Bayesian Adaptive Clinical Trials: A Soft Actor-Critic Approach

Matthew Willer

Advisor: Professor Daniel E. Rigobon

Submitted in partial fulfillment
of the requirements for the degree of
Bachelor of Science in Engineering
Department of Operations Research and Financial Engineering
Princeton University

May 2025

Abstract

Adaptive clinical trial designs are aimed to improve efficiency and enhance ethical considerations by dynamically allocating patients to treatments based on accruing evidence. This thesis proposes an adaptive clinical trial as a finite-horizon Markov Decision Process (MDP). The trial state comprises patient outcomes and Bayesian-updated treatment success probabilities and is sequentially updated at each decision point. To solve the resulting treatment allocation decision-making problem, we implement a Soft Actor-Critic (SAC) framework that leverages maximum entropy reinforcement learning to balance exploration and exploitation effectively. To further capture this balance, we employ a weight-adjusted Total Variation Distance (TVD) component to the reward function. This enables us to quantify the value of information gathered between decision points. We conducted numerical simulations where the agent was trained under two training schemes: one where outcomes were generated using the true treatment success probabilities, and another where outcomes were based on the agent's estimated probabilities. Across diverse hypothetical scenarios varying in cohort size, trial length, and prior knowledge, our SAC-based policy consistently approximated the ideal (oracle) policy in the true probability setting. The agent was able to achieve success proportions close to that of the optimal policy while judiciously allocating more patients to the superior treatment. When the model was trained on estimated probabilities, performance degraded under high uncertainty or poorly specified priors, sometimes favoring a fixed, non-adaptive approach. Our results underscore the potential and limitations of employing SAC in adaptive trial design. Our proposed model provides a foundation for utilizing reinforcement learning in a clinical trial setting, highlighting the need for accurate prior information to fully realize its benefits. Our framework establishes a rigorous testbed for adaptive patient allocation, providing both theoretical insights and practical guidelines for future clinical trial designs.

Acknowledgements

First and foremost, I want to thank God for inspiring me throughout my college experience and illuminating the ideas and creativity displayed throughout this paper.

Secondly, I would like to thank my family for the endless support and love that has guided me and built me up for the time I've spent at Princeton. Mom, I could not have developed my deep passion for math if it weren't for your influence and inspiration. You've always been an incredible source of empathy and your constant love is something I truly cherish. Dad, thank you for the support and guidance you've given me. You inspire me to be a better man and are truly someone that I look up to. I love you both very much.

To my Aunt Therese and Uncle Tony, I want to thank you both for being my New Jersey parents and always looking after me through my ups and downs during my time here. I'm forever grateful for the support and love you have shown me, and all the moments we'll share in the future.

To my Uncle Joe and Aunt Tracy, your incredible support since I was a kid is something I will be forever grateful for. You've both pushed me to explore areas of engineering and math that I've found a true love for and instilled a curiosity for problem-solving that I will carry forever.

To my friends, thank you. Endless bees and forest drives made these years extremely memorable. Spending day after day with you all brought light to even the most stressful times. Late-night talks and constant encouragement made this journey not only bearable but unforgettable. I am beyond grateful for the memories we've made and the moments we've shared. The brotherhood will always live forever, and I will never forget the moments we've shared.

To my friends back home, you guys have seen me at my highest and lowest. You've helped shape me to be who I am today, and I thank you for the impact you've had on me. Dylan, you're basically my brother—from the late talks driving through the hills of CA to the days spent at Stanford, I'll always cherish the memories we have and look forward to the ones to come.

Prof. Rigobon, I want to thank you for your rare ability to make existential uncertainty feel oddly comforting. You once said, "There's a very fine line between the madness of the masses and the wisdom of the crowds," and I've been trying to figure out which side to listen to ever since. Your honesty, your curiosity, and your refusal to offer easy answers made this process more challenging and infinitely more rewarding. Thank you, Dan.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

With the field of medicine constantly evolving, clinical trial results are one of the most important and expensive aspects of new drug development. The cost of bringing a new drug to market is estimated to be around \$2.6 billion [8], putting a strong emphasis on optimized trial design. Clinical trials themselves are intended to demonstrate the efficacy and/or safety of a treatment. This is done by testing a null hypothesis with a specified significance level. Rejecting the null hypothesis provides statistically significant evidence that the treatment is efficacious, and conversely failing to reject the null hypothesis leads to the conclusion that the treatment is ineffective. Intuitively, the statistical results can make or break whether a drug goes on to a further stage, and more importantly, whether the company developing the drug receives further funding or not.

In terms of clinical trial design itself, the traditional and most basic trial design allocates a fixed proportion p of subjects to treatment and $1 - p$ to a placebo. Trials designed as such do not employ any learning components as the trial progresses, which can result in sub-optimal expenditure in an area where many pharmaceutical or biotechnology companies are already heavily budget-constrained.

Recently, broader research on adaptive trial designs has grown, notably after the U.S. Food and Drug Administration (FDA) encouraged the use of adaptive designs in 2019 [10]. Adaptive trials focus mainly on treatment allocation, which varies depending on the phase of the trial. Earlier stage trials (e.g., Phase I) focus mainly on safety confirmation, drug administration, and maximum tolerated dose (MTD), middle stage trials (e.g., Phase II) focus mainly on finding the most successful dosage (MSD) amounts, and later stage trials (e.g., Phase III) typically focus on efficacy compared to an alternative or standardized treatment [20]. Despite the variability of the intervention depending on the stage of the trial, the adaptive framework, at a high level, is the same. Throughout the trial, patient response is monitored, and during the planned stages of intervention, allocation is changed based on this accumulated data. Structuring as such can be extremely helpful from both an exploration and a safety perspective, as it provides a greater ability to test multiple treatments on a given patient as well as the ability to terminate treatment early if adverse effects are apparent.

Importantly, the ability to discretize a clinical trial and adjust allocations at each intervention stage, you do so in a way that aims to minimize the number of patients assigned to the inferior arm. In principal, this adaptive ability enables a more optimized clinical trial design to be formulated, aimed at improving the overall outcomes of the patients in the trial.

1.2 Literature Review

1.2.1 Objectives and Paradigms of Adaptive Trial Design

Clinical trials traditionally follow a fixed design, where key parameters (i.e., sample size, allocation ratios, endpoints) are chosen in advance and no changes are allowed once the trial is underway [23]. On the other hand, adaptive clinical trial designs permit prespecified modifications to the trial procedures based on interim data analyses [23]. Such adaptations can occur at one or more interim points during the trial, which enables the experimenters to learn and adjust allocations as patient outcomes accumulate, rather than waiting until the trial's end. Structuring a trial in an adaptive way has the overarching goal of making trials more efficient and ethical. This goal can be achieved, for example, by stopping the trial early if the treatment is overwhelmingly successful or futile, allocating more patients to superior treatments, or focusing on responsive subpopulations [23]. Because of the mentioned efficiency and ethical benefits, adaptive designs have grown in popularity over the past two decades. To that end, Hatfield et al. (2016) note that adaptive designs have been most employed in oncology trials, namely due to greater regulatory acceptance towards adaptation [17]. Further, Park et al. (2018) highlight how adaptive designs can improve the probability of trial success, reduce resource use, shorten development time, and reduce the number of patients exposed to inferior treatments [23].

The primary motivations for adaptive designs include ethical and efficiency gains, as the adaptive nature enables the allocation of more patients to superior treatments as evidence accumulates. As such, many are drawn to the adaptive model's potential to improve patient outcomes during a clinical trial. Further, adaptive frameworks can shorten development time and reduce costs by stopping early for success/futility [23]. By using accumulating information, adaptive trials increase flexibility compared to traditional fixed designs, where you simply wait till the trials' conclusion to see the outcome. For example, the I-SPY 2 breast cancer trial adapts in real-time to patient biomarker profiles and interim efficacy results, allowing promising therapies to be identified faster [4, 37]. Likewise, the REMAP-CAP platform trial in critical care employs adaptive randomization to efficiently evaluate multiple interventions within a single trial framework. Within their trial design, they specifically highlight that the advantages of their design include "efficient use of data, improved participant safety, reduced downtime between trials, and enhanced knowledge translation" [3]. As such, the I-SPY 2 and REMAP-CAP trials illustrate how adaptation can answer more questions with fewer patients and in less time than separate conventional trials.

Despite clear advantages, adaptive trials also present challenges. Proper statistical control is crucial to avoid inflating Type I errors when multiple looks or modifications are introduced [23]. Complex adaptations must be prespecified and simulated extensively to convince regulators of their validity [10, 23]. Operationally, trial infrastructure must support rapid data turnaround and decision-making at interim analyses. Not surprisingly within the space of medicine, there are also concerns about transparency and potential bias. For instance, unblinded adaptations could inadvertently reveal information and influence investigator behavior [23]. Regulatory guidance now exists (e.g., the FDA's 2019 guidance mentioned before) emphasizing careful planning and methodological rigor. The following sections review major adaptive design frameworks, with emphasis on Bayesian methods, response-adaptive randomization, and covariate-adjusted designs, before exploring how machine learning techniques are expanding the adaptive toolbox.

1.2.2 Bayesian Adaptive Design Frameworks

Bayesian methods play a central role in many adaptive trial designs. The Bayesian framework naturally accommodates ongoing learning: prior information is combined with accumulating data to update posterior probabilities of treatment effects, which can then trigger adaptations. This approach offers flexibility in decision-making like probability-based stopping rules or adaptations, and can make use of all available information at interim analyses [37]. Berry (2006) describes the Bayesian perspective as "ideally suited for synthesizing information from multiple heterogeneous sources [and] its focus on probabilities of hypotheses for existing data makes it ideal for retrospective analyses" [5]. In Bayesian adaptive designs, interim analyses often evaluate the posterior probability that a treatment is superior. If this probability exceeds a predefined threshold, the trial might adapt by stopping early for efficacy or dropping inferior arms. Conversely, low posterior probabilities can trigger early futility stopping [37]. Such rules were used in the I-SPY 2 trial, which employed Bayesian hierarchical modeling and predictive probabilities to adaptively randomize patients among treatment arms. In their trial description, they highlight "regimens that show a high Bayesian predictive probability of being more effective than standard therapy will graduate from the trial with their corresponding biomarker signature(s)" [4]. As such, Berry and colleagues developed I-SPY 2 to improve efficiency in the Phase II setting by identifying effective regimens within patient subtypes (biomarkers) more quickly.

Bayesian adaptive designs have been especially influential in oncology. The BATTLE trial in lung cancer, for example, adaptively allocated patients to targeted therapies based on Bayesian analysis of biomarker–outcome relationships. Similar to I-SPY 2, the BATTLE trial focused on learning which subgroups benefited from which therapy, and subsequently exploiting that learned knowledge [37]. In platform trials like REMAP-CAP, Bayesian modeling enables simultaneous evaluation of multiple interventions and combinations. Namely, patient outcomes update posterior distributions for each intervention, guiding ongoing randomization ratios and potentially adding or dropping arms mid-trial [3]. These Bayesian platform trials exemplify how adaptive designs can answer multiple questions under one master protocol. Instead of using typical statistical analysis methods (i.e., hypothesis testing) that fixed trials rely on, Bayesian trials instead use computational/simulation-based methods like Monte Carlo simulation to calculate posterior probabilities. Further, the use of posterior error probabilities or Bayesian stopping rules (often corresponding to desired frequentist error levels), are employed to help strong control over error rates. Conveniently, the FDA guidance recognizes these aspects and outlines certain simulation-based approaches [10].

One advantage of Bayesian adaptive designs is the ability to incorporate prior knowledge such as historical data or expert belief into the analysis. This can improve efficiency when prior information is reliable, essentially borrowing strength to reduce the required sample size. However, heavy reliance on priors can be controversial, as if the prior is misspecified, it may bias results. This can be a big problem because regulators typically scrutinize the choice of certain parameters in confirmatory trials. Namely, scrutiny of a trial's power is broadly known, so naturally, the choice of priors is paramount. Another feature is predictive probability monitoring: at interim points, one can calculate the probability of eventual success if the trial were continued to the end, and stop early if this predictive probability is too low or high [37]. Thall et al. (2007) and others have demonstrated such Bayesian early stopping rules that maintain ethical equipoise while controlling error rates [32]. Although there is yet to be a widespread adoption of Bayesian adaptive trials in practice, the research efforts have grown considerably, especially with the growing application of machine learning in healthcare.

1.2.3 Response-Adaptive Randomization (RAR)

Response-adaptive randomization is a class of designs where the probability of assigning patients to each treatment arm is adjusted throughout the trial based on observed outcomes of participants enrolled so far [35]. The core idea, introduced in seminal works by Thompson (1933) and others, is to "play the winner", meaning allocate more patients to arms showing better efficacy, in order to treat patients more successfully on average [35]. In a two-arm trial, for example, if interim data suggest the experimental treatment is outperforming the control, an RAR design will progressively assign a greater fraction of new patients to the experimental arm. Formally, Thompson's Bayesian adaptive allocation (now often called Thompson Sampling) assigns patients according to the posterior probability each treatment is superior. Other RAR procedures include the "play-the-winner" rule and biased coin designs, which alter randomization ratios based on observed responses while maintaining some randomization to explore each arm [1].

RAR appeals for ethical reasons as it can reduce the number of patients exposed to inferior treatments, a point often highlighted as a key ethical advantage of adaptive trials [23]. In oncology trials, for instance, RAR might spare patients from an inferior standard therapy if the experimental treatment shows early promise. However, a well-known trade-off is that highly unbalanced allocations can reduce statistical power or precision in estimating treatment effects [25]. Meaning, that if one arm is allocated to only a small fraction of patients, comparing outcomes between arms becomes more uncertain. For this reason, many RAR designs impose constraints such as not letting the allocation probability for any arm go below a certain floor. Doing so aims to ensure continued learning about all treatments, and avoid the potential loss of power. Robertson et al. (2023) illustrate the RAR setup in a setting with K treatment arms. By letting $\mathbf{a_i} = (a_{0,i}, a_{1,i}, \ldots, a_{K,i})$ represent the allocation vector for patient i as well as denoting $\mathbf{a}^j = \{\mathbf{a_1}, \ldots, \mathbf{a_j}\}$ and $\mathbf{y}^{(j)}$ denote the sequence of allocations and responses observed for the first j patients respectively. In doing so, RAR determines the allocation probability $\pi_{k,i}$ conditioned on the previous allocations and responses observed, $\mathbf{a}^{(i-1)}$ and $\mathbf{y}^{(i-1)}$. Specifically, the authors denote this probability as follows:

$$\pi_{k,i} = \mathbb{P}(a_{k,i} = 1 \mid a^{(i-1)}, y^{i-1}) \tag{1.1}$$

where $a_{k,i}$ is a binary indicator variable that denotes the observed treatment allocation for patient i, taking a value of 1 if patient i and allocated to treatment k and 0 otherwise [25].

From a theoretical standpoint, Hu and Rosenberger (2006) established many asymptotic properties of RAR procedures, showing that certain designs can target a desired allocation proportion that optimizes an objective such as power or patient success [18]. On the other hand, critics have pointed out potential pitfalls of RAR in confirmatory trials. For example, Korn and Freidlin (2011) argued that in some cases RAR offers minimal ethical advantage yet complicates the trial and its analysis. This is the case because the clinical trial setting is highly fixated on the scientific/biological side of their research, and less on the mathematical framework of their trial if they're only getting a minor benefit. Blinded adaptive randomization can also be problematic as if outcomes are obvious (e.g., survival vs. death), investigators might guess the shifting allocation ratios, potentially unblinding the trial and introducing bias [23]. Despite these concerns, RAR is a great foundation to build off of for adaptive designs. Much literature has shown (typically via simulation) the design superiority over fixed designs, however as mentioned before actual implementation of such designs has been minimal.

1.2.4 Covariate-Adaptive and Covariate-Adjusted Designs

Covariates (patient characteristics) can play a crucial role in trial design adaptations. Covariate-adaptive randomization generally refers to methods that adjust allocation probabilities to achieve balance on baseline covariates across treatment arms. A classic example is Pocock & Simon's minimization (1975) method, which isn't response-adaptive, but does adaptively assign each new patient to the treatment that would minimize imbalance in stratification factors [24]. Such methods ensure the comparability of groups with respect to important prognostic variables, improving trial credibility and statistical power. Minimization and related covariate-adaptive schemes are widely used in practice (especially in smaller trials) to prevent random imbalances in covariates. They are typically considered separate from response-adaptive designs, we mention it here to highlight that adaptiveness in trial conduct can also target covariate balance and not strictly patient outcomes.

More pertinent to our focus is covariate-adjusted response-adaptive (CARA) designs, which naturally incorporate patient covariates into the response-adaptive algorithm [36]. Here, the idea is that treatment success probabilities may differ across subpopulations defined by covariates, hence an adaptive algorithm should account for these differences rather than treating the patient population as homogeneous. Rosenberger et al. (2001) pioneered CARA for binary outcomes, showing how to adjust the randomization ratio using both past outcomes and covariate information [27]. One approach is to effectively stratify the adaptive randomization by covariate profiles. For instance, in a trial with male and female patients, a CARA design might maintain separate adaptive randomization schedules for each sex, recognizing that treatment efficacy could differ by sex. Alternatively, covariates can be integrated through modeling, where one fits an outcome model that includes covariates, and then uses the model's predictions to guide allocation for incoming patients. Rosenberger and colleagues did so by formulating an adaptive allocation using a standard logistic response model. Specifically, they consider two treatments, A and B, with binary responses ($X_i = 1$ if success; $X_i = 0$ if failure. In letting p_i denote the probability of success for the ith patient, T_i the treatment indicator (1 if $A; 0$ if B), and z_i the K-vector of covariates, they assume the following standard logistic response model:

$$\text{logit}(p_i) = \alpha + \beta T_i + z_i'\gamma + T_i z_i'\delta. \tag{1.2}$$

Here, β represents the treatment main effect, γ the K-vector of covariate main effects, and δ the K-vector of treatment-covariate interactions. Using this model, maximum likelihood estimates of the parameters can be obtained from prior patient data. The resulting covariate-adjusted treatment effect is then mapped via the logistic function to determine the probability of assigning each new patient to treatment A, thereby favoring the treatment with a higher predicted success rate based on the patient's covariates.

Another noteworthy paper by Zhang et al. (2007) studied the theoretical properties of CARA under generalized linear models. They proved that with appropriate design, the allocation proportions can target an optimal solution even as covariates come into play. Specifically, they highlighted that under a CARA design, "we can find optimal allocation for each fixed value of the covariate" and subsequently characterized the asymptotic normality of estimators [38]. Cheung et al. (2014) extended this to logistic-regression settings for binary outcomes, providing methods to maintain both covariate balance and outcome adaptiveness [7]. The reason we mention these works is to highlight that standard statistical inference can be achieved despite the complex dependency of allocation on covariates and responses.

Tying back to the RAR framework we noted in the section above, Robertson et al. (2023) highlight that the RAR framework has an inherent flexibility that allows it to also depend on outcome-impacting covariates. Using the same notation defined in the RAR procedure, the authors further let $\mathbf{x}^{(j)} = \{\mathbf{x}_1, \ldots, \mathbf{x}_j\}$ denote a vector of observed covariates. In doing so, it naturally follows that the allocation probability $\pi_{k,i}$ under the CARA framework is:

$$\pi_{k,i} = \mathbb{P}(a_{k,i} = 1 \mid \mathbf{a}^{(i-1)}, \mathbf{y}^{i-1}, \mathbf{x}^{(i)}) \tag{1.3}$$

In practice, covariate-adjusted adaptivity is especially relevant to personalized medicine trials. If specific biomarkers or genetic signatures are known to predict treatment benefits, one may implement a biomarker-stratified adaptive randomization. In this setting, patients are first classified by biomarker-defined subgroups, and then within each subgroup, adaptive randomization allocates patients preferentially to the superior treatment for that subgroup [37]. This is a CARA design in spirit, though operationally it is likely more resemblent to running parallel adaptive trials in each stratum. The complication arises when biomarkers or predictive covariates are unknown and part of the trial's goal is to discover them. In such cases, more research is emerging around advanced methods like machine learning which aim to learn the covariate-treatment interaction structure on the fly.

1.2.5 Approximation Methods and Complexity Handling for MDP Adaptive Frameworks

The Simulation-based bounded Learning-adjusted ApproXimation (SLAX) model, introduced by Ahuja and Birge (2020), provided a very foundational understanding of the complexity of framing an adaptive clinical trial as a Markov Decision Process (MDP). On top of this, their work provides a practical solution for managing the complexity of adaptive clinical trial designs. To start, the authors recognize the impracticality of solving large-scale MDPs exactly in an adaptive trial setting, due to the curse of dimensionality [2]. Traditional adaptive trials often involve decision-making processes that quickly become computationally difficult as the number of treatments, patient outcomes, or trial periods increase. SLAX addresses this problem by employing approximation methods that make these computations manageable without significantly sacrificing accuracy or performance.

