A Markov Decision Process
for Response-Adaptive Randomization in
Clinical Trials

David Merrell*, Thevaa Chandereng†, Yeonhee Park‡

*Department of Computer Sciences, University of Wisconsin - Madison. †Department of

Biostatistics, Columbia University. ‡ Department of Biostatistics and Medical Informatics,

University of Wisconsin - Madison.

dmerrell@cs.wisc.edu, tc3123@cumc.columbia.edu, ypark56@wisc.edu

Summary

In clinical trials, response-adaptive randomization (RAR) has the appealing ability to assign
more subjects to better-performing treatments based on interim results. The traditional RAR
strategy alters the randomization ratio on a patient-by-patient basis; this has been heavily criti-
cized for bias due to time-trends. An alternate approach is blocked RAR, which groups patients
together in blocks and recomputes the randomization ratio in a block-wise fashion; the final anal-
ysis is then stratified by block. However, the typical blocked RAR design divides patients into
equal-sized blocks, which is not generally optimal.

This paper presents TrialMDP, an algorithm that designs two-armed blocked RAR clinical
trials. Our method differs from past approaches in that it optimizes the *size and number of blocks*
as well as their treatment allocations. That is, the algorithm yields a policy that adaptively

*To whom correspondence should be addressed.

chooses the *size and composition* of the next block, based on results seen up to that point in the trial. TRIALMDP is related to past works that compute optimal trial designs via dynamic programming.

The algorithm maximizes a utility function balancing *(i)* statistical power, *(ii)* patient outcomes, and *(iii)* the number of blocks. We show that it attains significant improvements in utility over a suite of baseline designs, and gives useful control over the tradeoff between statistical power and patient outcomes. It is well suited for small trials that assign high cost to failures.

We provide TRIALMDP as an R package: `https://github.com/dpmerrell/TrialMDP`.

Key words: Adaptive randomization; Clinical trial; Dynamic programming; Markov decision process; Reinforcement learning

1. INTRODUCTION

Randomization is a common technique used in clinical trials to eliminate potential bias and confounders in a patient population. Most clinical trials utilize fixed randomization, where the probability of assigning subjects to a treatment group is kept fixed throughout the trial. Response-adaptive randomization (RAR) designs were developed due to the captivating benefit of increasing the probability of assigning patients to more promising treatments, based on the responses of prior patients. A big downside for RAR designs is that the time between treatment and outcome must be short, in order to inform future patients' randomization.

Traditional RAR designs recompute the randomization ratio on a patient-by-patient basis (Thall and Wathen, 2007), usually after a burn-in period of fixed randomization. However, traditional RAR designs have been widely criticized (Karrison *and others*, 2003). Traditional RAR designs induce bias due to temporal trends in clinical trials. Temporal trends are especially likely to occur in long duration trials. Patients' characteristics might be completely different through-

out the trial or even at the beginning and end of the trial (Proschan and Evans, 2020). However, standard RAR analyses assume that the sequence of patients who arrive for entry into the trial represents samples drawn at random from two homogeneous populations, with no drift in the probabilities of success (Proschan and Evans, 2020; Chandereng and Chappell, 2020). This assumption is usually violated. For example, in the BATTLE lung cancer elimination trials (Liu and Lee, 2015), more smokers enrolled in the latter part of the trial than at the beginning.

Despite this serious flaw, there is not much literature to address the temporal trend issue in RAR designs. Villar et al. explored the hypothesis testing procedure adjusting for covariates for correcting type-I error inflation and the effect on power in RAR designs with temporal trend effects added to the model for two-armed and multi-armed trials (Villar *and others*, 2018). Karrison *and others* (2003) introduced a stratified group-sequential method with a simple example of altering the randomization ratio to address this issue. Chandereng and Chappell (2019) further examined the operating characteristics of the blocked RAR approach for two treatment arms proposed by Karrison *and others* (2003). They concluded that blocked RAR provides a good trade-off between ethically assigning more subjects to the better-performing treatment group and maintaining high statistical power. They also suggested using a small number of blocks since large numbers of blocks have low statistical power. However, Chandereng and Chappell (2019) designed trials with equal-sized blocks, which is not generally optimal.

Other works formulate adaptive trial design as a Multi-Armed Bandit Problem (MABP), employing ideas that are often associated with reinforcement learning—e.g., sequential decision-making and regret minimization. These entail sophisticated algorithms, such as Gittins index computations (Villar *and others*, 2015*a,b*) and dynamic programming (Hardwick and Stout, 1995, 1999, 2002). These works have important limitations. The Gittins index approaches of Villar *and others* assume either *(i)* a fully sequential trial with similar weaknesses to traditional RAR or *(ii)* a blocked trial with equal-sized blocks. The dynamic programming algorithms of Hardwick and

Stout yield allocation rules that *(i)* are deterministic, *(ii)* are fully sequential, or *(iii)* assume a blocked trial with a fixed number of blocks. At the time, Hardwick and Stout's approaches were also limited by computer speed and memory, which have improved famously over the years.

