.1.1 Transition Matrix Estimation

We would like to, conditional on a particular pathway being recruited to a site, estimate the mutation frequency and relative rate of mutation to different bases. While these transition vectors stay fixed in our model, they are difficult to choose values for. An example of such a challenge is the pol-η transition matrix. In order to try to find reasonable values for this matrix, we looked at the mutation spectra for bases which neighbored a mutation at an A/T site. The idea behind this is that bases directly neighboring A/T mutation sites are very likely to be in the same exo stripping region as the A/T mutation. Results from this analysis can be seen in Figure C.1. However, there is evidence that pol-ζ, another, minor mutation pathway, preferentially causes tandem, or distance one, mutations [70]. To adjust for this, we repeated this analysis, but instead of looking at mutations at distance one, we looked at mutations of distance two. Results can be seen in Figure C.2. Notably, we see significantly fewer C/G mutations than in the distance one case.

C.1.2 Cross colocalization

There are two versions of the colocalization in the literature. The first is described in the summary statistics section and is appropriate in the situation where you can get non-parametric estimates of the probability of mutation at each site. The second is the one that was used in previous work [79] and is useful when you only have access to model-based

Table C.1: Estimated transition matrix for sites one base away from an A/T mutation.

	Child Base			
Parent Base	A	C	G	T
A	.905	.026	.039	.030
C	.033	.880	.030	.057
G	.078	.038	.848	.036
T	.026	.033	.025	.916

Table C.2: Estimated transition matrix for sites two bases away from an A/T mutation. There appear to be significantly fewer C/G mutations than there are at distance one.

	Child Base			
Parent Base	A	C	G	T
A	.895	.026	.049	.031
C	.022	.909	.031	.038
G	.046	.035	.896	.022
T	.028	.039	.022	.911

estimates of the probability of mutation at each site. If we have N sequences, sequence i has length ℓ, m_i^r is the number of pairs of mutations a distance r apart from each other, and $m_i^+ = \binom{\#\text{ mutations in sequence } i}{2}$ is the total number of pairs of mutations in sequence i. Recall that the "SWM" [79] colocalization is

$$\text{coloc}_{\text{SWM}}(r) := \frac{1}{N}\sum_{i=1}^{N} \frac{m_i^r/m_i^+}{(\ell_i - r)/\binom{\ell_i}{2}} \tag{C.1}$$

We want to generalize this to a cross-colocalization, where we look at specific pairs of bases instead of simply whether a mutation occurred. To make this generalization, we need to understand the colocalization expression above. Both the numerator and the denominator

Table C.3: Estimated transition matrix for sites three bases away from an A/T mutation. The difference in base mutation appears to be largely eliminated.

	Child Base			
Parent Base	A	C	G	T
A	.918	.020	.041	.021
C	.017	.918	.025	.041
G	.033	.026	.920	.021
T	.018	.034	.019	.929

can be interpreted as probabilities. The denominator is the probability that you would choose a pair of locations at a distance r in sequence i if you were sampling uniformly at random from all the pairs. The numerator is the probability that you would choose a pair of mutations at distance r in sequence i if you were sampling uniformly at random from all the pairs of mutations. We will set up our expression for the cross-colocalization so that it has the same form.

We need some new variables: $m_i^{b_1,b_2,r}$ will be the number of pairs of mutations in sequence i at a distance r in which one base is b_1 and one base is b_2. $m_i^{b_1,b_2,+}$ will be the number of pairs of mutations in sequence i in which one base is b_1 and the other base is b_2. $n_i^{b_1,b_2,r}$ will be the number of pairs of positions in sequence i a distance r apart in which one base is b_1 and one base is b_2. $n_i^{b_1,b_2,+}$ will be the number of pairs of positions in sequence i in which one base is b_1 and one base is b_2. Then we define

$$\mathrm{coloc}_{\mathrm{SWM}}^{b_1,b_2}(r) := \frac{1}{N}\sum_{i=1}^{N} \frac{m_i^{b_1,b_2,r}/m_i^{b_1,b_2,+}}{n_i^{b_1,b_2,r}/n_i^{b_1,b_2,+}}.$$

C.2 Some notes on strandedness

One of the more subtle things that happens in somatic hypermutation is how mutations accumulate on the two strands, and how those mutations are transferred to the daughter

cells when mitosis occurs. AID is most active in the G1 phase, which is also when the most U's are found in the immunoglobulin genes [61]. Most of the A/T and C/G mutations are generated in the G1 phase, but some C/G mutations also occur in the S phase (when the DNA is replicated in preparation for mitosis). This suggests that the primary mutagenic pathways (e.g. AID, MMR, UNG) are active in G1. U's that are not processed by another pathway can introduce mutations by replicated over during the S phase, leading to an A being incorporated opposite the U. Notice also that to the extent that the mutations are generated in G1, faithful replication in the S phase will lead to mutations on one strand going to one daughter cell and mutations on the other strand going to the other daughter cell.

There is also some evidence of strand bias in the mutagenic processes.

Since AID followed by no repair, just replication over the resulting U, results in a transition (either C to T if the deamination occurred on the coding strand or G to A if the deamination occurred on the non-coding strand), the evidence of strand bias comes from the relative rates of C to T vs. G to A transitions.