At a high level, SLAX simplifies the complex decision-making process involved in adaptive clinical trials through two key innovations. The first is that instead of trying to exactly calculate the optimal decision for every possible scenario, SLAX uses a grid-based interpolation approach. Specifically, the authors employ Barycentric interpolation, which is a method that finds a polynomial that passes through a given set of points [2]. In their context, this set of points is an approximated grid of the state space, where they denote the state of the overarching trial system as a vector $\mathbf{h}_t$ whose components represent the fraction of observed patient successes and failures for each treatment (denoted by $j \in J$). This approximation is particularly useful due to the impracticality of trying to determine every possible scenario in a trial, simply because of computational burden. Under their MDP framework, this grid-based interpolation estimates the approximate value function for scenarios that haven't been explicitly computed by interpolating between a small set of representative scenarios. With the value function denoted by V_t, Ahuja and Birge define this interpolation as:

$$\tilde{V}_t(\mathbf{h}_t) = \sum_{i=1}^{d+1} \lambda_t^i \tilde{V}_t(\tilde{\mathbf{h}}_t^i), \tag{1.4}$$

where grid states $\tilde{\mathbf{h}}_t^i$ define a simplex around the interpolated point, and weights λ_t^i ensure convex combinations [2]. This novel approach significantly mitigates the computational overhead typically encountered in large-scale MDPs. In the context of adaptive trial designs, this greatly enables larger and more realistic trials to be simulated effectively. While interpolation introduces some approximation errors, SLAX carefully manages through bounding. Namely, Ahuja and Birge define p_t^j as the unknown (Beta distributed) probability of observing a success for treatment $j \in J$ and prove theoretical upper and lower bounds on this interpolated value. Their conservative lower bound is defined by a greedy policy that purely exploits current knowledge:

$$V_t^{\text{lower}} = n(T-t) \max_{j \in J} \left\{ \mathbb{E}\left[p_t^j \middle| \alpha_t^j, \beta_t^j \right] \right\}, \tag{1.5}$$

and a theoretical upper bound assuming perfect future knowledge:

$$V_t^{\text{upper}} = n(T-t) \mathbb{E}\left[\max_{j \in J} \{ p_t^j | \alpha_t^j, \beta_t^j \} \right]. \tag{1.6}$$

As such, the bounding ensures that the approximate value function remains plausible and reflects realistic limits

The second key innovation is that SLAX incorporates a novel ”learning component” into the decision-making process. Clinical trials inherently face a trade-off between exploiting current knowledge and exploring new knowledge when it comes to patient assignment decisions. How do you decide whether to assign patients to treatment arms you know are already effective, or assign to treatments that haven't been studied thoroughly yet? SLAX systematically balances these competing interests by explicitly integrating a learning incentive into the objective function of their MDP setup. This means that SLAX not only seeks immediate patient success but also strategically allocates resources to improve long-term knowledge about treatment effectiveness.

Practically, SLAX employs simulation-based sampling, meaning it uses randomly selected future outcomes rather than exhaustively evaluating every possible scenario. This further reduces computational complexity, enabling the model to be retrospectively tested on real-world trials. As such, the importance of SLAX lies in its pragmatic approach: it takes a considerable step towards bridging the gap between rigorous theoretical adaptive trial methods and their practical application in real-world clinical settings. By significantly reducing computational demands while preserving accuracy and adaptive flexibility, SLAX makes advanced Bayesian adaptive trial methods more accessible and implementable. For our work, SLAX serves as a foundation that supports further methodological developments, particularly those leveraging machine learning and reinforcement learning techniques that we explore in this thesis.

1.2.6 Reinforcement Learning and Bandit Algorithms for Allocation

Reinforcement learning (RL) provides a natural paradigm for adaptive decision-making in trials. An RL agent aims to “learn” the optimal policy for assigning treatments to patients through interaction, where each patient assignment and outcome can be seen as one trial of a sequential decision process. In particular, the multi-armed bandit (MAB) problem is very closely related to response-adaptive randomization. Under a pure exploration bandit setup, the goal is to identify the best treatment arm with minimal regret. This resonates quite strongly with Phase II and III trial objectives that are focused primarily on establishing efficacy and safety against the current standard treatment [33]. Classical bandit algorithms like Thompson Sampling (TS) and Upper Confidence Bound (UCB) methods have been proposed to govern patient allocation.

Varatharajah and Berry (2022) employ such algorithms in their contextual-bandit-based model, designed to dynamically optimize treatment assignment. The "context" in their setting is a patient-specific covariate, in which the authors frame the clinical trial as an online decision-making problem. In doing so, each patient's treatment assignment is determined and based on their individual disease-related context. The authors highlight that since contexts are observable (unlike treatment success probabilities), they can set up a different context-free multi-arm bandit problem for each context. Essentially, their approach learns a policy $\pi(x)$ that maps a given patient covariate profile x to a probability distribution over treatments [35]. They set up the MAB problem and subsequently used TS and UCB algorithms to determine treatment assignments aimed at maximizing outcomes. This kind of RL algorithm can be seen as a machine learning-driven CARA design: it continuously updates a model (or value function) to predict which treatment will be most effective for a given patient and randomizes accordingly. Simulations in their work showed improved overall patient outcomes compared to non-contextual bandit designs when applied retrospectively to a full clinical trial dataset.

Most recently, researchers have proposed hybrid trial designs that aim to balance regulatory compliance with real-world treatment optimization. One such proposal is the Randomize First Augment Next (RFAN) framework introduced by Klein et al. (2025) [19]. RFAN structures clinical trials into two sequential phases: an initial randomized stage followed by an adaptive augmentation stage. The randomized component mirrors a conventional RCT and is intended to satisfy regulatory requirements by generating unbiased treatment effect estimates across the trial population. Once a sufficient baseline is established, the trial transitions to the augmentation phase, where patient recruitment and treatment assignment are actively guided by model-based uncertainty. Specifically, RFAN employs deep Bayesian active learning to preferentially enroll patients whose covariates are expected to yield the greatest information gain.

Their work is targeted particularly towards underrepresented or heterogeneous subgroups, explicitly motivated by post-deployment performance metrics. To this end, the authors introduce two forward-looking trial objectives. The first they define as Post-Trial Mean Benefit (PTMB), which seeks to maximize the clinical utility of the treatment policy following deployment. The second is Post-Trial Fairness (PTF), which prioritizes equitable outcomes across sensitive subgroups. They importantly highlight how ”currently trials are often designed to maximize the chances of obtaining regulatory approval, and employ recruiting practices that are often deficient in terms of demographic diversity” [19]. While RFAN represents a meaningful and virtuous advancement toward integrating machine learning into regulatory-confirmed trial design, it is not without limitations. Its success hinges on the reliability of model uncertainty estimates, which is an assumption that may be fragile in settings with small sample sizes or high dimensionality. Further, the fixed structure of the two-stage design risks underutilizing early patient data, especially if the randomized stage is prolonged unnecessarily. Nevertheless, RFAN offers a promising blueprint for hybrid adaptive designs that explicitly align trial conduct with both regulatory standards.

The usage of deep RL is also gaining traction in adaptive clinical trial design, though it remains an emerging area of research. For instance, Matsuura et al. (2022) propose a deep reinforcement learning-based allocation rule for dose-response Phase II trials. In their framework, the RL agent seeks to directly optimize a chosen trial-specific performance metric, such as statistical power, target dose accuracy, or mean absolute error of the estimated dose-response curve [21]. This is done by employing deep RL to learn an optimal allocation rule π^*. As their paper denotes, the expected cumulative reward from being in state s and assigning dose k is defined as $Q_\pi(s,k)$. The overarching goal is to learn π^* such that $\max_k Q_\pi(s,k)$ is optimized for every state s. Because their state space comprises dose-specific means, variances, and allocation proportions, it is continuous and high-dimensional, meaning exact dynamic programming is infeasible. Instead, the policy $\pi(k|s)$ is parameterized by a deep neural network (DNN), and trained via Proximal Policy Optimization (PPO) [21]. PPO is a policy gradient method known for its stability and performance. Mathematically, the policy is computed as a softmax over neural outputs:

$$\pi(k|s) = \frac{\exp(u_k)}{\sum_{k'=1}^{K} \exp(u_{k'})}, \tag{1.7}$$

where u_k is computed from a two-layer neural network with ReLU activations:

$$z_j^{(1)} = f\left(\alpha_j^{(1)} + \sum_i \beta_{ji}^{(1)} s_i\right), \quad z_j^{(2)} = f\left(\alpha_j^{(2)} + \sum_{j'=1}^{J} \beta_{jj'}^{(2)} z_{j'}^{(1)}\right),$$

$$u_k = \alpha_k^{(3)} + \sum_{j'=1}^{J} \beta_{kj'}^{(3)} z_{j'}^{(2)}. \tag{1.8}$$

In setting it up this way, they enable the agent to interact with a simulated clinical trial environment by repeatedly sampling dose-response outcomes. The agent updates its policy to maximize expected returns under the chosen performance metric, and their simulation results demonstrate that agent-learned policies consistently outperform classical and asymptotic-optimal designs (e.g., D-optimal, TD-optimal). We importantly highlight here that their findings indicate the applicability of deep RL within a clinical trial context [21]. Namely, in this thesis, we aim to explore how deep RL can flexibly learn efficient and robust allocation strategies tailored to diverse trial objectives and constraints.

1.2.7 Covariate Balance & Robustness of Design

As mentioned when discussing CARA methods, covariate balance aims to ensure treatment and control groups are comparable across critical baseline characteristics, such as age, gender, disease severity, or biomarkers. Without this balance, differences between groups can confound treatment effect estimates, undermining the trial's internal validity. Robustness is another key consideration in any experimental design. In a trial setting, robustness aims to safeguard against the unpredictability of patient outcomes to ensure the trial's conclusions remain valid under a wide range of potential scenarios. Clinical trials have inherently small sample sizes, high-stakes decision-making, and ethical constraints, which all amplify the need for precise and trustworthy experimental designs. By addressing both covariate balance and robustness, trial designs can simultaneously improve statistical efficiency and maintain fairness, ensuring that patient outcomes and scientific insights are not compromised.

Harshaw, Sävje, Spielman, and Zhang introduce a novel and near-optimal approach to the balance-robustness trade-off of experimental design: the Gram-Schmidt Walk (GSW). The underlying motivation for this design lies in the shortcomings of fully randomized designs. Namely, the authors highlight that randomization can worsen the comparability of treatment groups as it introduces a degree of unpredictability in the assignment process [16]. This can result in imbalances in key baseline covariates between groups, in which experimenters retrospectively wish they rerandomized groups subsequent to the results of a given experiment. Harshaw and colleagues build on the principles of discrepancy theory, particularly results from

Banaszczyk (1998) and Bansal et al. (2019). These works tackled the problem of partitioning high-dimensional vectors into subsets with low discrepancy, achieving bounds that approach near-random partitions under specific conditions. The GSW design adapts these ideas to construct experimental designs that balance covariates while maintaining randomness, ensuring unbiased treatment effect estimates.

The GSW design is notably applicable in fields such as social sciences, machine learning, and resource allocation. Balancing trade-offs between structure and randomness is critical in these areas where balance or deterministic optimization is prioritized over the interplay of randomness and robustness. In contrast, the covariate matrix within a clinical trial must address unique challenges such as patient heterogeneity, ethical fairness, and regulatory requirements. Hence, blending balance and randomness to ensure robust causal inferences takes precedence. Research on the adaptation of the GSW to clinical trials currently does not exist. This is namely due to the intermittency of subject enrollment in many trials, as well as the extreme lack of standardization in design.

In this thesis, we initially explored incorporating the GSW design into an adaptive clinical trial setting. We note that the GSW could prove helpful even for simply balancing initial patient assignments at the start of a trial. Ultimately, the lack of real covariate-level patient data led us to shift away from incorporating this design, however is included in our future work section as a potential avenue for further improvement of the model presented in the chapters to come.

Chapter 2

Markov Decision Process Framework

This chapter establishes the setup, assumptions, and computational framework involved in modeling a two-arm clinical trial as a Markov Decision Process (MDP). The central assumptions in this framework are that patient response is binary and the trial is segmented into T discrete time steps (also referred to as intervention steps or decision points). That is, at each time step, a patient either achieves clinical success or does not, based on a pre-specified threshold. In practice, this threshold would be the primary endpoint established by the drug company running the trial, before the trial starting. The framework further assumes that the trial has a fixed number of subjects N per time step and that patient outcomes are evaluated at the end of each discrete interval. By doing so, this setting would then, in practice, establish set times where the experimenter could reallocate treatments or stop the trial prematurely. In the context of the model established in this chapter, this setting provides a defined set of periods that the model is run on, enabling adaptive decision-making as the trial progresses. Rather than focusing on traditional metrics such as the Average Treatment Effect (ATE), our approach concentrates on dynamically updating the posterior probabilities of treatment success for each arm. Focusing on posteriors facilitates a direct comparison of the success probabilities between the experimental and control treatments at each discrete step of the trial. As such, this forms the basis for adaptive patient allocation until the conclusion of the trial.

2.1 Adaptive Model Setup

2.1.1 Trial Structure and Distribution Assumptions

Consistent with existing Bayesian-adaptive models, we reiterate that the clinical trial setting has a fixed number of patients N allocated for each discrete time step. In terms of the treatments, we assume a clinical setting with two treatment arms (experimental treatment (E) and control (C)) that are statistically independent with no serial correlation in treatment effects. Hence, the probability distributions for treatment success are updated separately based on the observed patient responses for each arm, where such responses are independent events conditional on the current estimated success probability [2]. As mentioned, we consider each outcome y_i as binary, and defined as follows:

$$y_i = \begin{cases} 1, & \text{if patient } i \text{ achieves clinical success,} \\ 0, & \text{otherwise.} \end{cases}$$

Many existing adaptive frameworks, such as the SLAX approach employed by Ahuja and Birge (2020), assume that patients are independent and identically distributed (i.i.d.) [2]. We explored an approach that tried to incorporate patient covariate information into the state which would thereby induce a structured dependency among patients in relation to their baseline characteristics. The idea was that relaxing the i.i.d. assumption and explicitly considering covariates would better reflect the heterogeneity observed in a clinical setting. However, this complicates the posterior update scheme of the model, because the standard Beta-Bernoulli model, discussed in this section, relies on a conjugate relationship where the success probability is assumed to be constant across patients within each treatment arm. Hence, effective incorporation of covariates has to be done with a different update mechanism that integrates the effect of covariates on the outcome. Covariate-adjusted response-adaptive (CARA) randomization procedures are relatively popular and often rely on regression-based approaches [26]. These methods naturally tend to be more assumption-heavy and can introduce additional errors. Another way to go about incorporating covariates is through the usage of a contextual-bandit-based approach where the "context" could be some disease-related characteristics of a given patient that is then used to determine treatment assignment. Varatharajah & Berry (2022) incorporate covariates (contexts) into their approach formulating a different context-free multi-arm bandit (MAB) problem for each covariate. In doing so, their approach sequentially allocates new patients to treatment arms based on the standard MAB problem that only looks at past patients with a specific covariate [35]. This type of approach is less applicable in an MDP setting because it does not capture the sequential dependency inherent in the trial's dynamics. A contextual bandit model addresses decision-making problems by maximizing an immediate reward function $\mathbb{E}[r(s,a)]$ for a given context without considering future consequences, whereas an MDP is governed by the Bellman equation which encapsulates the cumulative, discounted future rewards. All of this to say, the i.i.d. patient assumption is held in the model.

The last assumption worth mentioning is that the trial itself is assumed to have planned interventions throughout. By discretizing the trial, we enable fixed periods of analysis on observed treatment effects as well as altering treatment assignments based on these observed factors. To put this formally, we let T be the total trial length, and assume interventions occur at pre-determined time steps $t \in [0, T]$. At each decision point, the success rates for each treatment arm, $p_t^{(E)} \in [0,1]$ and $p_t^{(C)} \in [0,1]$, are modeled as Beta distributed random variables as follows:

$$p_t^{(E)} \sim \text{Beta}(\alpha_t^{(E)}, \beta_t^{(E)}), \quad p_t^{(C)} \sim \text{Beta}(\alpha_t^{(C)}, \beta_t^{(C)}).$$

The overall true success rates for the two treatment arms are unknown at the beginning of the trial. As such, at $t = 0$, we assume the treatment and control arm success rates have starting prior distributions $(\alpha_0^{(E)}, \beta_0^{(E)})$ and $(\alpha_0^{(C)}, \beta_0^{(C)})$, which, particularly in later-stage trials such as Phase III, may be informed by previous trials or other known data points. At each subsequent time step t, these parameters are recursively updated as new patient outcomes are observed, forming the posterior distributions for $p_t^{(E)}$ and $p_t^{(C)}$.

After the assignment of patients to either arm E or C for a given time step t, we realize each patient's binary outcome and can model each $y_i \in \{0,1\}$ as a Bernoulli distributed random variable with success probability based on their treatment arm:

$$y_i \sim \text{Bernoulli}(p_t^{(E)}) \quad \text{if assigned to } E, \qquad y_i \sim \text{Bernoulli}(p_t^{(C)}) \quad \text{if assigned to } C$$

Following this, we can denote $\mathbf{y}_t^{(E)} = (y_1^{(E)}, y_2^{(E)}, \ldots, y_{n_E}^{(E)})$ and $\mathbf{y}_t^{(C)} = (y_1^{(C)}, y_2^{(C)}, \ldots, y_{n_C}^{(C)})$ as vectors of observed outcomes for patients assigned to the experimental treatment and control arms respectively.

As a result, we observe the overall trial dynamics at any given timestep $t \in [0, T]$, and denote these dynamics as follows:

- $n_t^{(E)} \in [0, N]$: the total number of **patients assigned to the experimental treatment arm** at time t
- $r_t^{(E)} \in [0, N]$: the total number of **observed successes in the experimental treatment arm** at time t

- $n_t^{(C)} \in [0, N]$: the total number of **patients assigned to the control arm** at time t
- $r_t^{(C)} \in [0, N]$: the total number of **observed successes in the control arm** at time t

Hence, we see that the total number of observed successes in each arm at a given time step t can be expressed as the dot product of a ones vector with the corresponding treatment outcome vector:

$$r_t^{(E)} = \mathbf{1}^\top \mathbf{y}_E, \quad r_t^{(C)} = \mathbf{1}^\top \mathbf{y}_C$$

where $\mathbf{1}$ is a vector of ones of appropriate dimension. In doing so, this formulation allows for a compact representation of cumulative successes and facilitates efficient updates as new patient outcomes are observed. Further, since we define y_i as Bernoulli distributed, it follows that $r_t^{(E)}$ and $r_t^{(C)}$ follow binomial distributions:

$$r_t^{(E)} \sim \text{Binomial}(n_t^{(E)}, p_t^{(E)}), \quad r_t^{(C)} \sim \text{Binomial}(n_t^{(C)}, p_t^{(C)})$$

A very notable advantage of defining $r_t^{(E)}$ and $r_t^{(C)}$ in this way is that the Beta distribution serves as a conjugate prior to the Binomial likelihood. Thus, this enables efficient Bayesian updating of success probabilities. So what follows is that the posteriors for the Beta distributed success rates can be updated recursively as such:

$$(p_t^{(E)} \mid r_t^{(E)}) \sim \text{Beta}(\alpha_{t-1}^{(E)} + r_t^{(E)}, \beta_{t-1}^{(E)} + n_t^{(E)} - r_t^{(E)}) \tag{2.1}$$

$$(p_t^{(C)} \mid r_t^{(C)}) \sim \text{Beta}(\alpha_{t-1}^{(C)} + r_t^{(C)}, \beta_{t-1}^{(C)} + n_t^{(C)} - r_t^{(C)}) \tag{2.2}$$

The proof of this update rule is outlined in Appendix C.1, and for completeness, it is worth noting that this update rule can be applied sequentially for individual patient responses. This is similarly done by exploiting the Beta-Bernoulli conjugacy, which naturally arises from the direct relationship between Bernoulli trials and the Binomial likelihood.

2.1.2 Hypothesis Testing Framework

Ubiquitously used in randomized experiment settings is a hypothesis testing framework. In the context of clinical trials, this is ultimately used to determine whether the observed data provide sufficient evidence to reject a null hypothesis in favor of an alternative hypothesis. In terms of variables already defined, this is typically analyzed via either a one-tailed or two-tailed test as follows:

- **One-tailed hypothesis test:** $H_0 : p_E \leq p_C, \quad H_1 : p_E > p_C$
- **Two-tailed hypothesis test:** $H_0 : p_E = p_C, \quad H_1 : p_E \neq p_C$

It is standard practice for the experimenter to specify a significance level α (the Type I error rate) and a desired power $1 - \beta$ (the probability of correctly rejecting H_0 when it is false). In later-stage trials (e.g., Phase III), α is typically set at 0.05 or 0.01 [28].

Power is arguably of greater importance, and is defined as the probability of correctly rejecting the null hypothesis H_0 when it is indeed false. Illustratively, power is the probability of finding significant results (i.e., treatment effect) if a real effect actually exists [30]. Hence, achieving certain powering of a trial involves calculating the required sample size based on the expected effect size as well as the level of significance desired. Namely, it is about finding a balance between having enough patients in the overall sample to reliably detect a meaningful effect, while minimizing unnecessary recruitment and resource usage. That being said, power is a much harder parameter to converge on compared to the significance level, as it has a strong reliance on the trial's choice of primary endpoint(s). For treatments in development, one of the main objectives is to develop a novel and more efficacious treatment to what already exists on the market. As such, intervention methods, disease targets, response metrics, etc., are highly variable across different trials, making standardization of primary endpoint decisions almost impossible.

While the model established in the subsequent sections leverages a Bayesian adaptive framework, the traditional metrics of Type I error rate and power continue to serve as essential benchmarks. It is important to clarify how these metrics relate to our implementation. Firstly, the overall MDP-based adaptive trial model that we set up is driven by a reinforcement learning (RL) algorithm that dynamically updates the posterior distributions of the treatment success probabilities and makes patient allocation decisions to maximize cumulative rewards. In this framework, decisions are made sequentially based on the evolving state of the trial rather than through an explicit classical hypothesis test at each interim stage. Secondly, even though we don't explicitly embed hypothesis testing within the decision rules, the design can theoretically still be evaluated from a regulatory perspective. To do this, simulation studies are conducted outside the core RL algorithm. By simulating many instances of the adaptive trial under different scenarios (i.e., under a specific null and alternative hypothesis) one can estimate the operating characteristics of the design. For example, for the type I error probability, we simulate a large number of trials under the null hypothesis and apply a pre-specified decision rule (such as stopping the trial when the posterior probability that $p_E > p_C$ exceeds a set threshold). With this, the proportion of simulated trials that lead to a false positive decision (incorrect rejection of H_0) provides an estimate of the overall Type I error rate. Similarly, in order to estimate power we can run simulations under the alternative hypothesis with a true treatment effect, and observe the proportion of simulated trials that correctly lead to a decision in favor of the experimental treatment estimates the trial's power. This simulation-based assessment is in line with the FDA's recommendations [10] for complex adaptive designs, where the analytical derivation of the distribution of test statistics can be intractable. The third and last note worth mentioning is that in practice, one could retrospectively analyze the trial outcomes (such as the final posterior distributions or cumulative success differences) and perform hypothesis tests to calculate Type I error and power estimates.

2.2 MDP Framework

Now that we have most of the trial mathematically defined, we introduce the finite-horizon MDP setup [31]. It's necessary to understand and define the dynamics of MDPs before specifically applying the framework to clinical trial design.

Given a finite state space $\mathcal{X} = \{x_1, x_2, \ldots, x_M\}$, $M \in \mathbb{N}^*$, and a finite action space $\mathcal{A} = \{a_1, a_2, \ldots, a_K\}$, $K \in \mathbb{N}^*$, the MDP dynamics are as follows:

- $\forall t \in [0, T]$ we let $x_t \in \mathcal{X}$ and $a_t \in \mathcal{A}$ denote the state at time t and action chosen at time t respectively.
- We define the transition probability (i.e. the probability of transitioning from state $x_t \in \mathcal{X}$ to state $x_{t+1} \in \mathcal{X}$) by: $P(x_t, x_{t+1}, a_t) = \mathbb{P}(x_{t+1} = y|x_t, a_t)$.
- Hence, with probability $P(x_t, x_{t+1}, a_t)$ the state goes from $x_t \in \mathcal{X}$ to $x_{t+1} \in \mathcal{X}$ with reward $R(x_t, x_{t+1}, a_t)$ obtained.

2.2.1 State Space ($\mathcal{X} \in \mathbb{R}^7$)

As mentioned in the setup of the trial structure, the framework being considered is a trial discretized by planned intervention periods. This setup fits well within a finite horizon MDP framework where, at each discrete time step (i.e., decision point) t, immediately after observing the outcomes for all N patients in a given cohort, the state of the trial captures our current belief about treatment outcomes and the overall trial progress. By employing a reinforcement learning agent — discussed in the next chapter — we leverage a rich state representation that includes both the basic Beta parameters and derived metrics that provide additional context for decision-making. This comprehensive, continuous state space is designed to capture the innate variability of clinical trials. As such, at time t, the state is defined as:

$$x_t \in \mathcal{X} = \left(t_{\text{norm}}, \overline{p}_t^{(E)}, \overline{p}_t^{(C)}, \sigma_t^{(E)}, \sigma_t^{(C)}, \eta_t, \rho_t\right),$$

where:

- $t_{\text{norm}} = \frac{t}{T} \in [0,1]$ is the normalized time step, indicating how far the trial has progressed.
- $\overline{p}_t^{(E)} = \frac{\alpha_t^{(E)}}{\alpha_t^{(E)}+\beta_t^{(E)}}$ is the current posterior mean of the experimental treatment's success probability. Beta parameter updating was previously discussed, and this metric is derived from the Beta distribution, serving as an updated estimate based on accumulated patient outcomes. This is incorporated into the state space as it provides a picture of how effective the experimental treatment appears to be up to time t.
- $\overline{p}_t^{(C)} = \frac{\alpha_t^{(C)}}{\alpha_t^{(C)}+\beta_t^{(C)}}$ is similarly the current posterior mean of the control treatment's success probability, offering a direct comparison to the experimental arm. As such, both of these posterior means provide some color to which treatment might yield better results, thereby guiding allocation decisions towards the treatment with higher observed efficacy.
- $\sigma_t^{(E)} = \sqrt{\frac{\alpha_t^{(E)}\cdot\beta_t^{(E)}}{\left(\alpha_t^{(E)}+\beta_t^{(E)}\right)^2\cdot\left(\alpha_t^{(E)}+\beta_t^{(E)}+1\right)}}$ is the standard deviation of the experimental success rate, giving a measure of uncertainty in the estimated success probability for this arm. This is included to compliment the above posterior means, where naturally a higher $\sigma_t^{(E)}$ helps indicate that the experimental treatment's

performance is less certain, suggesting more exploration might be needed.

- $\sigma_t^{(C)} = \sqrt{\frac{\alpha_t^{(C)} \cdot \beta_t^{(C)}}{\left(\alpha_t^{(C)}+\beta_t^{(C)}\right)^2 \cdot \left(\alpha_t^{(C)}+\beta_t^{(C)}+1\right)}}$ is the standard deviation of the control success rate, similarly giving a measure of uncertainty in the estimated success probability for this arm. Analogous to $\sigma_t^{(E)}$, this term helps provide clarity into how reliable the control estimates are when compared to the experimental treatment arm, crucial when considering shifts in future allocations. As such, both $\sigma_t^{(E)}$ and $\sigma_t^{(C)}$ are measures that help in the exploration-exploitation trade-off when determining both the validity of current estimates and the certainty of future allocation decisions.

- $\eta_t = \mathbb{P}(p_t^{(E)} > p_t^{(C)}) = \int_0^1 f_t^{(E)}(x)\, F_t^{(C)}(x)\, dx$ is the probability that the experimental treatment produces a higher success rate than the control. Rigorously, this probability is:

$$\mathbb{P}(p_t^{(E)} > p_t^{(C)}) = \int_0^1 \frac{x^{\alpha_t^{(E)}-1}(1-x)^{\beta_t^{(E)}-1}}{Beta(\alpha_t^{(E)}, \beta_t^{(E)})} \left(\int_0^x \frac{s^{\alpha_t^{(C)}-1}(1-s)^{\beta_t^{(C)}-1}}{Beta(\alpha_t^{(C)}, \beta_t^{(C)})}\, ds \right) dx.$$

  By integrating over the experimental treatment's PDF weighted by the control's CDF, this value encapsulates the overall confidence in the superiority of the experimental arm. As such, this component of the state aims to provide a probabilistic comparison that integrates the entire distributions, not just their means.

- $\rho_t = \frac{A_t^{(E)}}{A_t^{(E)}+A_t^{(C)}} = \frac{\sum_{s=1}^t n_s^{(E)}}{\sum_{s=1}^t \left(n_s^{(E)}+n_s^{(C)}\right)}$, where we define $A_t^{(E)}$ and $A_t^{(C)}$ as the cumulative number of patients assigned to the experimental treatment and control arms, respectively, up to time t. As such, ρ_t aims to capture the historic allocation ratio, i.e., the proportion of all patients (across every cohort up to time t) that have been assigned to the experimental treatment arm. By reflecting on how balanced (or unbalanced) past allocation decisions have been, ρ_t helps when assessing any needs to correct for potential over- or under-allocation to a particular treatment arm. This is a key consideration within a clinical trial, as highly skewed allocations can negatively impact overall ethical and statistical objectives.

To briefly summarize, the goal of defining the state space as above is to offer the RL agent a comprehensive snapshot of the trial's status. It captures the clinical trial's progress via a normalized time indicator while concurrently summarizing our current beliefs about the treatment outcomes through both the posterior means and uncertainties derived from the Beta distributions. The probability that the experimental treatment is superior, η_t, gives a holistic comparison of the two treatment arms that goes beyond merely comparing point estimates. Then lastly, the historical allocation ratio succinctly reflects the trial's past patient assignments, essentially contextualizing the degree of exploration or exploitation that has been applied to each arm. In the clinical trial setting, these elements are structured to work together to help balance patient benefit against the necessity to learn from ongoing data.

In terms of justifying the MDP validity of the state space defined above, first note that a state x_t is Markov if and only if $\mathbb{P}(x_{t+1} \mid x_t) = \mathbb{P}(x_{t+1} \mid x_1, \ldots, x_t)$ is satisfied [31]. Thus, the state representation presented above satisfies the Markov property by design. All relevant historical information — encoded in the Beta parameters and the cumulative allocation ratio — is summarized in the current state x_t. Consequently, given x_t, the future evolution of the trial (i.e., the next state and received reward) depends only on the current state and the chosen allocation action, not on the detailed sequence of past events. This key Markov property ensures that the environment's dynamics are memoryless, meaning the next state can be determined solely based on the current state. This is essential for applying standard RL algorithms, which assume that the current state contains all information needed to predict future outcomes [31].

2.2.2 Action Space ($\mathcal{A}$)

The action space is defined on a full-cohort basis, as a continuous value $a_t \in [0, 1]$. This value represents the proportion of the total number of patients in the cohort N to be allocated to the experimental treatment arm. Specifically, the action space is defined as:

$$\mathcal{A} = \{a \in \mathbb{R} \mid 0 \leq a \leq 1\}.$$

This proportion is then mapped to a discrete allocation for the experimental treatment. We are able to define a continuous action space because the RL agent used is designed to output a continuous action. This allows for fine-grained and smooth interpolation over the set of possible allocation ratios. Such a design is especially beneficial in the clinical trial setting with a large and continuous decision space, as it enables the agent to explore subtle variations in policy without the limitations imposed by a discretized action space. Although we initially considered a discretized set of percentage allocations (e.g., 25%, 50%, 75%, etc.) mainly to reduce the computational burden, we transitioned to a continuous action space for two main reasons:

1. The RL algorithm we employ (a variant of Soft Actor-Critic) naturally handles a continuous action space.

2. Restricting the possible allocations to a fixed set hindered optimal decision-making and the policy's overall performance.

To illustrate the action space in the context of the trial, at time t, after observing the N patient responses from the previous cohort (assigned at time $t-1$), the agent selects an action a_t. This action is interpreted as the proportion of N patients to allocate to the experimental arm. Specifically, the number of patients allocated to the experimental arm is:

$$n_t^{(E)} = a_t \cdot N,$$

and accordingly, $n_t^{(C)} = N - n_t^{(E)}$ patients are allocated to the control arm. These allocations are implemented at time t, and the cycle continues with new outcome observations at time $t+1$. Naturally, this observation-allocation cycle continues until the conclusion of the trial at time T.

The percentage-based formulation decouples the policy decision from the absolute cohort size, yielding several important advantages. First, it enables scalability, in the sense that the same decision-making process can be applied regardless of cohort size N since decisions are expressed in relative terms. Second, it enables smoother optimization, meaning small changes in the action cause only small, incremental adjustments in the number of patients allocated to the treatment arms. We note this, as the previously defined discrete action space meant that if a different allocation percentage was selected from the set (such as 25%, 50%, and 75% as mentioned before), then the allocations would jump significantly, especially as N grew. This "smooth" aspect is namely important as it is conducive to efficient policy optimization through gradient-based methods, facilitating precise adjustment of the exploration-exploitation trade-off. Third and last note here, the continuous definition aids with interpretability, as the output of the RL agent can be directly interpreted as a proportion. Hence, doing so makes it easier to understand and communicate the precise allocation strategy.

2.2.3 Transition Dynamics

At each discrete time step t (with $t = 1, 2, \ldots, T$ for a finite-horizon trial), the overall state of the trial is represented by

$$x_t = \Big(t_{\text{norm}},\, \overline{p}_t^{(E)},\, \overline{p}_t^{(C)},\, \sigma_t^{(E)},\, \sigma_t^{(C)},\, \eta_t,\, \rho_t\Big),$$

which we defined in the state subsection above.

The transition dynamics under our MDP framework proceed in the following sequential steps:

1. Action Mapping to Patient Allocation: At time t, a continuous action $a_t \in [0, 1]$ is chosen. This action represents the proportion of the current cohort (of size N) to be assigned to the experimental arm. We define the discrete number of patients allocated to the experimental arm by

$$n_t^{(E)} = \left\lfloor a_t \cdot N + \frac{1}{2} \right\rfloor,$$

and the resulting allocation for the control arm is:

$$n_t^{(C)} = N - n_t^{(E)}.$$

We then trivially update the cumulative allocation histories $A_t^{(E)}$ and $A_t^{(C)}$ as follows:

$$A_t^{(E)} = A_{t-1}^{(E)} + n_t^{(E)}, \quad A_t^{(C)} = A_{t-1}^{(C)} + n_t^{(C)}.$$

For completeness, note that allocation histories are initialized to 0 at the start of the trial: $A_0^{(E)} = A_0^{(C)} = 0$.

2. Stochastic Outcome Realization: In the model, outcomes are realized in one of three ways:

- **True Probabilities are Established and Used for Outcome Generation and Benchmarking:** In this setting, we assume there exist some underlying true success probabilities for both treatment arms that are **unknown** to the decision-maker (and in practice to everyone) until the trial is completed at time T. These true probabilities are used as the success probabilities for running independent Bernoulli trials to simulate outcomes. Conveniently, this allows us to establish an upper bound for the optimal allocation strategy, where we establish an "oracle" strategy that knows these true success probabilities throughout the trial and allocates patients accordingly.

- **True Probabilities are Fully Unknown:** In this second setting, we resort to using our estimates of success probabilities to generate outcomes for the independent Bernoulli trials. Specifically, we use the posterior means calculated from the Beta distributions of each treatment arm as plug-in estimates for the Bernoulli likelihood.

- **Real Outcomes from Clinical Trial:** In this last case, we establish the setting where patient outcomes are actually observed. This setting is very idealistic, as running a full clinical trial is unfortunately out of the scope of this thesis, however, it's worth noting how this model would theoretically update in this scenario. Inherently, if we are able to observe actual outcomes of treatments, then we would not need to use independent Bernoulli to generate outcomes. The posteriors would simply update based on these observed outcomes, enabling allocation decisions to be made based on the updated posteriors following the realized outcomes.

In the second case where we don't have the underlying true probability for generating outcomes, the $n_t^{(E)}$ patients assigned to the experimental treatment arm in period t generate the outcomes using Bernoulli trials as explained. Namely, the success probability used for the experimental treatment arm is the previous period's posterior, $p_{t-1}^{(E)}$, and similarly $p_{t-1}^{(C)}$ for the $n_t^{(C)}$ patients in the control arm. with $p_{t-1}^{(C)}$. Note that here at time t, these probabilities are estimated by the posteriors at $t-1$:

$$p_{t-1}^{(E)} = \frac{\alpha_{t-1}^{(E)}}{\alpha_{t-1}^{(E)} + \beta_{t-1}^{(E)}}, \qquad p_{t-1}^{(C)} = \frac{\alpha_{t-1}^{(C)}}{\alpha_{t-1}^{(C)} + \beta_{t-1}^{(C)}}.$$

As such, we have that $r_t^{(E)}$ and $r_t^{(C)}$ denote the number of successes observed in the experimental and control arms, respectively. Thus,

$$r_t^{(E)} \sim \text{Binomial}\big(n_t^{(E)}, p_{t-1}^{(E)}\big), \qquad r_t^{(C)} \sim \text{Binomial}\big(n_t^{(C)}, p_{t-1}^{(C)}\big).$$

These random variables capture the trial's inherent outcome variability.

3. Bayesian Update of Latent Parameters: Using the conjugate relationship between the Beta distribution and the Binomial likelihood, we know the latent parameters are updated as follows:

$$\alpha_t^{(E)} = \alpha_{t-1}^{(E)} + r_t^{(E)}, \quad \beta_t^{(E)} = \beta_{t-1}^{(E)} + n_t^{(E)} - r_t^{(E)},$$

$$\alpha_t^{(C)} = \alpha_{t-1}^{(C)} + r_t^{(C)}, \quad \beta_t^{(C)} = \beta_{t-1}^{(C)} + n_t^{(C)} - r_t^{(C)}.$$

These updated parameters now represent the new posterior distributions for the treatment success probabilities.

4. State Recalculation: With the updated Beta parameters and cumulative allocation histories, the state is recalculated for the next decision time step of the trial $(t+1)$ as:

- Updated Posterior Means:

$$\bar{p}_t^{(E)} = \frac{\alpha_t^{(E)}}{\alpha_t^{(E)} + \beta_t^{(E)}}, \quad \bar{p}_t^{(C)} = \frac{\alpha_t^{(C)}}{\alpha_t^{(C)} + \beta_t^{(C)}}.$$

- Updated Posterior Standard Deviations (Uncertainties):

$$\sigma_t^{(E)} = \sqrt{\frac{\alpha_t^{(E)}\,\beta_t^{(E)}}{\left(\alpha_t^{(E)} + \beta_t^{(E)}\right)^2 \left(\alpha_t^{(E)} + \beta_t^{(E)} + 1\right)}},$$

$$\sigma_t^{(C)} = \sqrt{\frac{\alpha_t^{(C)}\,\beta_t^{(C)}}{\left(\alpha_t^{(C)} + \beta_t^{(C)}\right)^2 \left(\alpha_t^{(C)} + \beta_t^{(C)} + 1\right)}}.$$

- Recalculated Probability of Superiority:

$$\eta_t = \mathbb{P}\big(p_t^{(E)} > p_t^{(C)}\big) = \int_0^1 f_t^{(E)}(x)\, F_t^{(C)}(x)\, dx,$$

  with $f_t^{(E)}(x)$ and $F_t^{(C)}(x)$ defined as before.

- Updated Allocation Ratio:

$$\rho_t = \frac{A_t^{(E)}}{A_t^{(E)} + A_t^{(C)}}.$$

- Updated Normalized Time:

$$t_{\text{norm}} = \frac{t}{T}$$

5. Stopping Conditions: We lastly highlight the termination step, noting that the transition process repeats at each decision step until $t = T$. Hence, at termination, the final state is:

$$x_T = \Big(1,\, \bar{p}_T^{(E)},\, \bar{p}_T^{(C)},\, \sigma_T^{(E)},\, \sigma_T^{(C)},\, \eta_T,\, \rho_T\Big).$$

Formally, if we denote by $\mathbb{P}_{\text{alloc}}(n_t^{(E)} \mid a_t)$ the deterministic probability mass function corresponding to the mapping $\lfloor a_t \cdot N + \frac{1}{2} \rfloor$, and by $\mathbb{P}_{\text{obs}}(r_t^{(E)}, r_t^{(C)} \mid x_{t-1}, n_t^{(E)})$ the joint probability of observing outcomes $r_t^{(E)}$ and $r_t^{(C)}$ given the previous state and patient counts, then the overall transition kernel is given by the following:

$$\mathbb{P}(x_t \mid x_{t-1}, a_t) = \mathbb{P}_{\text{alloc}}(n_t^{(E)} \mid a_t) \cdot \mathbb{P}_{\text{obs}}(r_t^{(E)}, r_t^{(C)} \mid x_{t-1}, n_t^{(E)})\,.$$

This product factorization reflects the fact that (1) given the action a_t, the allocation to the experimental arm is deterministically fixed, and (2) once patient counts are assigned, the observations are generated stochastically via independent Binomial processes. Hence, the overall one-step state update can be thought of as being composed of a deterministic mapping (from action to patient allocation) followed by a stochastic outcome generation process, which together updates the state according to the Bayesian updating rules.

To mention the Markov property in our MDP framework, we importantly highlight that the state vector x_t fully encapsulates all the necessary historical information required to predict future outcomes. In our formulation, the state at time t of each component is derived from all previous outcomes and allocations. Specifically:

- The posterior means $\overline{p}_t^{(E)}$ and $\overline{p}_t^{(C)}$ are computed from the Beta parameters $\alpha_t^{(E)}$, $\beta_t^{(E)}$, $\alpha_t^{(C)}$, and $\beta_t^{(C)}$ which have been updated using all past observed outcomes.

- The uncertainties $\sigma_t^{(E)}$ and $\sigma_t^{(C)}$ provide a measure of the current estimation precision based on the updated Beta parameters.

- The probability of superiority η_t is derived from the full distributions of $p_t^{(E)}$ and $p_t^{(C)}$.

- The cumulative allocation ratio ρ_t summarizes all past allocation decisions.

Because these state components are updated at every time step using the outcomes and decisions from the previous step, the updated state contains all relevant information from the past. Formally, the transition probability

$$\mathbb{P}(x_t \mid x_{t-1}, a_t)$$

captures the entire effect of the history through x_{t-1}. Explicitly, once the state x_{t-1} is known, the distribution over x_t depends solely on the current action a_t and the probabilistic mechanism for outcome generation and Bayesian updating. As such, our state representation satisfies the following Markov property:

$$\mathbb{P}(x_{t+1} \mid x_t, a_t) = \mathbb{P}(x_{t+1} \mid x_1, x_2, \ldots, x_t, a_t),$$

meaning that the future state of the system depends only on the present state (and the current action) and not directly on the sequence of states and actions that preceded it. This "memoryless" characteristic is essential for the standard application of reinforcement learning algorithms, ensuring that all information necessary for optimal decision-making is contained in x_t.

2.2.4 Reward Function ($R(x_t, a_t, x_{t+1})$)

The reward at time step t is designed to capture not only the immediate benefit in patient outcomes but also the value of information acquired and the degree to which the allocation decisions promote balanced exploration of the treatments. Formally, we define the reward function as:

$$R_t = R_{\text{immediate}} + \lambda_{\text{TVD}} \cdot R_{\text{information}} + \lambda_{\text{explore}} \cdot R_{\text{explore}}, \tag{2.3}$$

where:

- $R_{\text{immediate}}$ is the immediate reward, equal to the total number of clinical successes observed in the current cohort.
- $R_{\text{information}}$ quantifies the information gained through a change in the separation of the success probability posteriors, as measured by the Total Variation Distance (TVD).
- R_{explore} is an exploration bonus that incentivizes balanced allocations between treatment arms, with a decaying weight as the trial progresses.

- λ_{TVD} and λ_{explore} are hyperparameters that regulate the trade-off between immediate outcomes, information gathering, and exploration.

Immediate Reward: We define the immediate reward as the total number of observed patient successes at time t. If we denote

$$r_t^{(E)} = \mathbf{1}^\top \mathbf{y}_t^{(E)} \quad \text{and} \quad r_t^{(C)} = \mathbf{1}^\top \mathbf{y}_t^{(C)}$$

as the observed successes in the experimental and control arms respectively, then:

$$R_{\text{immediate}} = r_t^{(E)} + r_t^{(C)}.$$

This term directly measures patient benefit and is analogous to the clinical outcome of interest. It forms the core objective of any clinical trial: to maximize the number of patients receiving effective treatment.