There is also evidence of strand bias in the other pathways. There are 2x more A mutations on the coding strand than T mutations, and since these mutations are introduced by the MMR pathway, it follows that there is some strand bias in this sort of repair. The differences here could be a downstream result of the AID strand bias, although it seems to be a slightly larger effect than the potential AID strand bais. In addition, there are 3.5x more G to C than C to G transversions. These transversions are thought to primarily come from the creation of an abasic site from the U introduced by AID.

C.3 Probit Gaussian Cox Process

One of the key components of our model which induces spatial colocalization in mutations is the probit Gaussian Cox process (PGCP). Given an unmutated sequence, S, of length L, the first step of the process is to sample what we call prelesions. For each site in the sequence,

the prelesion random variable is defined by the following,

$$X_i = \begin{cases} \mathbf{1}(S_i = C), & \text{w.p. } p_i \\ 0, & \text{otherwise} \end{cases}$$

where $X_i = 1$ indicates that there is a prelesion at site i, and p_i is the prelesion probability, potentially depending on the sequence context at position i.

After sampling the prelesion vector, we then sample a Gaussian process with a prespecified mean, variance, and lengthscale. The latter two are targets of estimation and the mean is fixed. Prelesions become lesions if the Gaussian process at their position, re-scaled to the unit interval, is positive. For example, if there is a prelesion at position 10 in a sequence of length 100, it will become a lesion only if $G_{0.1} > 0$, where G is the aforementioned Gaussian process. For the Gaussian process, we use a squared exponential kernel and a fixed mean of -10, which produces the following marginal distribution and covariance,

$$G_i \sim \mathcal{N}(-10, \sigma^2)$$
$$\text{Cov}(G_i, G_j) = \sigma^2 \exp(-\frac{(i-j)^2}{2\ell^2})$$

where σ^2, ℓ are parameters we would like to estimate from observed sequences. An example of a realization of a PGCP can be seen in Figure C.1.

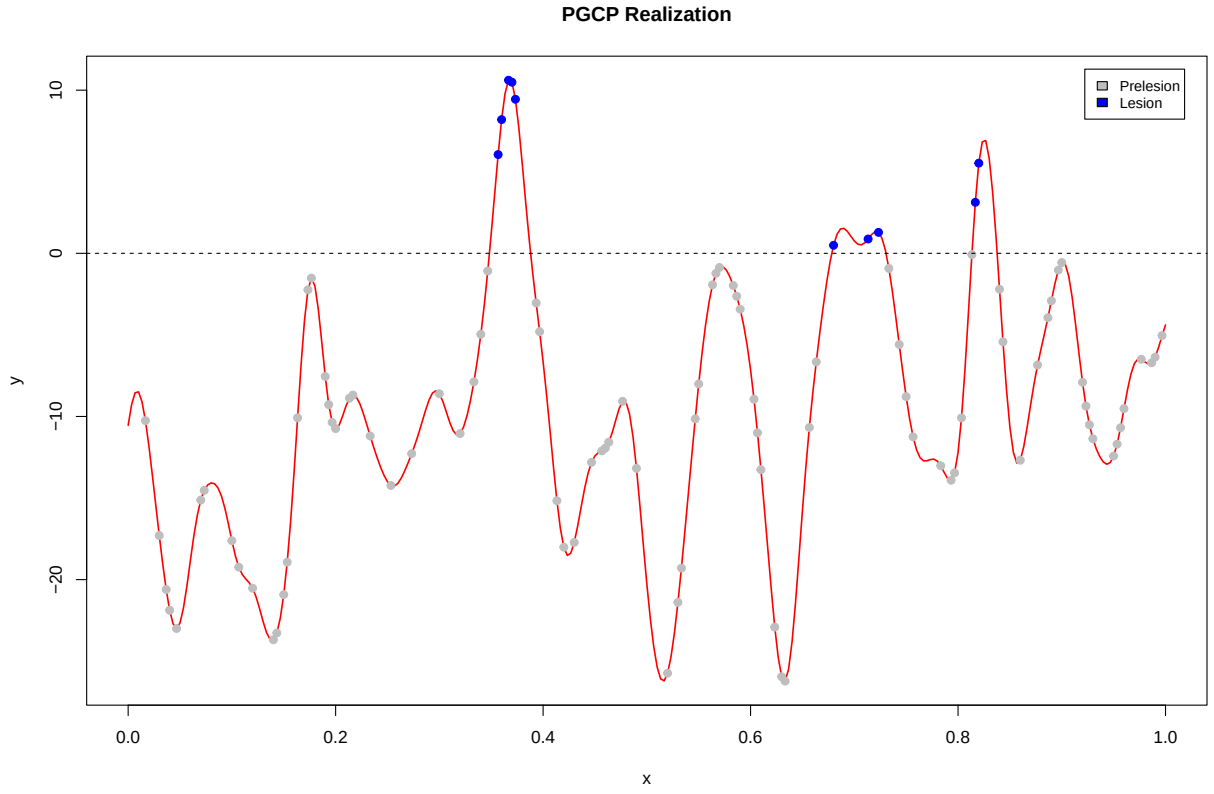

Figure C.1: A realization of a probit Gaussian Cox process. Grey points indicate prelesions which did not become lesions. Blue points indicate lesions. The covariance structure of the Gaussian process is such that lesion sites cluster together. This clustering structure allows for spatial colocalization in mutations that mimics those in real sequences.