Information Gain Bonus: To quantify the value of information gathered between decision points, we use the Total Variation Distance (TVD) between the Beta posterior distributions of the two treatment arms. The TVD between two probability distributions is a canonical measure of dissimilarity that plays an essential role in many areas of probability, statistics, and machine learning [6]. For two distributions P and Q defined on a common measurable space with density functions $p(x)$ and $q(x)$, the TV distance is defined as:

$$\text{TVD}(P, Q) = \frac{1}{2} \int |p(x) - q(x)| \, dx.$$

Alternatively, it can be expressed as the maximum difference between the probabilities assigned to any measurable set:

$$\mathrm{TVD}(P,Q) = \max_{S \subset \mathcal{X}} |P(S) - Q(S)|\,.$$

This formulation highlights its physical interpretation: it represents the maximum bias that can be induced by using one distribution instead of the other over any event S. In this sense, TVD tells us the worst-case error one might incur if decisions were based on a distribution that differs from the true one.

TVD is especially attractive due to several mathematically desirable characteristics, highlighted by Bhattacharyya et al. (2022):

1. **Boundedness:** TVD is bounded between 0 and 1. This means that for any two probability distributions, $\mathrm{TVD}(P,Q) \in [0,1]$. A TVD of 0 indicates identical distributions, while a TVD of 1 (or very close to 1) indicates complete dissimilarity.

2. **Metric Properties:** TVD is a proper metric on the space of probability distributions. It satisfies the usual metric axioms: non-negativity, symmetry, triangle inequality, and $\mathrm{TVD}(P,Q) = 0$ if and only if $P = Q$.

3. **Invariance under Bijections:** TVD remains unchanged under any bijective transformation of the sample space. That is, if f is a bijection, then:

$$\mathrm{TVD}(P,Q) = \mathrm{TVD}(P \circ f^{-1}, Q \circ f^{-1}).$$

   This invariance property is very useful when the underlying space is transformed, ensuring that the measure of dissimilarity does not depend on the particular representation of the data.

In our model, TVD is used to quantify the difference between the posterior distributions of the experimental and control treatment arms. Consider the two Beta distributions:

$$P_E(x) = \frac{x^{\alpha_E-1}(1-x)^{\beta_E-1}}{\text{Beta}(\alpha_E, \beta_E)} \quad \text{and} \quad P_C(x) = \frac{x^{\alpha_C-1}(1-x)^{\beta_C-1}}{\text{Beta}(\alpha_C, \beta_C)},$$

which represent our beliefs about the success probabilities for arms E and C, respectively. The TVD between these two distributions is then given by:

$$\text{TVD}(P_E, P_C) = \frac{1}{2}\int_0^1 |P_E(x) - P_C(x)|\, dx.$$

In the context of the adaptive trial, we are less interested in the absolute TVD at each time step and more in its change between time steps. If new patient outcomes cause the TVD to increase, it indicates that the posterior distributions are diverging from one another; put differently, the new data have increased our ability to distinguish between the efficacy of the experimental and control arms. Formally, we define the information gain component of the reward as:

$$R_{\text{information}} = \text{TVD}_{\text{new}} - \text{TVD}_{\text{old}},$$

where a positive difference suggests that the experimenter has acquired informative evidence that could guide future allocation decisions.

The selection of TVD to measure information gain is motivated by several factors:

- **Physical Interpretation:** TVD quantifies the maximum possible difference in probabilities across all events. In our clinical trial context, it provides a worst-case measure of bias between the two treatment distributions. If TVD increases, it means there is a stronger signal in favor of one treatment over the other.

- **Mathematical Robustness:** The TV distance is a metric, ensuring that our measurement of information gain is both reliable and consistent. Its bounded nature guarantees that the contribution to the reward from the TVD term is well-controlled.

- **Invariance and Applicability:** TVD's invariance under bijections implies that its value is independent of how the outcome space is represented. This is particularly useful when working with continuous Beta distributions, as their functional forms remain comparable under natural transformations.

In practice, to compute this integral numerically, we discretize the interval $[0, 1]$ into n evenly spaced grid points $\{x_i\}_{i=1}^n$ with $\Delta x = 1/n$, thereby approximating:

$$\text{TVD}(P_E, P_C) \approx \frac{1}{2} \sum_{i=1}^{n} |p_E(x_i) - p_C(x_i)| \, \Delta x.$$

Let TVD_{t-1} denote the TVD at the previous decision step and TVD_t the TVD after updating the posteriors at time t. The information gain is then computed as:

$$R_{\text{information}} = \text{TVD}_t - \text{TVD}_{t-1}.$$

A positive value for $R_{\text{information}}$ indicates that the new data have increased the separation between the experimental and control distributions, implying that additional information has been acquired about the relative efficacy of the treatments. This bonus component is critical for steering the agent toward decisions that favor learning—thus improving future allocation decisions—especially when clinical differences are subtle.

Exploration Bonus: Balanced patient allocation between treatment arms is essential for ensuring robust estimation of treatment effects. To promote this balance, the exploration bonus is defined as:

$$R_{\text{explore}} = \left(\frac{\min\{n_t^{(E)},\, n_t^{(C)}\}}{\max\{n_t^{(E)},\, n_t^{(C)}\}} \right) \cdot \left(1 - \frac{t}{T} \right).$$

The fraction $\frac{\min\{n_t^{(E)}, n_t^{(C)}\}}{\max\{n_t^{(E)}, n_t^{(C)}\}}$ is a measure of balance, attaining a value of 1 when allocations are perfectly balanced and values closer to 0 when they are skewed. The multiplicative factor $1 - \frac{t}{T}$ introduces a decay over time, reflecting the intuition that early in the trial it is particularly important to gather balanced information, while toward the end, exploiting the accumulated knowledge becomes more critical.

Final Reward Function: Combining all components, the final reward function at time t is given as:

$$R_t = \underbrace{(r_t^{(E)} + r_t^{(C)})}_{R_{\text{immediate}}} + \lambda_{\text{TVD}} \cdot \underbrace{(\text{TVD}_t - \text{TVD}_{t-1})}_{R_{\text{information}}} + \lambda_{\text{explore}} \cdot \underbrace{\left[\left(\frac{\min\{n_t^{(E)}, n_t^{(C)}\}}{\max\{n_t^{(E)}, n_t^{(C)}\}} \right) \cdot \left(1 - \frac{t}{T} \right) \right]}_{R_{\text{explore}}}.$$

In this formulation:

1. $R_{\text{immediate}}$ ensures that the model directly rewards the number of patients who experience clinical success, aligning with the primary clinical objective.

2. The term $\lambda_{\text{TVD}} \cdot R_{\text{information}}$ encourages the agent to favor policies leading to state updates where the posterior distributions become more distinct. Such a separation suggests that the data is informative enough to discern treatment efficacy differences, which is paramount for adapting allocations effectively.

3. The exploration bonus, scaled by λ_{explore}, promotes a balanced allocation among treatment arms, especially during the early phases of the trial. This balance mitigates the risk of prematurely favoring one arm and ensures that both treatments are sufficiently explored to attain reliable posterior estimates.

The hyperparameters λ_{TVD} and λ_{explore} control the relative importance of the long-term information-acquisition and balanced allocation objectives concerning the immediate clinical successes. Tuning these parameters allows the decision-maker to navigate the trade-off between maximizing current patient benefits and investing in learning that improves subsequent decisions—a key feature in adaptive response designs.

Chapter 3

Soft Actor-Critic for Policy Optimization in Adaptive Trials

To solve the above MDP outlined in Chapter 2, we employ a deep reinforcement learning algorithm: Soft Actor-Critic (SAC). SAC is an off-policy actor-critic method that maximizes a trade-off between return and entropy. We chose a SAC algorithm for its ability to handle continuous action spaces and for its robust exploration via entropy regularization. Haarnoja et al. (2018), who developed the algorithm, found that SAC achieves state-of-the-art performance when applied to real-world tasks, specifically when compared to other off-policy, model-free deep reinforcement learning (RL) algorithms that exhibit high sample complexity and are brittle to hyperparameters [15]. As such, we chose SAC as is well-suited for the high-stakes, data-constrained setting of clinical trials. In this section, we provide a deeper overview of SAC's components, including entropy regularization, twin critic networks, reparameterization trick, and automatic temperature tuning. After establishing the overarching algorithm structure, we contextualize why they are beneficial for an adaptive clinical trial setting, and relate SAC's theoretical motivations to our application.

The components of SAC and its specific use case and context are detailed in this chapter. However, to provide some initial clarity, consider the diagram below (adapted from [9]), which we provide to help visualize the overall flow of the algorithm.

At a high level, the *Trial Environment* (grey box) continuously simulates the clinical trial by generating new patients—this represents the current state—and by presenting several treatment arms available for assignment. An arrow labeled "State" carries this information to the *Actor* (purple box), which uses a deep neural network to process the state and output an action (i.e., the treatment arm to assign).

Once the action is selected, it flows from the Actor (with an accompanying notation indicating stochastic sampling via a normal distribution) back to the Trial Environment. The environment then simulates the outcome based on that action, producing a reward (e.g., reflecting patient response and exploration bonuses) and a next state. These pieces of information—the initial state, the reward, the action taken, and the resulting next state—are grouped into a transition tuple (State, Reward, Action, Next State).

This complete transition is then recorded in the *Replay Buffer*, which stores past experiences. The Replay Buffer serves as the source for sampling mini-batches used to train the Actor and the twin Critic networks (labeled as Q_1 and Q_2). The Critics use these samples to compute temporal-difference (TD) losses and update their value estimates, while the Actor is updated using feedback from the Critics (including the entropy-regularized loss) to improve its policy. The cyclic flow—from environment to actor to environment, storing transitions, and then updating the networks—forms the core of our SAC framework in the clinical trial context.

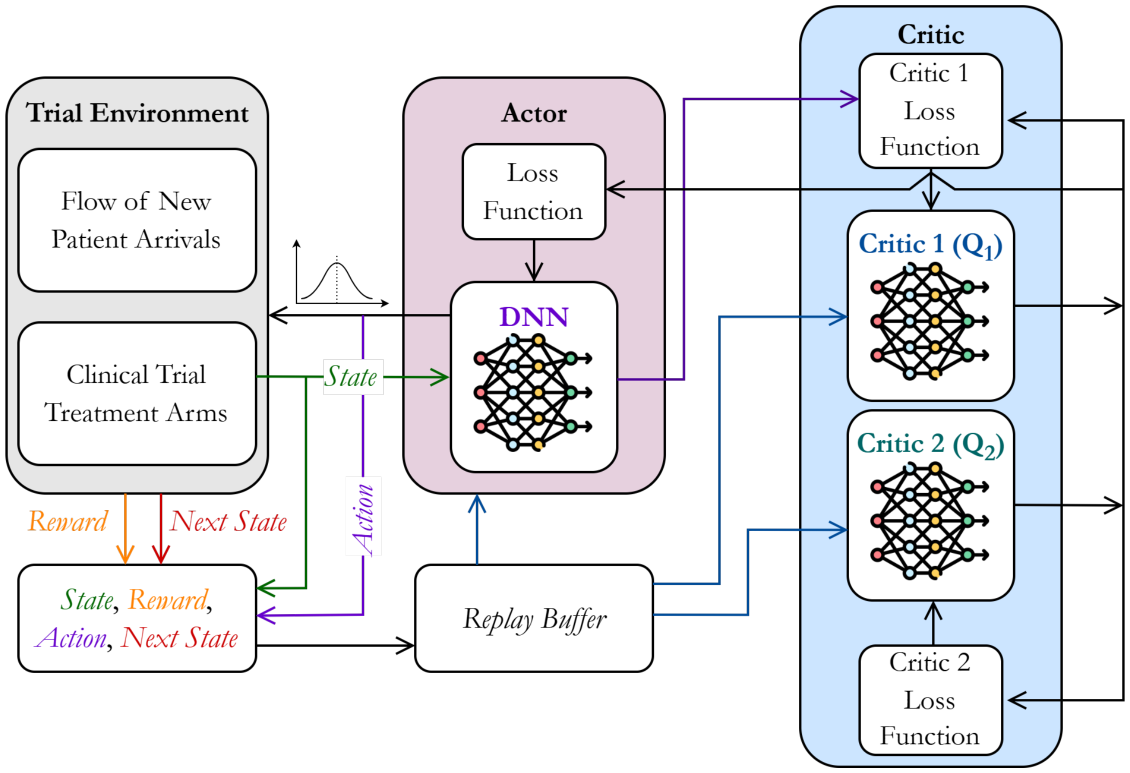

Figure 3.1: SAC Diagram

3.1 Entropy-Regularized Objective for Efficient Exploration

Unlike standard RL algorithms that maximize only the expected return, SAC augments the objective with an entropy term [29]. Entropy $H(\pi(\cdot \mid x))$ is a measure of randomness in the policy's action selection. SAC's actor is trained to maximize a weighted sum of expected reward and policy entropy. Haarnoja et al. (2018) illustrate this, highlighting that the actor seeks to "succeed at the task while acting as randomly as possible" [14]. This is formalized by the maximum entropy RL objective:

$$J_\pi = \mathbb{E}_{x_t \sim \rho^\pi,\, a_t \sim \pi}\Big[\sum_{t=0}^{T-1} \gamma^t \big(r(x_t, a_t) + \alpha\, \mathcal{H}(\pi(\cdot \mid x_t))\big)\Big], \tag{3.1}$$

where $\mathcal{H}(\pi(\cdot \mid x)) = -\mathbb{E}_{a\sim\pi(\cdot|x)}[\log \pi(a|x)]$ is the entropy and α is a temperature parameter that balances the importance of the entropy term relative to the reward. SAC typically considers an infinite-horizon discounted setting, but the intuition applies to our finite-horizon discretized trial case, where we effectively treat $\gamma \approx 1$ for episode return. By injecting entropy into the objective, SAC encourages exploration. As explained briefly in Chapter Two, previous methods explored were heavily discretized to reduce computational complexity, whereas with SAC the policy gets higher objective values by being stochastic. The main point is that this prevents premature convergence to a deterministic policy that might be suboptimal.

In the context of an adaptive trial, this entropy bonus is extremely valuable. It ensures that the allocation policy $\pi(a \mid x)$ doesn't collapse too quickly to always assigning one arm, which could happen if the agent gets a few good outcomes early for one treatment. Instead, SAC's policy is incentivized to maintain diversity in actions, meaning it keeps some randomness in patient assignment. This built-in exploration mechanism complements our intrinsic reward shaping. Even if the reward shaping from the information gain and exploration bonuses wanes later in the trial, SAC's entropy term continues to encourage trying alternative allocations, aiming to reduce the risk of getting stuck in a suboptimal allocation strategy.

From a theoretical standpoint, maximum entropy RL can be seen as a way of regularizing the policy search so that the agent considers a wide range of strategies. This is critical in a domain like clinical trials where over-committing to one arm based on inadequate data can have ethical and scientific consequences, as discussed in other chapters. SAC's entropy term can be viewed as an automated form of "exploration noise" that persists throughout training. Indeed, Haarnoja et al. report that this approach leads to more stable and reliable learning, preventing the policy from converging to a poor local optimum prematurely [14]. In our experiments, this translated to the agent continuing to allocate some patients to the worse-performing arm if there remained uncertainty, rather than fully exploiting in a greedy manner too early.

What's nice about how the algorithm is set up, is that you can adjust how much randomness SAC aims for with the α temperature (more on that below), effectively setting a desired target entropy (e.g., one might set it to $-\dim(\mathcal{A})$ as a heuristic). In our continuous one-dimensional action space, a typical target entropy might be a modest negative value (since a uniform random policy on [0,1] has a certain entropy). By tuning α, we can control the exploration-exploitation balance: a higher α encourages more exploration (specifically, think stochasticity), and a lower α makes the policy put more weight on reward maximization, which leads to a greedier and more deterministic behavior. Conveniently, Haarnoja et al. (2018) devised an automatic entropy tuning mechanism for SAC to adjust α so that the observed entropy of the policy approaches a target level (more on this as well below). This is particularly useful in our setting because the optimal amount of exploration may change over the course of training or differ between phases of the trial. Thus, being able to allow the algorithm to self-tune the exploration level is quite convenient and reduces the need for extensive manual hyperparameter guessing.

3.2 Twin Critic Networks and Stabilized Learning

SAC is an actor-critic algorithm, which means it uses function approximators for both the policy (actor) and the value function (critic). In fact, SAC employs two Q-value networks (critics) in parallel, which is why it's referred to as twin critics. Each critic $Q_{\phi_1}(x, a)$ and $Q_{\phi_2}(x, a)$ estimates the expected return (cumulative reward) from state x after taking action a and thereafter following the policy. Why two critics? This design is inspired by the *clipped Double Q-learning* trick introduced by Fujimoto et al. (2018) in the TD3 algorithm, which SAC incorporates for continuous control [11].

The issue it addresses is overestimation bias in value function learning. Specifically, with function approximation and noisy updates, a single critic can produce over-optimistic Q-value estimates, which in turn can lead the actor astray, since the actor tries to maximize the Q. By training two independent critics and using the minimum of their predictions for key computations (such as in the actor update or in forming the target values), SAC ensures a more conservative estimate of value. Essentially, the pessimistic bound (minimum of two Qs) reduces the likelihood of overestimating the returns of any state-action, thereby improving training stability.

In practice, the twin critics in SAC are trained with standard temporal-difference (TD) learning. They minimize a bellman error loss concerning a target value that involves the next state's value [12]. SAC's critic loss for each Q_{ϕ_i} is as follows:

$$J_Q(\phi_i) = \mathbb{E}_{(x_t, a_t, r_t, x_{t+1})}\Big[\big(Q_{\phi_i}(x_t, a_t) - y_t\big)^2\Big], \tag{3.2}$$

where the target $y_t = r_t + \gamma\big(\min_{j=1,2} Q_{\bar{\phi}_j}(x_{t+1}, a'_{t+1}) - \alpha \log \pi(a'_{t+1} \mid x_{t+1})\big)$. Here $a'_{t+1} \sim \pi(\cdot \mid x_{t+1})$ is a sample action from the current policy at x_{t+1}, and $\bar{\phi}_j$ indicates the parameters of a target network, which is an exponential moving average of past critic parameters for stability. Specifically, this target y_t uses the minimum of the two target Q-networks and also subtracts the entropy term, since the actor's objective includes entropy. In taking the minimum, if one critic inadvertently outputs an overestimated value, the other (typically lower) critic's value will be used, preventing an inflated target [11]. This technique has been shown to significantly stabilize training in continuous control tasks. In a clinical trial setting, it should go without saying that data (1) tends to be highly scarce, and (2) can be highly variable from patient to patient. Consequently, function approximation could easily overshoot due to noise, so the twin-critic approach is quite helpful by providing a form of built-in regularization of the value estimates.

Stability in value estimation is important for clinical trial RL because erratic or divergent critical estimates could lead to unsafe or highly suboptimal policies. Zhao et al. (2009) who developed reinforcement learning trials for discovering individualized treatment regimens using Q-learning, highlight that ”estimating the value function is the most important component” [39]. With twin critics, SAC achieved more stable learning, evidenced by the low variance achieved in performance across random seeds. Particularly, Haarnoja et al. note that SAC, with these improvements, is very stable, achieving similar performance across different runs [14]. This reliability is crucial, as in a high-stakes scenario we prefer an algorithm that we can trust to converge similarly each time (or in each simulation), rather than one that occasionally collapses or wildly oscillates.

3.2.1 Stochastic Policy and the Reparameterization Trick

The actor in SAC outputs a stochastic policy $\pi(a \mid x)$, modeled by a neural network that produces the parameters of a Gaussian distribution (i.e., mean and variance) from which actions are sampled. In our implementation, the action is one-dimensional and bounded to the interval $[0, 1]$. To ensure that the unconstrained Gaussian sample is mapped to this interval, we apply a squashing function. While some SAC implementations use a tanh function (possibly followed by a shift/scale transformation), our implementation employs a logistic sigmoid function directly because it naturally outputs in $[0, 1]$ and is simple to differentiate [34]. Specifically, the network first produces an unconstrained Gaussian sample via the reparameterization trick. That is, if the policy outputs mean $\mu(x)$ and standard deviation $\sigma(x)$ (which, contextually, are its current beliefs about what the best allocation should be, given the current state x), we sample:

$$\epsilon \sim \mathcal{N}(0, 1)$$

which introduces stochasticity, and then computes the latent variable:

$$\tilde{a} = \mu(x) + \sigma(x)\,\epsilon.$$

As mentioned, to ensure the action lies in $[0, 1]$, we just apply the logistic sigmoid:

$$a = \sigma(\tilde{a}) = \frac{1}{1 + e^{-\tilde{a}}}.$$

This entire process defines a deterministic mapping:

$$a = f_\theta(x, \epsilon) = \sigma\big(\mu(x) + \sigma(x)\,\epsilon\big), \tag{3.3}$$

with θ denoting the policy parameters.

Importantly, when applying the squashing function, the probability density must be corrected via the change-of-variables formula. If we define the pre-squashed variable as $\tilde{a}$ and note that $a = \sigma(\tilde{a})$ (with σ being the logistic sigmoid), the derivative of a with respect to $\tilde{a}$ is:

$$\frac{da}{d\tilde{a}} = a(1 - a).$$

Thus, the transformed density is:

$$\pi_\theta(a \mid x) = \mathcal{N}(\tilde{a}; \mu(x), \sigma(x)^2)\,\left|\frac{d\tilde{a}}{da}\right| = \mathcal{N}(\tilde{a}; \mu(x), \sigma(x)^2)\,\frac{1}{a(1 - a)}, \tag{3.4}$$

where $\tilde{a} = \sigma^{-1}(a)$. So consequently, we have that the log-likelihood becomes:

$$\log \pi_\theta(a \mid x) = \log \mathcal{N}(\tilde{a}; \mu(x), \sigma(x)^2) - \log\big(a(1 - a)\big). \tag{3.5}$$

A key component enabling efficient training of the policy is the reparameterization trick. Motivated by Haarnoja et al. (2018), instead of sampling actions directly from the Gaussian distribution parameterized by the policy — which would complicate gradient computation — we express the action as a deterministic function of the state, the policy parameters, and an independent noise source [14]. As a result, the expected value:

$$\mathbb{E}_{a\sim\pi}[Q(x,a)] = \mathbb{E}_{\epsilon\sim\mathcal{N}(0,1)}\Big[Q\big(x, f_\theta(x,\epsilon)\big)\Big] \tag{3.6}$$

allows us to push the gradient operator inside the expectation and differentiate with respect to θ via standard backpropagation.

Thus, for the entropy-regularized actor objective:

$$J_\pi = \mathbb{E}_{a\sim\pi_\theta}\Big[Q_{\min}(x,a) - \alpha \log \pi_\theta(a \mid x)\Big], \tag{3.7}$$

its gradient can be estimated as:

$$\begin{aligned}\nabla_\theta J_\pi &= \nabla_\theta \mathbb{E}_{a\sim\pi_\theta}[Q_{\min}(x,a) - \alpha \log \pi_\theta(a \mid x)] &(3.8)\\ &\approx \mathbb{E}_{\epsilon\sim\mathcal{N}(0,1)}\big[\nabla_\theta(Q_{\min}(x, f_\theta(x,\epsilon)) - \alpha \log \pi_\theta(f_\theta(x,\epsilon) \mid x))\big], &(3.9)\end{aligned}$$

where $Q_{\min}(x,a) = \min(Q_{\phi_1}(x,a), Q_{\phi_2}(x,a))$.

Note that the gradient estimate can be fully expanded using $f_\theta(x,\epsilon) = \sigma\big(\mu(x) + \sigma(x)\,\epsilon\big)$ (which we don't shown).

In the trial context, having a smoothly adjusting stochastic policy is beneficial. It means the agent can fine-tune the probability of allocation to the experimental arm as a continuous variable. Small changes in its neural network output lead to small changes in a_t, which is important for stable learning. The reparameterization trick ensures that the policy gradient algorithm can properly credit/blame the policy parameters for the outcomes observed, which is non-trivial because outcomes are noisy (Bernoulli) and the policy's effect is indirect (through probabilities). Thanks to reparameterization, SAC can handle this credit assignment through the chain rule, which likely contributes to its sample efficiency. It has been observed that reparameterized policy gradients improve learning speed in continuous domains, which aligns with our need to learn from relatively few trials. The results obtained when running simulations will be discussed in the following chapters, however, we note that the SAC agent was able to learn an allocation policy fairly effectively. This was done within a few thousand simulation episodes, using varying lengths of T.

Previous iterations of the model and overarching framework didn't really come close to converging on a policy, even after thousands of simulation episodes and many hours. That being said, the current implementation is still considerably beaten computationally by approaches like Ahuja and Birge (2020), which we've previously discussed. In their algorithm (SLAX), they employ a grid-based state discretization, minimal triangulation to partition this approximate state space into a finite number of convex regions, and barycentric interpolation to approximate the value function. SLAX then applies several techniques that allow them to run very computationally efficiently, including restriction of both the time horizon (using limited lookahead) and action set (allocating patients in blocks) [2]. Apart from SLAX, other algorithms, including traditional bandit algorithms, require analytical derivations or grid searches. Establishing an accurate grid in the context of clinical trials has to be done very carefully, and if mismodeled, can be very problematic. By using the SAC, we avoid this and take advantage of the fact that the algorithm learns by gradient descent.

3.2.2 Automatic Entropy Tuning (Temperature Auto-Adjustment)

An important hyperparameter in SAC is the entropy weight α mentioned earlier. Rather than fixing α, SAC often includes an automatic tuning mechanism to adjust α during training [15]. The idea is to set a target entropy (e.g. a desired average entropy for the policy) and then have α updated by gradient descent to minimize the difference between the current policy entropy and the target. Concretely, the temperature α is updated by minimizing the following loss:

$$J(\alpha) = \mathbb{E}_{(s_t,a_t)\sim\mathcal{D}}\Big[-\alpha\Big(\log \pi_\theta(a_t \mid s_t) + \mathcal{H}_{\text{target}}\Big)\Big], \tag{3.10}$$

where $\mathcal{D}$ represents the distribution of state–action pairs collected in the replay buffer, and $\mathcal{H}_{\text{target}}$ is the target entropy that we set. The gradient $\nabla_\alpha J(\alpha)$ is zero at the optimum when the average $\log \pi_\theta(a \mid s) + \mathcal{H}_{\text{target}}$ equals zero, meaning the policy's entropy matches the target. Thus by updating α in the direction that reduces this discrepancy, we can enable SAC to effectively self-tune the level of exploration.

This mechanism is leveraged in our implementation to avoid manually searching for the right α, which can get quite tedious when trying to tune other hyperparameters at the same time. That said, not having to tune α is especially useful in our domain because the optimal exploration rate is not known a priori. It's reasonably expected that more exploration will be needed early in the trial and presumably less later. As such, we tackle this in two ways, with the first being the R_{explore} term in the reward function, and the second the entropy tuning, which further adjusts the overall randomness of the policy during training. The goal here is to maintain a balanced exploration-exploitation trade-off, and overall we set up this model to err on the side of caution. However, what's nice about the auto-tuning of α is that if our reward shaping already encourages sufficient exploration and the observed policy entropy is above the target, then α will decrease. In turn, this reduces the extra penalty which then allows the policy to focus more on maximizing reward. Conversely, if the agent begins to exploit too heavily (i.e., the policy becomes too deterministic with entropy falling below the target), then α will increase in order to push the policy back toward exploration.

This adaptability makes SAC robust across different configurations. Haarnoja et al. (2018) extended SAC with this automatic entropy tuning mechanism and found that it significantly improved stability across hyperparameter choices [15]. In our training, we experimented with target entropy values of $[-2, -1, -0.5]$. As briefly mentioned before, existing literature commonly sets the target entropy to $-\dim(\mathcal{A})$ [15]. In doing so, the resulting policy can automatically learn to be near-deterministic in states where one treatment's posterior is overwhelmingly superior while remaining sufficiently stochastic in ambiguous states. This is all done without us having to explicitly code such behavior.

3.2.3 Curriculum and Trial-level Training

As we have developed, the overall system is formulated as an MDP where at each discrete time step, the clinical trial state is updated using Bayesian principles. The training process involves two nested loops:

1. A curriculum-learning loop that gradually increases the complexity of the environment.

2. A trial-level training loop that gathers trajectories of state transitions and performs gradient updates on the actor and critic networks.

Curriculum Learning and Environment Adaptation

Curriculum learning is a training strategy in which an agent is initially exposed to simplified or easier versions of a task before gradually progressing to more challenging variants. This deliberate ordering leverages the idea that early mastery of basic skills can serve as a foundation, thus reducing the need for extensive trial-and-error when dealing with complex scenarios later on. By structuring the learning process into phases of increasing difficulty, the agent can refine its policy incrementally. In turn, this leads to faster convergence and improved performance in the full-scale task that we ultimately want the agent to perform well on [22].

In the context of our clinical trial environment, the decision-making process is complex due to multiple interacting components:

- The environment evolves over a finite horizon T, and the agent must learn to make allocation decisions across multiple periods.
- The underlying model uses Bayesian updates to adjust beliefs about treatment success probabilities based on patient outcomes.
- The reward function incorporates immediate outcomes, information gain, and exploration incentives.

As such, if the agent is thrown into a full-scale problem with a long horizon and full complexity of the state space right from the start, the extensive state and action spaces and the inherent uncertainty may overwhelm the training process. Naturally, this can lead to unstable learning or convergence to suboptimal policies. This is amplified by the stochasticity of a clinical trial setting to begin with, and we saw this when previously exploring basic Q-learning approaches to determine the optimal policy.

To illustrate this, the expected return that the agent attempts to maximize is given by:

$$J(\pi) = \mathbb{E}\left[\sum_{t=0}^{T-1} \gamma^t R_t\right], \tag{3.11}$$

Where R_t is the composite reward. Curriculum learning starts the agent with a smaller effective T, so if we define by $T_{\text{phase}} = T_{\text{base}} \cdot T_{\text{factor}}$, where $T_{\text{factor}} < 1$, then the agent initially optimizes:

$$J_{\text{phase}}(\pi) = \mathbb{E}\left[\sum_{t=0}^{T_{\text{phase}}-1} \gamma^t R_t\right], \tag{3.12}$$

This reduced horizon simplifies the credit assignment problem and lowers the variance in estimates, making the RL updates more stable.

As such, the reward is adjusted in each phase by the adjustment of the weights for the information gain and exploration bonuses by a scaling factor k. Specifically, the environment's hyperparameters are updated as:

$$\lambda_{\text{TVD}}^{\text{phase}} = \lambda_{\text{TVD}}^{\text{base}} \cdot k, \tag{3.13}$$

$$\lambda_{\text{explore}}^{\text{phase}} = \lambda_{\text{explore}}^{\text{base}} \cdot \big(1 + (1 - T_{\text{factor}})\big) = \lambda_{\text{explore}}^{\text{base}} \cdot \big(2 - T_{\text{factor}}\big). \tag{3.14}$$

Trial-Level Training

Within each curriculum phase, training is conducted over several full clinical trials, which we can refer to as episodes. Specifically, for each episode e:

1. The environment is reset, setting $\alpha_0^{(E)}, \beta_0^{(E)}$ (and similar for the control arm) to their initial values and resets the cumulative counters.

2. An action is selected, where for each time step t in the episode (with maximum T_{phase}:

   - The agent selects an action a_t using its policy network: $a_t \sim \pi_\theta(\cdot \mid x_t$. This continuous action is then mapped to a discrete allocation.

   - The environment simulates outcomes for both arms using an underlying $p_t^{(E)}$ and $p_t^{(C)}$, and subsequently the successes are sampled:

   $$r_t^{(E)} \sim \text{Binomial}(n_t^{(E)}, p_t^{(E)}), \quad r_t^{(C)} \sim \text{Binomial}(n_t^{(C)}, p_t^{(C)}).$$

   - The Beta parameters are updated using the update previously discussed.

   - The total reward is computed

3. The updates are based on mini-batches sampled uniformly from the replay buffer. The training updates follow the SAC methodology:

- ***Critic Update:*** For a mini-batch $\{(s_i, a_i, r_i, s'_i, d_i)\}_{i=1}^{B}$, the target for the critic is computed as:

$$y_i = r_i + \gamma\,(1 - d_i)\left(\min_{j=1,2} Q_{\text{target},j}(x'_i, a'_i) - \alpha\,\log \pi(a'_i \mid x'_i)\right), \tag{3.15}$$

where a'_i is sampled from the actor's policy for state x'_i (using the reparameterization trick), $Q_{\text{target},j}$ are the target critic networks, γ is the discount factor, α is the temperature parameter governing the entropy bonus, and d_i is just a "done" flag. From here, the critic loss for each critic j is computed as:

$$L_{Q_j} = \frac{1}{B}\sum_{i=1}^{B}\left(Q_j(x_i, a_i) - y_i\right)^2, \tag{3.16}$$

And finally, gradient descent is applied to minimize L_{Q_j}.

- ***Actor Update:*** The actor's objective is to maximize the expected return while also maximizing the policy's entropy. This can be written as:

$$J_\pi = \mathbb{E}_{x\sim D,\,a\sim\pi}\Big[Q_{\min}(x, a) - \alpha\,\log \pi(a|x)\Big], \tag{3.17}$$

where $Q_{\min}(s, a) = \min\{Q_1(s, a), Q_2(s, a)\}$. Note that this is equivalent to minimizing the actor loss:

$$L_\pi = \frac{1}{B}\sum_{i=1}^{B}\left(\alpha\,\log \pi(a'_i \mid x_i) - Q_{\min}(x_i, a'_i)\right), \tag{3.18}$$

where a'_i are reparameterized samples from the current actor's policy. The gradients are computed using the chain rule thanks to the reparameterization trick.

- ***Temperature Tuning:*** The temperature parameter α is adjusted by minimizing:

$$L_\alpha = -\frac{1}{B}\sum_{i=1}^{B} \log\alpha \, \left(\log\pi(a'_i \mid x_i) + \mathcal{H}_{\text{target}}\right), \tag{3.19}$$

where $\mathcal{H}_{\text{target}}$ is the target entropy that aims to control the randomness of the policy.

- ***Soft Updates of Target Networks:*** The target critic networks are updated using exponential moving averages:

$$\theta_{\text{target}} \leftarrow \tau\,\theta_{\text{current}} + (1-\tau)\,\theta_{\text{target}}, \tag{3.20}$$

where τ is a small constant controlling the soft update of the target network.

4. The algorithm evaluates, where after each episode it computes the average reward over the most recent episodes, logs the total success proportion (total successes divided by $N \times T$), and checks for improvements relative to the best-observed performance

5. Finally, at the end of training in a curriculum phase (or overall phases), the function returns a list of episode rewards and a history of evaluation metrics (i.e., average reward, success proportion, elapsed time).

The training process gradually exposes the SAC agent to increasingly complex versions of the clinical trial by scaling the trial length and adjusting reward weights. Within each phase, episodes are executed where illustrated above that the agent selects actions, observes outcomes, receives a compounded reward signal, and improves its policy and value function approximations by sampling from a replay buffer and applying gradient descent on actor and critic losses (also tunes the temperature parameter as we've discussed).

Chapter 4

Comparative Performance in Simulated Clinical Trials

We perform an extensive numerical simulation study to evaluate the performance of our adaptive clinical trial framework and its SAC-based patient allocation strategy. In these simulations, we consider a range of hypothetical scenarios that capture diverse patient outcomes under controlled clinical settings. It is important to note that, given the inherent complexity and uncertainty of clinical trials, no fully optimal solution can be obtained. Nevertheless, by systematically comparing the reinforcement learning agent's policy against a variety of heuristic benchmarks, we can rigorously assess performance differences.

This simulation analysis primarily serves as a robust testbed for quantifying key performance metrics. Most notably, we analyze the success proportion (the fraction of patients achieving favorable outcomes) and the information gain (as defined in the reward function in Chapter 2). Furthermore, the results provide valuable insights into the trade-offs between exploration and exploitation that underpin our adaptive methodology, and they enable us to analyze the evolution of the agent's decision-making trajectory across various scenarios.

Although simulation studies remain within the theoretical realm, they serve as a critical "proof of concept" that showcases our model's potential to improve patient allocation strategies across a suite of different clinical trial situations. In this chapter, we describe the design of our simulation experiments, present the performance metrics obtained, and discuss the implications of our findings for future clinical trial designs.

4.1 Simulation Methodology

Before outlining our simulation methodology, we clarify the distinction between an *episode* and a *simulation*. In our framework, an episode refers to one complete run of the clinical trial environment, from start to finish. During this complete run, the agent makes a sequence of N allocation decisions at each of the T discrete decision points. Consequently, each episode generates a total of $N \cdot T$ outcomes. In contrast, a simulation consists of many independent episodes run under the same (or varying) conditions. This ensemble of episodes is used to capture variability and obtain robust statistical estimates of key performance metrics.

Each episode simulates a complete clinical trial modeled via a Beta–Binomial framework. For every trial, we assume known *true* success probabilities, $p_{\text{true}}^{(E)}$ for the experimental arm and $p_{\text{true}}^{(C)}$ for the control arm. The environment is parameterized by these probabilities, the cohort size N, the number of decision periods T, and the initial Beta priors for each arm. At each decision point, the agent selects an allocation, and subsequently the environment then simulates patient responses through binomial draws based on the true success probabilities. The resulting trajectory, including states, actions, rewards, and updated beliefs, is stored in a replay buffer for further training updates.

Our simulation execution proceeds in several stages:

1. **Hyperparameter Tuning:** For each test scenario defined by distinct trial

settings, we perform hyperparameter tuning to optimize the SAC agent.

2. **Curriculum Training:** The agent is trained using a curriculum learning strategy (as detailed in Chapter 3), which subdivides the trial into phases of increasing complexity.

3. **Policy Evaluation:** After training, we run multiple simulation episodes to assess policy performance and compare the SAC agent against several heuristic benchmarks: Fixed, Greedy, Pure Learning (PL), and Ideal (Oracle). Key performance metrics, including success proportion, allocation ratios, and information gain, are computed from these runs.

We define the heuristics we compare our SAC-based model's outcomes to as follows:

- **Fixed:** The fixed policy serves as a baseline, allocating a constant, fixed proportion of 0.5 to both of the treatment arms. This is what a majority of clinical trial designs currently implement and is not adaptive. Because of this, comparison to a fixed design is one of the most important contributions of this thesis.

- **Greedy:** The greedy policy allocates all patients at each time step to the treatment arm with the higher estimated success probability (posterior mean). Specifically, at time t it allocates 100% of patients to the treatment arm with largest value $\frac{\alpha_t^{(\cdot)}}{\alpha_t^{(\cdot)}+\beta_t^{(\cdot)}}$ (where $(\cdot)$ is either E or C). It's conventionally termed "greedy" because it maximizes immediate expected successes based solely on current point estimates, without accounting for uncertainty. Hence, it can be thought of as the "earning" component of the learning vs. earning tradeoff.

- **Pure Learning (PL):** The PL policy allocates all patients at each time step to the treatment arm with a higher probability of being superior, determined by integrating over the full posterior distributions of both treatment arms. Specifically, this policy computes $\mathbb{P}(p^{(E)} > p^{(C)}) = \int_0^1 f_E(x)\, F_C(x)\, dx,$, (where $f_E(x)$ and $F_C(x)$ are the PDF and CDF of the Beta distributions for the experimental and control arm respectively) and allocates 100% of patients to the experimental arm if this probability exceeds 0.5, and 100% to control if not. As the name suggests, this heuristic can be thought of as the "learning" component of the learning vs. earning tradeoff.

- **Ideal (Oracle):** This policy assumes perfect knowledge of the underlying *true* treatment success probabilities and allocates all patients to the treatment with the highest true probability of success (i.e., 100% to the experimental treatment arm if $p^{(E)}_{\text{true}} > p^{(C)}_{\text{true}}$). Although this approach is not implementable in practice because true probabilities are unknown, we use it to provide somewhat of an upper bound on the performance. It's important to note that since patient outcomes are Bernoulli, other heuristics may perform better by chance, however, this enables comparison to a policy with full prior knowledge.

4.2 Simulation Results and Key Findings

To assess the efficacy of our adaptive clinical trial framework, we evaluated the performance of our SAC-based patient allocation strategy through extensive numerical simulations. Two distinct training schemes were employed:

1. **True-Probability Training:** In this scheme, the agent was trained for 5,000 episodes using patient outcomes generated from Bernoulli draws based on the true success probabilities, $p^{(E)}_{\text{true}}$ and $p^{(C)}_{\text{true}}$. Here, the outcomes perfectly reflect the actual treatment effects.

2. **Estimated-Probability Training:** In a parallel setting, the agent was also trained for 5,000 episodes, but in this case, outcomes were generated using the

model's evolving estimated success probabilities, computed as

$$\frac{\alpha_t^{(E)}}{\alpha_t^{(E)} + \beta_t^{(E)}} \quad \text{and} \quad \frac{\alpha_t^{(C)}}{\alpha_t^{(C)} + \beta_t^{(C)}}.$$

Although the agent does not have direct access to the underlying true values, it instead receives outcomes that statistically represent them.

The primary objective of employing these two schemes is to quantify the impact of uncertainty in outcome generation on the agent's learning process and performance. In both cases, the agent observes only the stochastic outcomes without knowing the true underlying probabilities.

For each training scheme, we simulated 10 different hypothetical trial scenarios — varying the true success probabilities, the initial prior estimates, the cohort size (N), and the number of decision periods (T). During training, we tuned hyperparameters and validated the model every 50 episodes, allowing us to monitor the training trajectory over time. After the training phase, we conducted 500 independent simulations (each consisting of 500 evaluation episodes) to robustly evaluate policy performance. As mentioned, these simulations compare the performance of the SAC policy against the heuristic policies defined above.

The key performance metrics are defined as follows. The *success proportion* is the fraction of successful outcomes recorded within a single clinical trial episode, computed as $S/(N \times T)$. Here, we define $S = \sum_{t=1}^{T}[r_t^{(E)} + r_t^{(C)}]$ as the total number of successes observed in that episode. The standard error of the success proportion is estimated as the standard deviation of the per-episode success proportions divided by the square root of the number of episodes. The *allocation ratio* for the experimental arm is defined as $A_E/(A_E + A_C)$, where we define A_E and A_C in chapter two as the total numbers of patients assigned to the experimental treatment and control arms, respectively, over the episode. Lastly, the *information gain* quantifies the reduction in uncertainty about the treatment success probabilities achieved during an episode, defined in the reward function section of chapter Chapter Two.

4.2.1 True-Probability Training Results

The results for the training scheme where the model is trained to realize Bernoulli patient outcomes based on the true success probabilities for each arm are highlighted in Table 4.1 below.

Parameters				**Expected Proportion of Successes (Std. Error)**				
$p_{\text{true}}^{(E)}; p_{\text{true}}^{(C)}$	$(\alpha_0^{(E)}, \beta_0^{(E)}); (\alpha_0^{(C)}, \beta_0^{(C)})$	N	T	Fixed	Ideal	Greedy	PL	SAC
0.70; 0.50	(1,1); (1,1)	100	8	0.599 (0.0008)	0.700 (0.0007)	0.614 (0.0029)	0.621 (0.0028)	0.700 (0.0007)
0.65; 0.50	(3,7); (5,5)	25	20	0.573 (0.0009)	0.650 (0.0009)	0.498 (0.0010)	0.500 (0.0011)	0.622 (0.0009)
0.65; 0.40	(2,8); (8,2)	40	18	0.524 (0.0008)	0.650 (0.0008)	0.400 (0.0008)	0.399 (0.0008)	0.616 (0.0008)
0.60; 0.55	(4,6); (5,5)	30	25	0.575 (0.0008)	0.601 (0.0008)	0.550 (0.0008)	0.551 (0.0008)	0.579 (0.0008)
0.58; 0.50	(3,7); (4,6)	40	15	0.541 (0.0009)	0.580 (0.0009)	0.501 (0.0009)	0.500 (0.0009)	0.563 (0.0009)
0.57; 0.55	(1,1); (1,1)	100	8	0.560 (0.0008)	0.571 (0.0008)	0.552 (0.0008)	0.553 (0.0007)	0.566 (0.0008)
0.54; 0.50	(2,2); (2,2)	60	25	0.520 (0.0006)	0.540 (0.0005)	0.526 (0.0008)	0.526 (0.0008)	0.529 (0.0006)
0.53; 0.50	(1,1); (1,1)	40	20	0.515 (0.0008)	0.531 (0.0008)	0.517 (0.0008)	0.516 (0.0008)	0.522 (0.0007)
0.52; 0.50	(10,10); (10,10)	50	12	0.509 (0.0009)	0.519 (0.0009)	0.510 (0.0009)	0.511 (0.0009)	0.509 (0.0009)
0.20; 0.17	(2,8); (2,8)	80	20	0.184 (0.0004)	0.200 (0.0005)	0.192 (0.0006)	0.192 (0.0007)	0.194 (0.0004)

Table 4.1: Expected proportion of successes (Std. Error) across simulation parameters with training outcomes realized using true probability of success values

When the SAC policy is trained using the true probabilities for outcome generation, we see it tends to closely follow the Ideal performance in several scenarios. Only in the second to last row, do we see a slight under-performance relative to Greedy and PL. This under-performance can possibly be attributed to the very minor difference in actual treatment success probability for the experimental treatment versus control. It is sensible that the agent is converging to the optimal policy because it realizes outcomes during training based on the underlying true success probabilities. In principle, we consider this to be similar to the law of large numbers, in which the agent should eventually have a good understanding of what these true success probabilities are and be able to determine the optimal policy based on that knowledge.

The performance is steady across varying levels of N, T, and prior values, indicating the model is functioning properly. However, it's very important to note that, while the model may have a good understanding of these actual success probabilities, because of the way the reward function is designed, the agent is incentivized to explore different treatment allocations before converging to a definite policy.

This is an important distinction to clarify because, under other heuristics, we do not know with certainty if a given arm will receive an allocation throughout an entire trial. As we've mentioned previously, this could be very problematic in a real clinical trial setting, especially when it comes to establishing statistical conclusions about the treatment arms. Thus, the agent does a good job of establishing certainty of superiority before allocating a majority to a single treatment arm. For example, consider the trial scenario in the third row where the true success probabilities are 0.70 and 0.50 for the treatment and control arms respectively, while the priors non-informative. The true success probabilities heavily favor the experimental treatment arm (20% difference), and we see the SAC agent achieves a higher expected proportion of successes PL, Greedy, and Fixed, and identical to the ideal case. We visualize the SAC agent's decision-making in Figure 4.1 below. This model output highlights three important areas that the model is addressing correctly. In the top plot, we see that the agent originally allocates around 70% of patients to the treatment arm, meaning it still values learning more about the treatment success probabilities initially. As the trial progresses, we see the agent eventually converge to the optimal allocation, meaning it obtained enough evidence to allocate everyone to the treatment arm by the 6th period of the trial. From the perspective of learning about both treatment arms, this is ideal: there are still enough patients allocated to the control arm to draw statistically significant conclusions, however, the allocation gets progressively more skewed toward the experimental treatment arm as the trial progresses. The latter touches on the "ethics" component of adaptive trials that we've discussed. Instead

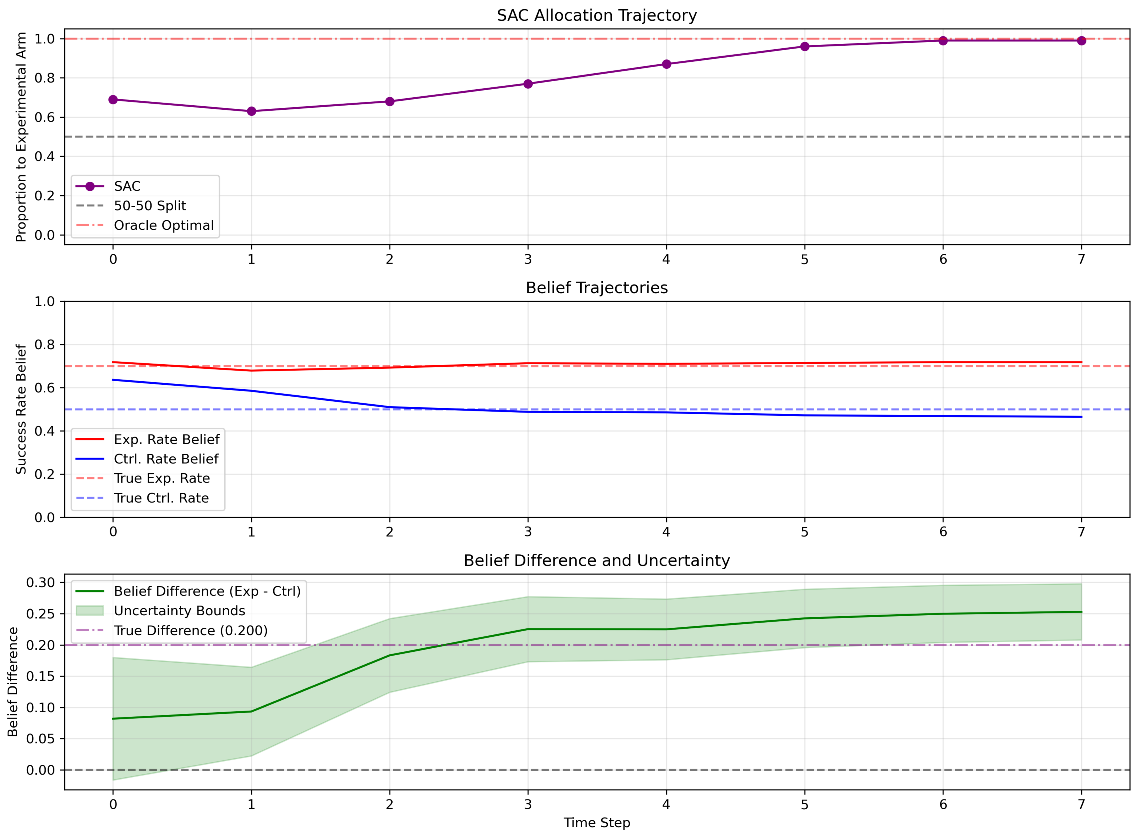

Figure 4.1: SAC Trajectory - Outcomes Realized with True Success Probabilities During Training

of quickly exploiting information that could be noisy, it waits to make any highly unbalanced decisions until certainty is established. Further, even though the agent is still learning about the treatments, it allocates more and more patients to the superior arm which enables more patients to progressively be treated with better treatment.

In terms of the middle plot, we see that the agent begins with some initial prior beliefs about the true success probabilities, different from what they are. As the trial progresses, the agent quickly converges to the actual values, enabling it to then approach the optimal treatment allocation policy. That said, the fact that it quickly converges, yet still allocates some patients to the control arm highlights both the learning component of the reward function and the maximum entropy RL objective. Even if the agent has a solid foundational idea of the superior treatment, it errs on the side of caution until certain.

Lastly, the bottom plot indicates the belief difference with uncertainty bounds based on the posterior variances of the true treatment arms. We can see that the agent starts off with high uncertainty and an incorrect belief difference, however as the trial progresses we see the belief difference approaches the true difference, and the uncertainty of the agent decreases. This plot is very indicative of the allocations observed in the top plot, where when the agent has higher levels of uncertainty, it employs a more balanced allocation strategy until its belief becomes more certain.

Thus, the agent trained using outcomes generated with the true success probabilities performs as we would expect it to. It understands the environment but still confirms the reality before making implementing less balanced allocation strategies. Conversely, the Greedy strategy performs worse as it aims to exploit initial outcomes, hoping these beliefs are reality. In doing so, it is faced with an unfortunate reality when presented with potentially noisy observed outcomes at the start of the trial, in which it tries to exploit an arm that may not be superior. In terms of the fixed strategy, since it isn't able to exploit knowledge gain as the trial progresses, it naturally performs worse when the true success probabilities are different. That said, in cases like the scenario in the ninth row, we see the Fixed strategy performs comparatively, which we would expect.

To further evaluate the agent's performance, we consider the hyperparameter tuning results across the trial scenarios. We note that the ordering (1-10) is in line with the rows from Table 4.1. The hyperparameter tuning results reveal that the SAC agent's optimal settings change depending on the trial's characteristics. This indicates how the agent modulates its learning strategy in response to uncertainty and treatment effect size. First, we note that λ_{aux} represents the hyperparameter that scales the auxiliary loss term in the actor update of our SAC agent. In our implementation, the actor's overall loss is composed of the standard SAC objective—aiming

Scenario	actor_lr	critic_lr	γ	τ	hidden_dim	α	λ_{aux}	target_entropy	Best Performance
1 (High Uncertainty)	0.0001	0.0002	0.995	0.005	256	0.3	0.0	-0.5	0.6863 ± 0.0032
2 (Dynamic Effect)	0.0002	7.7×10^{-5}	0.99	0.005	256	0.2	0.075	-0.5	0.6192 ± 0.0051
3 (Misleading Prior)	7.5×10^{-5}	7.7×10^{-5}	0.99	0.005	256	0.2	0.05	-0.7	0.6200 ± 0.0048
4 (Clear Advantage)	0.0001	0.0001	0.995	0.007	256	0.2	0.05	-0.5	0.5944 ± 0.0033
5 (High-Risk Modest)	0.0002	7.7×10^{-5}	0.999	0.005	256	0.2	0.075	-0.7	0.5682 ± 0.0037
6 (Large Cohort, Low-Effect)	0.0002	0.0002	0.99	0.003	256	0.3	0.05	-0.7	0.5665 ± 0.0039
7 (Minimal Clinically Important)	0.0001	7.7×10^{-5}	0.99	0.003	256	0.2	0.0	-0.7	0.5327 ± 0.0027
8 (Subtle Edge)	7.5×10^{-5}	0.0002	0.99	0.005	256	0.3	0.075	-0.7	0.5235 ± 0.0042
9 (Barely Superior)	0.0001	0.0002	0.995	0.005	256	0.3	0.0	-0.7	0.5183 ± 0.0033
10 (Low Event Rate)	0.0001	7.7×10^{-5}	0.999	0.005	256	0.3	0.075	-0.7	0.1968 ± 0.0019

Table 4.2: Hyperparameter Tuning Results (True Probability Training)

to maximize the expected Q-value while incorporating an entropy bonus—plus an additional auxiliary loss. This auxiliary loss is computed as the mean squared error between the actor's deterministic action (obtained via the network's mean output after the sigmoid transformation) and a heuristic target action derived from state features (such as the difference between the estimated success probabilities of the experimental and control arms). Hence, the weight λ_{aux} determines the relative influence of this extra loss signal; a higher value encourages the policy to more closely follow the heuristic guidance, which can accelerate learning when the initial priors are weak or misleading, while a lower or zero value allows the agent to rely predominantly on the SAC objective itself.

Observing Table 4.2 we see that in Scenario 8 (Subtle Edge, Uninformative Priors), the very low actor learning rate (7.5×10^{-5}) combined with a higher critic learning rate (0.0002) suggests that when prior information is minimal, the agent updates its policy very cautiously while relying more on relatively aggressive value estimation to extract signal from the outcomes. Here, a moderate auxiliary loss weight ($\lambda_{\text{aux}} = 0.075$) provides extra guidance during the early exploration stages. By contrast, Scenario 2 (Dynamic Treatment Effect) opts for a higher actor learning rate (0.0002) and a lower target entropy (-0.5), encouraging rapid adjustments in policy to swiftly capture the pronounced treatment advantage, while the critic learning rate remains very low (7.7×10^{-5}) to stabilize evaluation of the value function. In Scenario 1 ("High Uncertainty - Large Cohort"), the increase in the discount factor ($\gamma = 0.995$) prioritizes long-term rewards from abundant data, and the elimination of auxiliary loss ($\lambda_{\text{aux}} = 0$) reflects a reliance on the primary reward signal rather than external guidance. In scenarios such as 3 and 9, where treatment differences are marginal or the initial priors are strongly misleading, the agent selects similar low learning rates to avoid volatile policy changes. However, the agent adjusts other parameters (e.g., a lower actor α in Scenario 3) to counterbalance misleading information.

In total, these hyperparameter configurations indicate that the SAC agent dynamically modulates its learning speed, degree of exploration, and stability based on the underlying trial structure. In a more rigorous setting, we could have tuned the hyperparameters over many more different configurations to notice underlying patterns. However, we note that the automatic tuning is quite convenient and removes the complexity of manually attempting to tune the hyperparameters.

4.2.2 Estimated-Probability Training Results

Now looking at the case where the SAC agent is trained solely based on its estimates of the true success probabilities, we see a different story. Here, the agent performs worse than in the previous case but is still able to outperform certain heuristics across the scenarios. This setting aims to mimic a more "realistic" trial setting. Doing so creates much more uncertainty, in which the agent may converge to a policy it believes to be optimal, which may not be the case if initial estimates don't reflect the true success values. Although we characterize this trial as more "realistic", we recognize that there are several implicit assumptions. Particularly, this characterization is directly related to the initial Beta priors assigned to the two treatment arms. In a Phase III or IV trial, one might have a good understanding of the treatment's true success

Parameters				**Expected Proportion of Successes (Std. Error)**				
$p_{\text{true}}^{(E)}; p_{\text{true}}^{(C)}$	$(\alpha_0^{(E)}, \beta_0^{(E)}); (\alpha_0^{(C)}, \beta_0^{(C)})$	N	T	Fixed	Ideal	Greedy	PL	SAC
0.70; 0.50	(1,1); (1,1)	100	8	0.600 (0.0008)	0.701 (0.0007)	0.619 (0.0029)	0.618 (0.0029)	0.666 (0.0007)
0.65; 0.50	(3,7); (5,5)	25	20	0.572 (0.0010)	0.650 (0.0010)	0.499 (0.0011)	0.501 (0.0010)	0.566 (0.0011)
0.65; 0.40	(2,8); (8,2)	40	18	0.525 (0.0008)	0.649 (0.0008)	0.400 (0.0008)	0.400 (0.0008)	0.430 (0.0008)
0.60; 0.55	(4,6); (5,5)	30	25	0.574 (0.0008)	0.600 (0.0008)	0.551 (0.0009)	0.550 (0.0009)	0.565 (0.0009)
0.58; 0.50	(3,7); (4,6)	40	15	0.540 (0.0009)	0.580 (0.0009)	0.501 (0.0009)	0.502 (0.0009)	0.523 (0.0010)
0.57; 0.55	(1,1); (1,1)	100	8	0.562 (0.0008)	0.570 (0.0008)	0.554 (0.0007)	0.553 (0.0007)	0.557 (0.0008)
0.54; 0.50	(2,2); (2,2)	60	25	0.521 (0.0006)	0.540 (0.0006)	0.525 (0.0009)	0.525 (0.0008)	0.501 (0.0006)
0.53; 0.50	(1,1); (1,1)	40	20	0.514 (0.0008)	0.531 (0.0008)	0.518 (0.0009)	0.516 (0.0008)	0.507 (0.0008)
0.52; 0.50	(10,10); (10,10)	50	12	0.509 (0.0009)	0.519 (0.0009)	0.510 (0.0009)	0.510 (0.0009)	0.501 (0.0009)
0.20; 0.17	(2,8); (2,8)	80	20	0.184 (0.0004)	0.200 (0.0005)	0.194 (0.0006)	0.195 (0.0006)	0.179 (0.0004)

Table 4.3: Expected proportion of successes (Std. Error) across simulation parameters with training outcomes realized using estimated probability of success values

probability from an earlier stage trial, the agent would effectively be trained based on the underlying true probability. Further, in most cases, the true success probability of the control arm will be known in advance, which helps reduce the stochasticity of the agent's training.

Consider the third row of Table 4.3. In this scenario, the initial priors indicate the control arm is superior, while the experimental treatment arm is superior in reality. In this overarching training setting, we know the model was trained based on these initial beliefs, so naturally, the agent is off to a poor starting point. Because of this noisy training, the agent has a harder time converging to the optimal policy, directly impacting the success proportions we observe. Interestingly, the Fixed strategy performs best, apart from the Ideal policy. This indicates that in scenarios where initial beliefs are uncertain or incorrect, implementing a fixed strategy may be the best route. There is a reason the fixed, randomized design has been implemented in clinical trials for so long, however, it's still somewhat of a surprising result considering the noticeable true treatment difference of 25% in this scenario. Considering the number of time steps T is considerable at 18, and the number of patients per cohort is 40, we expected better performance from the agent in this setting.

Looking at the scenario in the ninth row of Table 4.3, we observe the following plot. In this plot, we can see that the SAC agent performs poorly in the case where

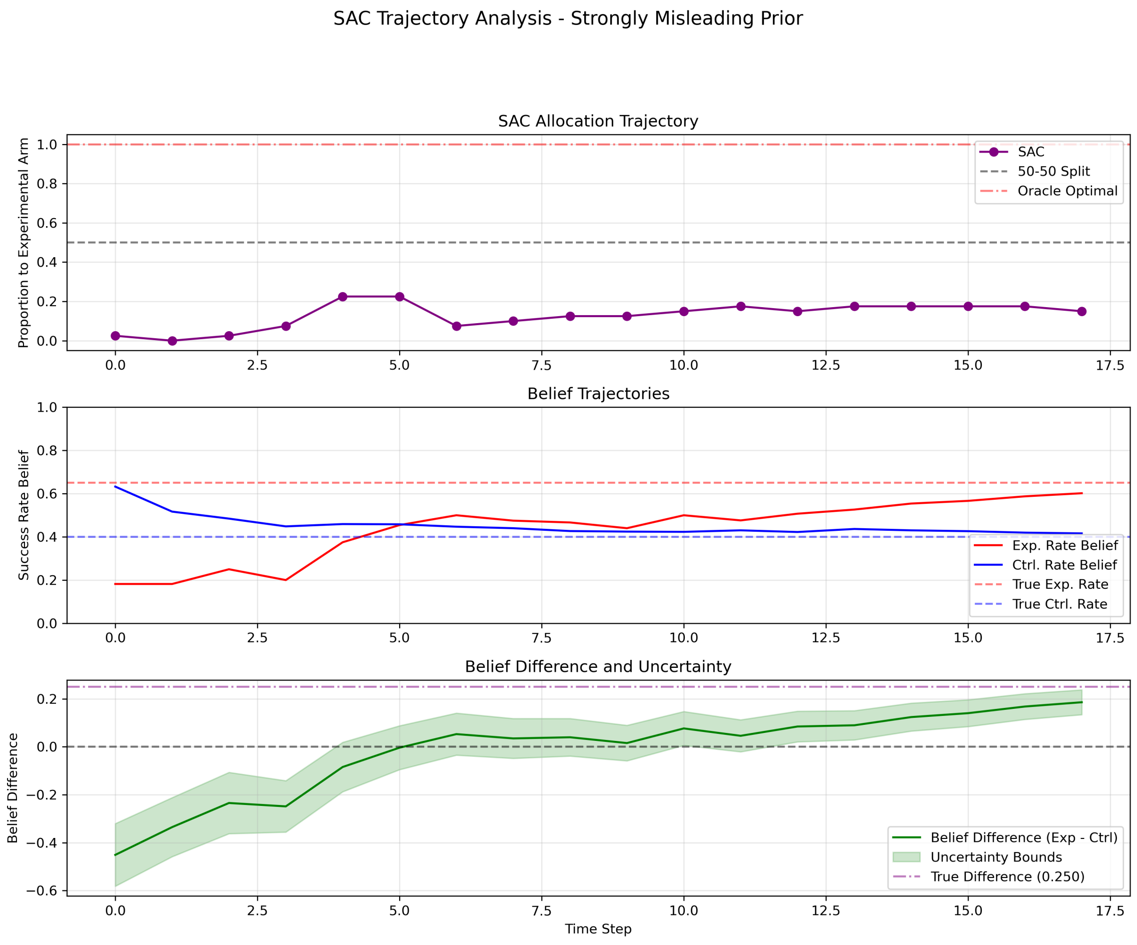

Figure 4.2: SAC Trajectory - Outcomes Realized with Estimated Success Probabilities During Training

true success probabilities for the treatment arms are different, but the priors are misleading. Figure 4.2 highlights the same three ideas discussed in the true probability training scenario. In the top plot, we see the agent allocating less than 20% to the experimental treatment arm throughout the trial. Since the treatments are so different, the allocations are different than what we observe, however, we know that the agent was trained on priors that completely misrepresent the true probabilities.

The middle plot highlights that although the training indicated the opposite, throughout the trial the agent is actually able to still converge on the true probability values. However, this altering belief occurs towards the middle of the trial, and only towards the end does the agent have better beliefs.

The overall understanding is highlighted in the bottom plot, where the belief uncertainty never quite achieves the true difference. Although it is approaching the true value, the agent takes a long time to get here. Hence the agent's initial belief that the control is significantly better than the experimental treatment carries through for almost a third of the trial.

Overall, the agent trained using outcomes generated with the estimated success probabilities performs inferior to the other setting, which we assumed would be the case. In a real-world setting, you would hope that the initial beliefs of the treatment are in the correct region, in which case this model would perform well and converge to the optimal policy. As mentioned, a setting like a Phase III clinical trial would be much better for this agent to perform well, as there would be a good understanding of what the true success probabilities are based on previous trials. In doing so, the agent would perform more in line with the agent trained using the true success probabilities. This training setting better enables the agent to provide a good optimal policy. However, we note overall that in scenarios with high uncertainty, where initial beliefs are significantly different than true values, the agent can't sufficiently train on the correct scenario, and hence doesn't perform ideally.

We include the tuning results for this setting as well, seen in Table 4.2.2. We note that it is hard to draw conclusions from these hyperparameter choices, as the agent experienced significantly more noise than when trained using true probability training results.

Scenario	actor_lr	critic_lr	γ	τ	hidden_dim	α	λ_{aux}	target_entropy	Performance
1 (High Uncertainty)	0.0001	0.0002	0.999	0.005	256	0.2	0.0	-0.7	0.6847 ± 0.0039
2 (Dynamic Effect)	0.0002	0.0002	0.99	0.007	256	0.2	0.05	-0.7	0.5100 ± 0.0060
3 (Misleading Prior)	7.5×10^{-5}	0.0001	0.99	0.003	256	0.2	0.05	-0.7	0.4087 ± 0.0043
4 (Clear Advantage)	7.5×10^{-5}	7.7×10^{-5}	0.99	0.005	256	0.2	0.075	-0.5	0.5920 ± 0.0024
5 (High-Risk Modest)	0.0002	7.7×10^{-5}	0.99	0.003	256	0.3	0.075	-0.5	0.5687 ± 0.0036
6 (Large Cohort, Low-Effect)	0.0002	7.7×10^{-5}	0.99	0.003	256	0.2	0.05	-0.5	0.5654 ± 0.0050
7 (Minimal Clinically Important)	0.0002	7.7×10^{-5}	0.995	0.003	256	0.3	0.05	-0.5	0.5365 ± 0.0023
8 (Subtle Edge)	7.5×10^{-5}	7.7×10^{-5}	0.995	0.003	256	0.3	0.0	-0.7	0.5280 ± 0.0032
9 (Barely Superior)	7.5×10^{-5}	0.0002	0.999	0.003	256	0.3	0.0	-0.7	0.5162 ± 0.0038
10 (Low Event Rate)	0.0001	7.7×10^{-5}	0.995	0.003	256	0.2	0.075	-0.5	0.1969 ± 0.0018

Table 4.4: Hyperparameter Tuning Results (Estimated-Probability Training)

4.3 Results Summary

In evaluating our Soft Actor-Critic (SAC) reinforcement learning agent within the adaptive clinical trial model, two distinct training methodologies were utilized: *True-Probability Training* and *Estimated-Probability Training*. The results illuminate critical differences in policy performance across these training paradigms, providing insights into the efficacy and robustness of our adaptive trial strategy under varying levels of uncertainty.

Under true probability training, where patient outcomes were generated directly from known true success probabilities, the SAC agent consistently achieved near-optimal performance. Specifically, the SAC policy closely matched the *Ideal* (Oracle) policy across most trial scenarios, reflecting its ability to leverage accurate training signals effectively. For instance, in the scenario with a significant 20% true success rate differential (experimental arm at 70%, control at 50%), the SAC policy reached the ideal success proportion of 0.700, surpassing Fixed (0.599), Greedy (0.614), and Pure Learning (PL, 0.621) strategies. Importantly, the SAC policy demonstrated robust exploration initially, avoiding premature convergence and ensuring statistically meaningful comparisons by adequately sampling both arms before allocating predominantly to the superior treatment.

Visualizing agent trajectories (Figure 4.1) further reinforced that the SAC policy balances the learning and exploitation trade-off adeptly. Initially maintaining balanced allocations due to uncertainty, the SAC agent gradually shifted allocations decisively once sufficient evidence accumulated. This strategy addresses both ethical and statistical considerations intrinsic to clinical trials: patients are increasingly allocated to the superior arm over time, while adequate data from both arms ensure statistical validity.

Hyperparameter tuning results under true probability training (Table 4.2) further validate the agent's adaptability. Different scenarios required distinct parameter configurations, indicating that the SAC agent dynamically modulated its learning rate, entropy targets, and auxiliary loss weight based on scenario-specific characteristics. Particularly noteworthy was the nuanced balancing of actor and critic learning rates and entropy targets, highlighting the SAC agent's capacity to adjust learning dynamics appropriately in response to uncertainty and treatment effect magnitude.

In contrast, estimated probability training, which aims to simulate a more realistic scenario by generating outcomes based on evolving estimated success probabilities rather than known truths, presented greater challenges. Here, the SAC agent exhibited reduced efficacy compared to True-Probability Training, primarily due to increased uncertainty inherent in the learning environment. However, it still outperformed or closely matched heuristic strategies like Fixed, Greedy, and PL in several cases, although the gap from the Ideal policy widened noticeably.

For instance, in scenarios where initial priors were misleading—such as the third scenario (true probabilities of 65% vs. 40%, but initially favoring the control arm)—the SAC policy significantly underperformed (success proportion 0.430) relative to Fixed (0.525). This underscores the critical dependence of policy effectiveness on accurate initial prior beliefs. Scenarios like these illuminate conditions under which a non-adaptive Fixed strategy may remain preferable, especially when initial uncertainty or misinformation is substantial.

The trajectory visualization (Figure 4.2) from Estimated-Probability Training underscored this limitation. Although the SAC agent eventually corrected its beliefs about true probabilities mid-trial, significant allocations had already been influenced by initially incorrect priors, delaying convergence to optimal allocations.

Hyperparameter tuning under Estimated-Probability Training (Table 4.2.2) revealed less distinct patterns than under True-Probability Training, reflecting higher noise and uncertainty in the agent's learning signals. Nevertheless, appropriate hyperparameter choices still influenced performance positively, particularly learning rates and auxiliary loss weights, though the agent's performance was inherently more volatile.

In summary, these simulation results robustly demonstrate the strengths and limitations of employing an SAC reinforcement learning agent within an adaptive clinical trial framework. Under idealized conditions, the agent achieves near-optimal performance by effectively balancing exploration and exploitation. However, its performance is sensitive to initial belief accuracy, highlighting critical considerations for practical deployment. Future exploration could focus on methods for making the agent's performance more robust against uncertainty and incorrect prior beliefs, potentially enhancing its utility in realistic adaptive clinical trial contexts.

The code for this model can be found at `github.com/mbwiller/SAC-Adaptive-Trial`

Chapter 5

Future Work

In terms of future work, there are several avenues we could take to build upon the framework. To start, evaluating how the model performs on a real trial dataset would provide valuable insights into its applicability in a real-world setting. A dataset used by Varatharajah and Berry (2022) is the International Stroek Trial (IST). This trial comprised 19,435 patients with suspected acute ischaemic stroke, across 467 hospitals in 36 different countries. The main goal of the IST was to determine the safety and efficacy of aspirin and subcutaneous heparin. Under a fixed, randomized design, patients were allocated to treatment arms using a factorial allocation scheme. A factorial trial is one that allows testing of multiple different treatment interventions and examines different combinations of such. In the IST specifically, there were four treatment arms, defined as ”immediate aspirin”, ”immediate heparin”, ”avoid aspirin”, and ”avoid heparin”. The primary outcomes analyzed were defined as ”death within 14 days” and ”death or dependency at 6 months” [13]. With around a 99% follow-up rate, the overall trial was able to obtain an entire dataset of outcomes, which can be found publicly online.

Including treatment allocations and patient outcomes, the dataset also published patient-level covariate information, and a suite of other variables from the trial outcomes. Hence, the IST provides an intriguing setting to test the performance of our SAC-based model, where instead of simulating patient outcomes, the true responses can actually be "observed" for the most part. We state for the most part, because in the case where the model may assign a treatment to a patient that they weren't actually given, their observed out would need to be estimated in some manner. This could be done by incorporating covariates into the model and realizing outcomes based on all of the patients in the dataset with a given covariate, similar to what Varatharajah and Berry incorporated into their contextual bandit model [35].

Another avenue worth considering is the incorporation of the Gram-Schmidt Walk, which is outlined in Appendix D.1.2. The GSW framework was explored extensively, and the ideas for future consideration are outlined in the Appendix.

Appendix A

Key Parameters Influencing the SAC Clinical Trial Model

Table A.1: Key Parameters Influencing the SAC Clinical Trial Model

Parameter	**Description**	**Typical Values / Role**
N	Number of patients per cohort.	25, 30, 40, 50, 60, 80, 100
T	Total number of discrete time periods (cohorts) in the trial.	5, 8, 12, 15, 18, 20, 25
$\alpha_0^{(E)}$	Initial alpha for experimental arm's Beta distribution.	1, 3, 4, 5, 10
$\beta_0^{(E)}$	Initial beta for experimental arm's Beta distribution.	1, 5, 6, 7, 10
$\alpha_0^{(C)}$	Initial alpha for the control arm's Beta distribution.	1, 4, 5, 10
$\beta_0^{(C)}$	Initial beta for control arm's Beta distribution.	1, 5, 6, 8, 10
$p_{\text{true}}^{(E)}$	True probability of success for experimental treatment arm.	0.53, 0.55, 0.60, 0.65, 0.70
$p_{\text{true}}^{(C)}$	True probability of success for control arm.	0.40, 0.50, 0.55
λ_{TVD}	Scaling factor for the TVD information gain reward.	0.08, 0.1, 0.12, 0.15
λ_{explore}	Scaling factor for the exploration bonus.	0.04, 0.05, 0.07, 0.08
actor_lr	Learning rate for the actor network; controls the speed and stability of policy updates.	1×10^{-4}, 3×10^{-4}, 1×10^{-3}
critic_lr	Learning rate for the critic networks; affects convergence rate of value function estimation.	1×10^{-4}, 3×10^{-4}, 1×10^{-3}
γ	Discount factor for future rewards; balances immediate vs. long-term reward contributions.	Typically 0.99 or 0.995 (range: 0.95–0.999)
τ	Soft update coefficient for target networks; determines the rate target networks track the online networks for stable learning.	Around 0.005 (or 0.007 for slightly faster tracking)
hidden_dim	Number of hidden units in the neural network layers; influences model capacity and the ability to capture complex patterns.	128, 256
α	Temperature parameter in SAC that controls the trade-off between exploration and exploitation by weighting the entropy term.	Typically around 0.2; often auto-tuned
target_entropy	The target entropy level for the policy when using auto-tuning; a less negative value encourages more exploration.	E.g., -0.7, -1.0, -2.0

Appendix B

Additional Outputs

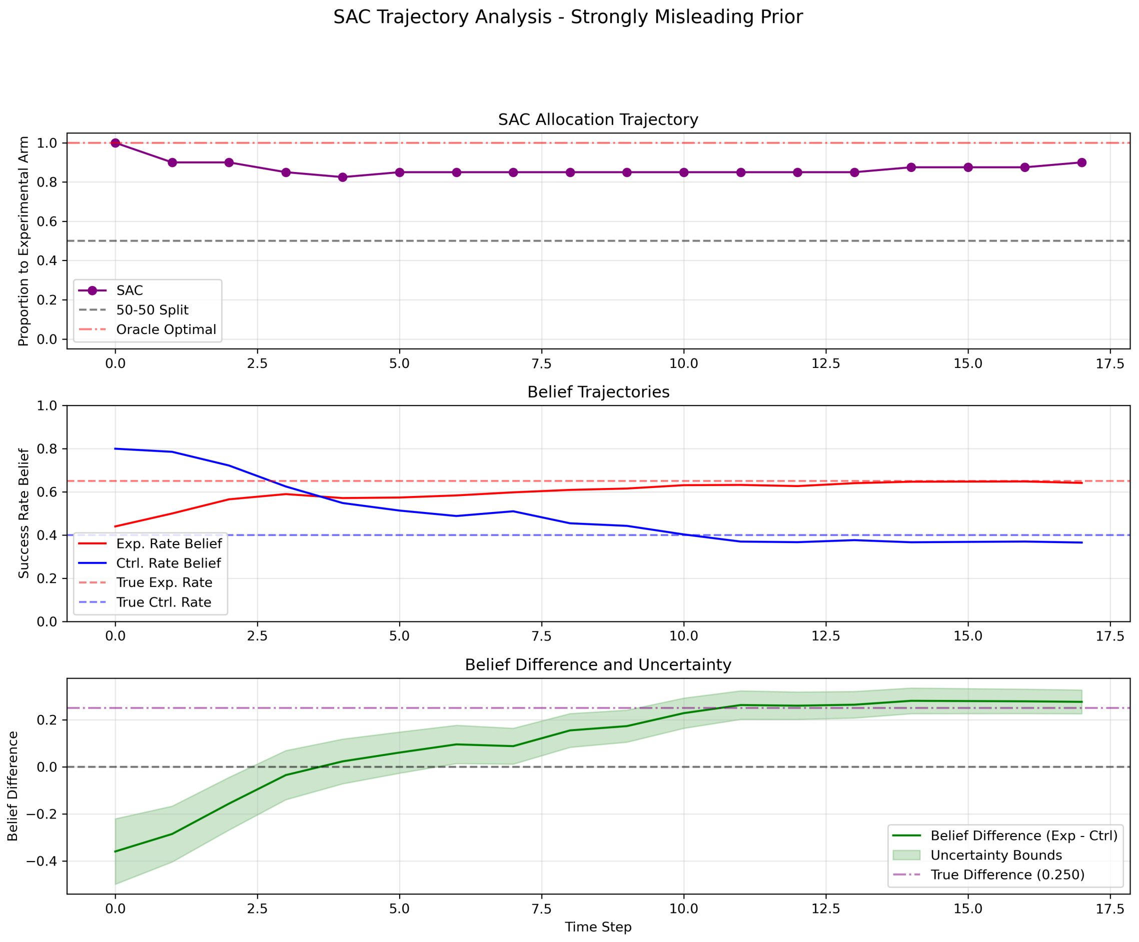

Figure B.1: SAC Trajectory (Subtle Edge - Uninformative Priors) - Outcomes Realized with True Success Probabilities During Training

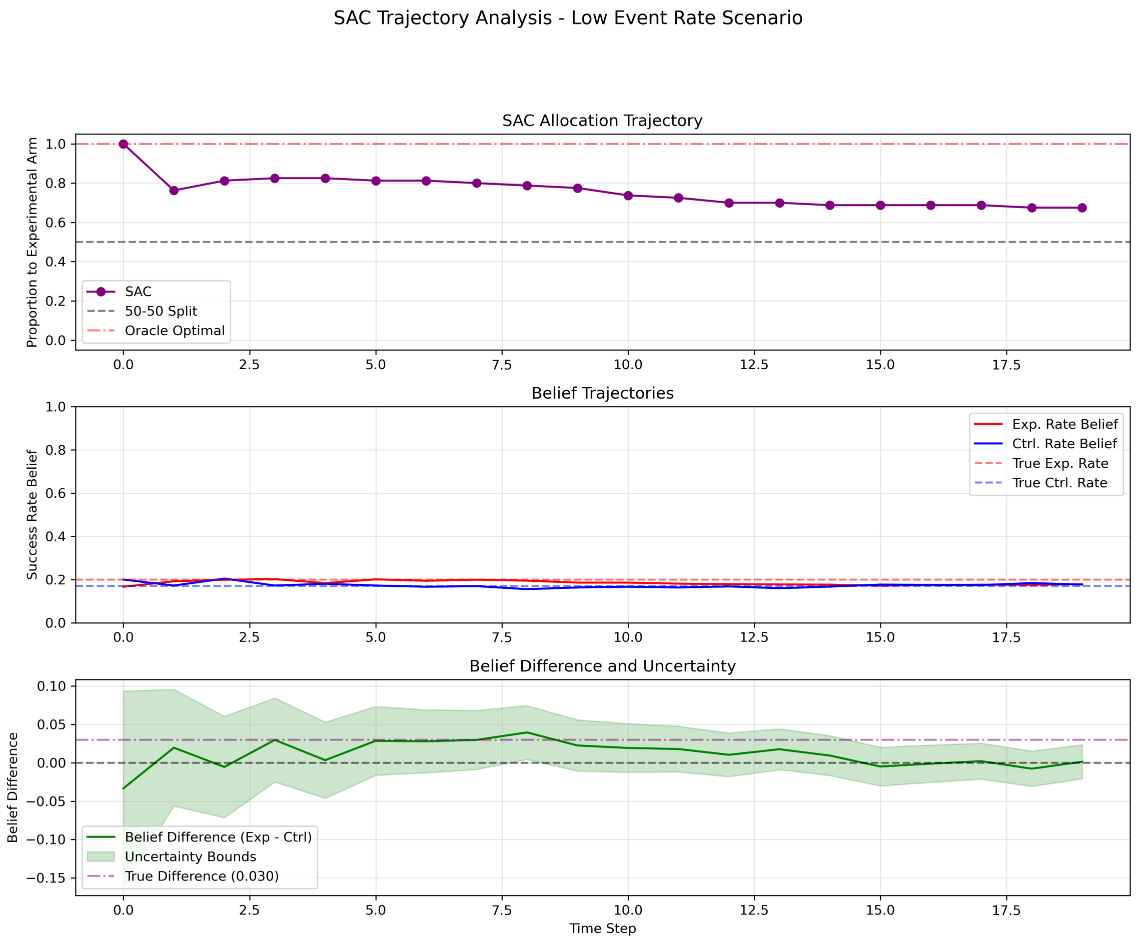

Figure B.2: SAC Trajectory (Low Event Rate) - Outcomes Realized with True Success Probabilities During Training

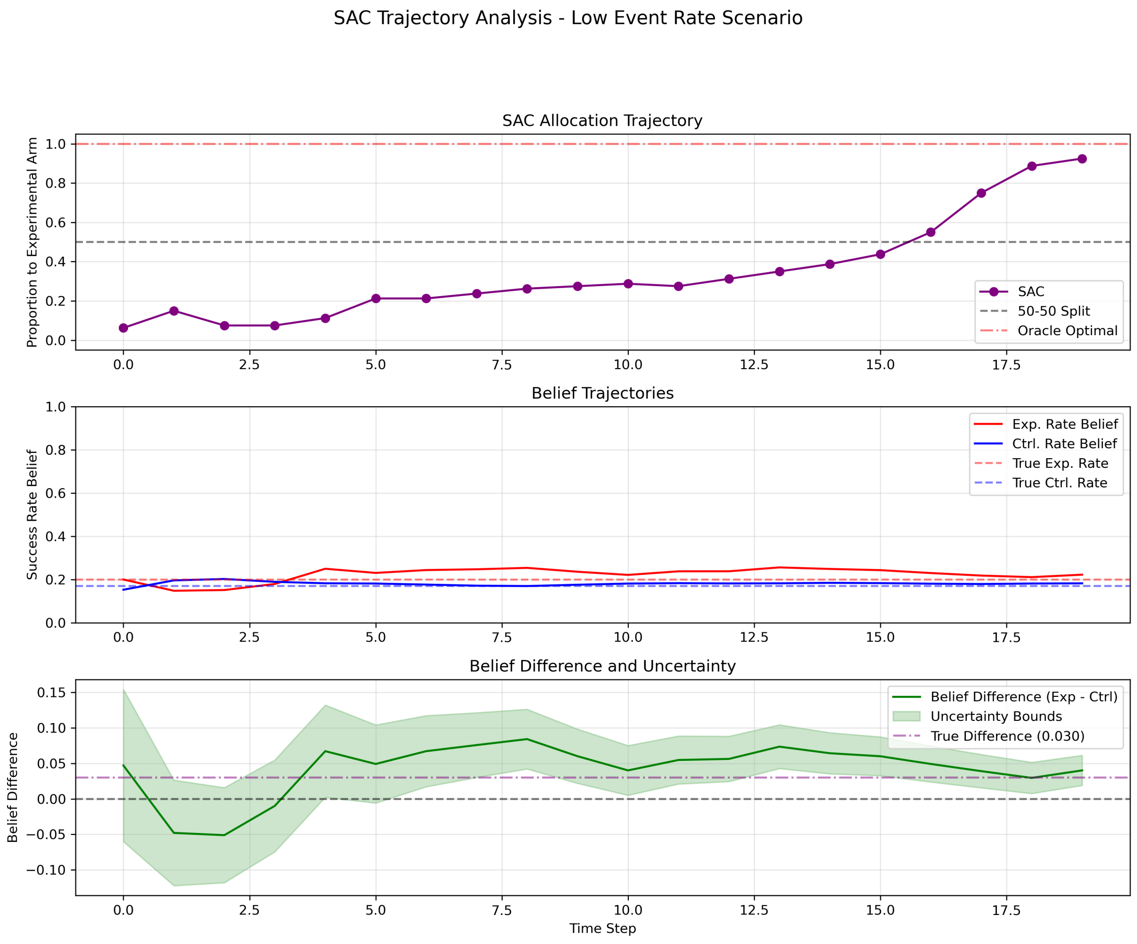

Figure B.3: SAC Trajectory (Low Event Rate) - Outcomes Realized with Estimated Success Probabilities During Training

Appendix C

Proofs

C.1 Proof of Beta-Binomial Conjugacy

Proof. To prove this result, we consider the true success probability for the treatment arm at time step t, denoted as $p_t^{(E)}$. Going forward, the superscript "(E)" is removed for clarity. We assume that p_t follows a Beta-distributed prior:

$$p_t \sim \text{Beta}(\alpha_{t-1}, \beta_{t-1})$$

and use the fact that its probability density function (PDF) is defined as:

$$f_{p_t}(p_t) = \frac{p_t^{\alpha_{t-1}-1}(1-p_t)^{\beta_{t-1}-1}}{\mathrm{B}(\alpha_{t-1}, \beta_{t-1})}$$

where $\mathrm{B}(\alpha_{t-1}, \beta_{t-1})$ is the Beta function:

$$\mathrm{B}(\alpha_{t-1}, \beta_{t-1}) = \int_0^1 p^{\alpha_{t-1}-1}(1-p)^{\beta_{t-1}-1} dp$$

To clean this up, we use the Gamma function representation of the Beta function:

$$\mathrm{B}(\alpha, \beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$$

and thus have:

$$f_{p_t}(p_t) = \frac{\Gamma(\alpha_{t-1}+\beta_{t-1})}{\Gamma(\alpha_{t-1})\Gamma(\beta_{t-1})} p_t^{\alpha_{t-1}-1}(1-p_t)^{\beta_{t-1}-1}$$

Because we model patient outcomes as binary, each individual outcome $y_i \in \{0,1\}$ follows a Bernoulli distribution, where the success probability is the unknown true response rate of the assigned treatment arm. Given that we observe r_t successes out of n_t total patients at time step t, the likelihood function follows a Binomial distribution:

$$\mathbb{P}(r_t \mid p_t) = \binom{n_t}{r_t} p_t^{r_t}(1-p_t)^{n_t-r_t}$$

It follows from Bayes' Theorem that the posterior distribution for p_t given the observed successes r_t is:

$$\mathbb{P}(p_t \mid r_t) = \frac{\mathbb{P}(r_t \mid p_t) f_{p_t}(p_t)}{f_{r_t}(r_t)}$$

where $f_{r_t}(r_t)$ is the probability mass function (PMF) of the observed number of successes r_t thus far:

$$f_{r_t}(r_t) = \int_0^1 \mathbb{P}(r_t \mid p_t) f_{p_t}(p_t)\, dp_t = \binom{n_t}{r_t} \frac{\Gamma(\alpha_{t-1}+\beta_{t-1})}{\Gamma(\alpha_{t-1})\Gamma(\beta_{t-1})} \int_0^1 p_t^{r_t+\alpha_{t-1}-1}(1-p_t)^{n_t-r_t+\beta_{t-1}-1}\, dp_t$$

Combining the terms in the Bayes expression, we can write the conditional posterior for p_t as:

$$\mathbb{P}(p_t \mid r_t) = \frac{\binom{n_t}{r_t} \frac{\Gamma(\alpha_{t-1}+\beta_{t-1})}{\Gamma(\alpha_{t-1})\Gamma(\beta_{t-1})} p_t^{r_t+\alpha_{t-1}-1}(1-p_t)^{n_t+\beta_{t-1}-r_t-1}}{\binom{n_t}{r_t} \frac{\Gamma(\alpha_{t-1}+\beta_{t-1})}{\Gamma(\alpha_{t-1})\Gamma(\beta_{t-1})} \int_0^1 p_t^{r_t+\alpha_{t-1}-1}(1-p_t)^{n_t+\beta_{t-1}-r_t-1}\, dp_t}$$

which simplifies to:

$$\mathbb{P}(p_t \mid r_t) = \frac{p_t^{r_t+\alpha_{t-1}-1}(1-p_t)^{n_t+\beta_{t-1}-r_t-1}}{\int_0^1 p_t^{r_t+\alpha_{t-1}-1}(1-p_t)^{n_t+\beta_{t-1}-r_t-1}\,dp_t}$$

and finally by noticing the Beta function in the denominator:

$$\int_0^1 p_t^{r_t+\alpha_{t-1}-1}(1-p_t)^{n_t+\beta_{t-1}-r_t-1}\,dp_t = \mathrm{B}(\alpha_{t-1}+r_t, \beta_{t-1}+n_t-r_t)$$

we are left with the following expression:

$$\mathbb{P}(p_t \mid r_t) = \frac{p_t^{r_t+\alpha_{t-1}-1}(1-p_t)^{n_t+\beta_{t-1}-r_t-1}}{\mathrm{B}(\alpha_{t-1}+r_t, \beta_{t-1}+n_t-r_t)} \tag{C.1}$$

Therefore, the full recursive update rules for the posterior parameters of the Beta distributed random variable is:

$$\alpha_t = \alpha_{t-1} + r_t \quad \text{and} \quad \beta_t = \beta_{t-1} + n_t - r_t$$

This completes the proof for the Beta-Binomial conjugacy. □

Appendix D

Algorithms

The algorithm on the following page provides a high-level overarching flow of how the adaptive trial simulation algorithm runs from start to finish. In the subsequent pages, we outline the overarching scheme of the Gram-Schmidt Walk, which was an algorithm explored when considering the incorporation of covariates into the adaptive framework. We include it here as a component of future work, specifically due to the novelty of the algorithm and what it aims to accomplish.

Algorithm 1 Adaptive Clinical Trial with SAC-Based Patient Allocation

```
1:  Input: Trial parameters: $N, T$; Prior Beta parameters $\alpha_0$, $\beta_0$ for both arms; true
    success probabilities $p_{\text{true}}^{(E)}, p_{\text{true}}^{(C)}$; SAC hyperparameters; reward shaping parame-
    ters.
2:  Output: Learned SAC policy $\pi$ and performance metrics.
3:  1. Initialization
4:  Instantiate environment $\mathcal{E}$ with parameters.
5:  Initialize SAC agent with actor $\pi_\theta$, critics $Q_{\phi_1}, Q_{\phi_2}$, and replay buffer.
6:  2. Training with Curriculum Learning
7:  for each phase $p = 1$ to $P$ do
8:      Set horizon $T_p \leftarrow T_{\text{factor}_p} \cdot T$
9:      Update environment $\mathcal{E}_p$
10:     for each episode $e$ in phase $p$ do
11:         Reset $\mathcal{E}_p$ to obtain $x_0$
12:         for $t = 0$ to $T_p - 1$ do
13:             Sample action $a_t \sim \pi_\theta(s_t)$
14:             Simulate environment and update:
15:             $\alpha_E \leftarrow \alpha_E + r_E$
16:             $\beta_E \leftarrow \beta_E + (n_E - r_E)$
17:             $\alpha_C \leftarrow \alpha_C + r_C$
18:             $\beta_C \leftarrow \beta_C + (n_C - r_C)$
19:             Compute reward $r_t$ and store transition $(x_t, a_t, r_t, x_{t+1})$
20:         end for
21:         Update SAC agent using collected transitions
22:     end for
23: end for
24: Soft-update target networks:
25: $\theta_{\text{target}} \leftarrow \tau \cdot \theta_{\text{critic}} + (1 - \tau) \cdot \theta_{\text{target}}$
26: 3. Policy Evaluation
27: Evaluate oracle benchmark:
28: $S(n_E) = \sum_{i=1}^{n_E} \text{outcome}_E(i) + \sum_{j=1}^{NT-n_E} \text{outcome}_C(j)$
29: Evaluate heuristic and SAC policies on Bernoulli outcomes using $p_{\text{true}}^{(E)}, p_{\text{true}}^{(C)}$
30: Compute performance metric: Success Proportion $= \frac{\text{Total Successes}}{N \cdot T}$
31: 4. Parameter Study
32: for each parameter set do
33:     Repeat Steps 1–3
34:     Record metrics: mean success rate, SE, regret
35: end for
36: Aggregate results and return summary
```

D.1 Incorporating the Gram-Schmidt Walk

The Gram-Schmidt Walk (GSW) was explored initially, when our work was considering the incorporation of covariate balance.

D.1.1 Why Incorporate Covariates into the Adaptive Structure?

The Gram-Schmidt Walk algorithm, proposed in 2021, provides an optimal and computationally efficient method for achieving robustness and covariate balance within randomized experiments [16]. Specifically, the GSW uses an orthogonalization process (analogous to the Gram-Schmidt procedure in linear algebra) to sequentially assign treatments to incoming subjects in a way that minimizes emerging covariate differences between groups. The result is a randomization scheme that is far more balanced than pure random assignment, yet still retains a probabilistic element to avoid deterministically assigning everyone. Harshaw et al. (2021) show that the mean squared error of treatment effect estimates under GSW is dramatically lower than with traditional randomization, approaching the theoretical optimum given the constraints [16]. In the context of the adaptive framework presented in this thesis, we could explore the integration of the GSW within each interim allocation update: whenever the allocation percentages are adjusted based on the action chosen in the MDP, the actual assignment of treatments to individual patients is done via a GSW procedure to ensure covariates remain balanced. In doing so, the hope would be to couple the ethical and operation advantages of an adaptive design with the statistical benefits of covariate balance. To illustrate this, consider a trial with $N = 20$, where we are at intervention period t and have chosen action action $a_t = 75\%$. Under this, we allocate 15 patients to the experimental treatment arm and 5 to the control arm. Rather than simply randomizing 15 of those patients to E and 5 to C, the GSW-based approach assigns those 20 patients in a way that keeps covariates as balanced as possible across the arms. The goal in doing would be to incorporate a structured randomization approach with the aim of significantly improving the reliability of the trial outcomes and reducing confounding.

D.1.2 Outlining the GSW Algorithm

We include, for better understanding, the process in which the GSW algorithm balances augmented covariates by iteratively assigning patients. Harshaw et al. (2021) provide the full framework and resulting proofs behind the algorithm [16]. Outlined below are the specific components of the GSW design that are coupled with the overarching adaptive framework, namely the mathematical setup of the GSW design. If we were using a dataset to test the model retrospectively, we would need full covariate data for each patient in the trial, enabling the GSW to be incorporated. The use of individual covariate information is rarely included within existing adaptive frameworks, as obtaining patient-level data is typically very difficult, and other methods that try to assign distributions to covariates become incredibly assumption-heavy. In the subsections that follows, we show the overarching algorithm that Harshaw et al. (2021) detail [16]. This is included (1) due to the work we did exploring the feasibility of implementing the GSW into the MDP framework, and (2) to highlight the potential benefits of utilizing the GSW in a clinical trial setting. We note that the latter has not been done, and hence, we present this as a component of possible future work.

Augmented Covariate Matrix (B)

To facilitate covariate balancing, the GSW design uses an augmented covariate matrix:

$$\mathbf{B} = \begin{bmatrix} \sqrt{\phi}\mathbf{I} \\ \\ \xi^{-1}\sqrt{1-\phi}\mathbf{X}^\top \end{bmatrix}, \tag{D.1}$$

where:

- $\phi \in [0,1]$: A trade-off parameter controlling the balance-robustness trade-off. Specifically, $\phi = 0$ maximizes covariate balance, while $\phi = 1$ prioritizes robustness (randomization).
- $\mathbf{I}$: $n \times n$ identity matrix, ensuring individual-level randomness.
- $\mathbf{X}^\top$: Transpose of the $n \times d$ covariate matrix X, where each row x_i contains the d-dimensional covariates for patient i.
- $\xi = \max_{i\in[n]} \|\mathbf{x}_i\|$: The maximum row norm of $\mathbf{X}$, ensuring appropriate scaling.

Objective of the GSW Design

The GSW algorithm seeks to optimize the assignment vector $\mathbf{z}$, where $z_i \in \{-1, +1\}$, representing the assignment of patient i to the control (-1) or treatment $(+1)$ group, respectively. As such, $\mathbf{z}$ determines Z^+ and Z^-, where $Z^+ = \{i \in [n] : z_i = +1\}$ and $Z^- = \{i \in [n] : z_i = -1\}$. The design balances the augmented covariates by minimizing the discrepancy between groups:

$$\min_{\mathbf{z}\in\{-1,+1\}^n} \|\mathbf{Bz}\|_\infty$$

where:

- $\mathbf{Bz}$: Represents the weighted imbalance of augmented covariates between groups.

- $\| \cdot \|_\infty$: Denotes the infinity norm, ensuring the maximum imbalance across all covariates is minimized.

Algorithm for Assigning Patients

The GSW algorithm operates iteratively to construct the assignment vector $\mathbf{z}$:

1. **Initialization:**
   - Set $\mathbf{z}_1 = \mathbf{0}$, where all patients initially have fractional assignments.
   - Let $t = 1$.
2. **While $\mathbf{z}_t \notin \{-1, +1\}^n$:**
   (a) Identify the set of active patients:
   $$A = \{i \in [n] : |\mathbf{z}_t(i)| < 1\}.$$
   (b) Select a pivot patient $p \in A$ uniformly at random.
   (c) Compute the step direction u_t to minimize the imbalance:
   $$u_t = \arg\min_{u \in U} \|Bu\|_\infty,$$
   where U is the set of vectors satisfying $u_p = 1$ and $u_i = 0$ for all $i \notin A$.
   (d) Compute the step size:
   $$\delta^+ = \min \left(\frac{1 - \mathbf{z}_t(i)}{u_t(i)} \right)_{i \in A}$$
   $$\delta^- = \min \left(\frac{-1 - \mathbf{z}_t(i)}{u_t(i)} \right)_{i \in A}$$

(e) Randomly select the step:

$$\delta_t = \begin{cases} \delta^+, & \text{with probability } \frac{\delta^-}{\delta^+ + \delta^-}, \\ -\delta^-, & \text{with probability } \frac{\delta^+}{\delta^+ + \delta^-}. \end{cases}$$

(f) Update the fractional assignment vector:

$$\mathbf{z}_{t+1} = \mathbf{z}_t + \delta_t u_t.$$

3. Increment t and repeat until all assignments are in $\{-1, +1\}^n$.

Output of the GSW Algorithm

After completing the iterations:

- The assignment vector $\mathbf{z} \in \{-1, +1\}^n$ is finalized, with each patient assigned to treatment $(+1)$ or control (-1).
- The augmented covariate imbalance $\|B\mathbf{z}\|_\infty$ is minimized, ensuring balance across all linear functions of the covariates.

The main component of this setup is the choice of the parameter ϕ, which is chosen by the experimenter. For implementation, one could retrospectively simulate possible trial outcome by implementing the GSW using varying levels of ϕ. However, taking $\phi = \frac{1}{2}$, implies an equal emphasis on covariate balance and robustness of the design. We note that most clinical trials likely have more of an emphasis on robustness, so the choice of ϕ closer to 1 may be more acceptable.

Bibliography

[1] V. Ahuja and J. R. Birge. Response-adaptive designs for clinical trials: Simultaneous learning from multiple patients. *European Journal of Operational Research*, 248(2):619–633, 2016.

[2] V. Ahuja and J. R. Birge. An approximation approach for response-adaptive clinical trial design. *Informs journal on computing*, 32(4):877–894, 2020.

[3] D. C. Angus, S. Berry, R. J. Lewis, F. Al-Beidh, Y. Arabi, W. van Bentum-Puijk, Z. Bhimani, M. Bonten, K. Broglio, F. Brunkhorst, et al. The remap-cap (randomized embedded multifactorial adaptive platform for community-acquired pneumonia) study. rationale and design. *Annals of the American Thoracic Society*, 17(7):879–891, 2020.

[4] A. Barker, C. Sigman, G. J. Kelloff, N. Hylton, D. A. Berry, and L. Esserman. I-spy 2: an adaptive breast cancer trial design in the setting of neoadjuvant chemotherapy. *Clinical Pharmacology & Therapeutics*, 86(1):97–100, 2009.

[5] D. A. Berry. Bayesian clinical trials. *Nature reviews Drug discovery*, 5(1):27–36, 2006.

[6] A. Bhattacharyya, S. Gayen, K. S. Meel, D. Myrisiotis, A. Pavan, and N. Vinodchandran. On approximating total variation distance. *arXiv preprint arXiv:2206.07209*, 2022.

[7] S. H. Cheung, L.-X. Zhang, F. Hu, and W. S. Chan. Covariate-adjusted response-adaptive designs for generalized linear models. *Journal of Statistical Planning and Inference*, 149:152–161, 2014.

[8] J. A. DiMasi, H. G. Grabowski, and R. W. Hansen. Innovation in the pharmaceutical industry: new estimates of r&d costs. *Journal of health economics*, 47:20–33, 2016.

[9] H. Du, Z. Li, D. Niyato, J. Kang, Z. Xiong, D. I. Kim, et al. Enabling ai-generated content (aigc) services in wireless edge networks. *arXiv preprint arXiv:2301.03220*, 2023.

[10] Food and Drug Administration. Adaptive designs for clinical trials of drugs and biologics: Guidance for industry. FDA Guidance Document, Silver Spring, MD, 2019.

[11] S. Fujimoto, H. Hoof, and D. Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018.

[12] S. Fujimoto, D. Meger, D. Precup, O. Nachum, and S. S. Gu. Why should i trust you, bellman? the bellman error is a poor replacement for value error. *arXiv preprint arXiv:2201.12417*, 2022.

[13] I. S. T. C. Group et al. The international stroke trial (ist): a randomised trial of aspirin, subcutaneous heparin, both, or neither among 19 435 patients with acute ischaemic stroke. *The Lancet*, 349(9065):1569–1581, 1997.

[14] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. Pmlr, 2018.

[15] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

[16] C. Harshaw, F. Sävje, D. A. Spielman, and P. Zhang. Balancing covariates in randomized experiments with the gram–schmidt walk design. *Journal of the American Statistical Association*, 119(548):2934–2946, 2024.

[17] I. Hatfield, A. Allison, L. Flight, S. A. Julious, and M. Dimairo. Adaptive designs undertaken in clinical research: a review of registered clinical trials. *Trials*, 17:1–13, 2016.

[18] F. Hu, W. F. Rosenberger, and L.-X. Zhang. Asymptotically best response-adaptive randomization procedures. *Journal of Statistical Planning and Inference*, 136(6):1911–1922, 2006.

[19] O. N. Klein, A. Hüyük, R. Shamir, U. Shalit, and M. van der Schaar. Towards regulatory-confirmed adaptive clinical trials: Machine learning opportunities and solutions. *arXiv preprint arXiv:2503.09226*, 2025.

[20] V. L. Mahan et al. Clinical trial phases. *International Journal of Clinical Medicine*, 5(21):1374, 2014.

[21] K. Matsuura, J. Honda, I. El Hanafi, T. Sozu, and K. Sakamaki. Optimal adaptive allocation using deep reinforcement learning in a dose-response study. *Statistics in Medicine*, 41(7):1157–1171, 2022.

[22] S. Narvekar, B. Peng, M. Leonetti, J. Sinapov, M. E. Taylor, and P. Stone. Curriculum learning for reinforcement learning domains: A framework and survey. *Journal of Machine Learning Research*, 21(181):1–50, 2020.

[23] J. J. Park, K. Thorlund, and E. J. Mills. Critical concepts in adaptive clinical trials. *Clinical epidemiology*, pages 343–351, 2018.

[24] S. J. Pocock and R. Simon. Sequential treatment assignment with balancing for prognostic factors in the controlled clinical trial. *Biometrics*, pages 103–115, 1975.

[25] D. S. Robertson, K. M. Lee, B. C. López-Kolkovska, and S. S. Villar. Response-adaptive randomization in clinical trials: from myths to practical considerations. *Statistical science: a review journal of the Institute of Mathematical Statistics*, 38(2):185, 2023.

[26] W. F. Rosenberger and O. Sverdlov. Handling covariates in the design of clinical trials. 2008.

[27] W. F. Rosenberger, A. Vidyashankar, and D. K. Agarwal. Covariate-adjusted response-adaptive designs for binary response. *Journal of biopharmaceutical statistics*, 11(4):227–236, 2001.

[28] J. Shreffler and M. R. Huecker. *Hypothesis Testing, P Values, Confidence Intervals, and Significance.* StatPearls [Internet]. StatPearls Publishing, Treasure Island (FL), updated 2023 mar 13 edition, 2023. Available from: https://www.ncbi.nlm.nih.gov/books/NBK557421/.

[29] B. Sukhija, S. Coros, A. Krause, P. Abbeel, and C. Sferrazza. Maxinforl: Boosting exploration in reinforcement learning through information gain maximization. *arXiv preprint arXiv:2412.12098*, 2024.

[30] K. Suresh and S. Chandrashekara. Sample size estimation and power analysis for clinical research studies. *Journal of Human Reproductive Sciences*, 5(1):7–13, Jan 2012. Retraction in: J Hum Reprod Sci. 2015 Jul-Sep;8(3):186. doi: 10.4103/0974-1208.165154.

[31] R. S. Sutton, A. G. Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[32] P. F. Thall and J. K. Wathen. Practical bayesian adaptive randomization in clinical trials. *European Journal of Cancer*, 43(5):859–866, 2007.

[33] The American Cancer Society medical and editorial content team. Types and phases of clinical trials. `https://www.cancer.org/cancer/managing-cancer/making-treatment-decisions/clinical-trials/what-you-need-to-know/phases-of-clinical-trials.html#:~:text=Treatments%20that%20have%20been%20shown,against%20the%20current%20standard%20treatment`, 2020. Last Revised: August 18, 2020.

[34] J. C. Urenda, O. Csiszár, G. Csiszár, J. Dombi, O. Kosheleva, V. Kreinovich, and G. Eigner. Why squashing functions in multi-layer neural networks. In *2020 IEEE international conference on systems, man, and cybernetics (SMC)*, pages 1705–1711. IEEE, 2020.

[35] Y. Varatharajah and B. Berry. A contextual-bandit-based approach for informed decision-making in clinical trials. *Life (Basel)*, 12(8):1277, 2022.

[36] S. S. Villar and W. F. Rosenberger. Covariate-adjusted response-adaptive randomization for multi-arm clinical trials using a modified forward looking gittins index rule. *Biometrics*, 74(1):49–57, 2018.

[37] W. Xiong, J. Roy, H. Liu, and L. Hu. Leveraging machine learning: Covariate-adjusted bayesian adaptive randomization and subgroup discovery in multi-arm survival trials. *Contemporary Clinical Trials*, 142:107547, 2024.

[38] L.-X. Zhang, F. Hu, S. H. Cheung, and W. S. Chan. Asymptotic properties of covariate-adjusted response-adaptive designs. 2007.

[39] Y. Zhao, M. R. Kosorok, and D. Zeng. Reinforcement learning design for cancer clinical trials. *Statistics in medicine*, 28(26):3294–3315, 2009.