This paper presents TRIALMDP, an algorithm that designs blocked RAR trials. TRIALMDP is most closely related to the MABP-based approaches mentioned above. However, it models a blocked RAR trial as a Markov Decision Process (MDP), a generalization of the MABP. It relies on a dynamic programming algorithm, similar to those of Hardwick and Stout. However, our method differs in that it optimizes the *size and number of blocks* as well as their treatment allocations. That is, the algorithm yields a policy that adaptively chooses the *size and composition* of the next block, based on results seen up to that point in the trial. The current version of TRIALMDP is tailored for two-armed trials with binary outcomes. Future versions may permit a more general class of trials.

Our paper has the following structure. In Section 2, we describe our problem formulation and algorithmic solution. In Section 3, we compare TRIALMDP's designs with other designs that have been widely used in clinical trials. We use our proposed method to redesign a phase II trial in Section 3.2. We discuss TRIALMDP's limitations and potential improvements in Section 4. Our Supplementary Materials include appendices that justify some of our mathematical and algorithmic choices.

2. PROPOSED METHOD

2.1 *Problem formulation*

Class of trials. In this paper we focus on blocked RAR trials with two arms and binary outcomes. We label the arms A and B ("treatment" and "control", respectively) and outcomes 0 and 1 ("failures" and "successes"). A trial has access to some number of available patients, N. The trial proceeds in K blocks. We require that all results from the current block are observed before
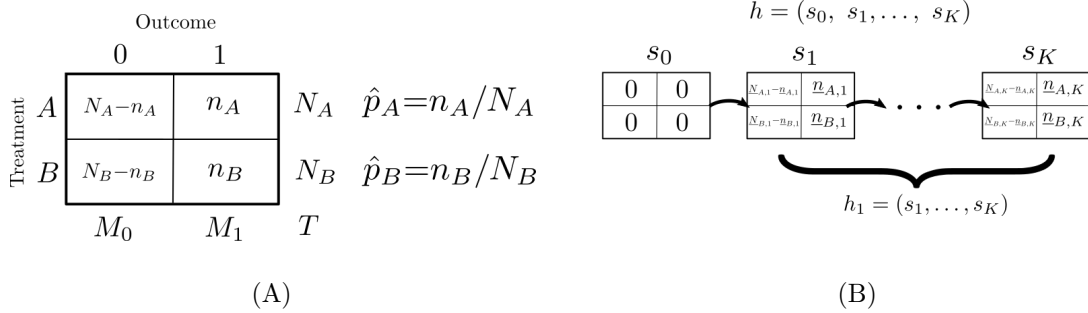
Fig. 1. (A) Contingency table notation. (B) Trial history notation. A history h is a sequence of cumulative contingency tables, $(s_0, \ldots, s_K)$. A subscript h_k indicates a history's k^{th} *suffix*.

the next block begins. Importantly, we allow K to adapt as the trial progresses. This gives the trial useful kinds of flexibility. In general, a trial may attain better characteristics if it permits differently-sized blocks.

Let p_A, p_B denote the treatments' success probabilities. We assume a frequentist test is performed at the end of the trial, with the following null and alternative hypotheses $\mathcal{H}_0, \mathcal{H}_A$:

$$\mathcal{H}_0 : \ p_A = p_B \qquad \mathcal{H}_A : \ p_A > p_B$$

We focus specifically on the one-sided Cochran-Mantel-Haenzsel (CMH) test, which is well-suited for stratified observations; in our setting, the strata are blocks of patients. It has been argued that blocked RAR trials with CMH tests are more robust to temporal trend effects than, e.g., traditional RAR trials with chi-square tests (Chandereng and Chappell, 2019).

Notation: tables and histories. We establish some notation for clarity. A 2×2 contingency table has the following attributes: N_A and N_B, the numbers of patients assigned to each treatment; n_A and n_B, the numbers of successes for each treatment; M_0 and M_1, the total numbers of failures and successes; and T, the total number of outcomes recorded in the table. Symbols $\hat{p}_A$, $\hat{p}_B$ represent point estimates of the treatment success probabilities. See Figure 1 for illustration.

Each block of the trial has its own contingency table with corresponding quantities. We use

a subscript to indicate the block. For example, the kth block of the trial has its own table with quantities $N_{A,k}$, $n_{A,k}$, T_k, and so on.

At any point we can summarize the state of the trial in a 2×2 contingency table, s, of *cumulative* results. That is, s contains *all* of the trial's observations up to that point; or, put another way, s is the sum of all preceding block-wise contingency tables. We typically refer to s as a *state*. We use an underline to indicate a quantity computed from a state. For example, after completing k blocks we have quantities $\underline{N}_{A,k}=\sum_{i=1}^{k} N_{A,i}$; $\underline{n}_{A,k}=\sum_{i=1}^{k} n_{A,i}$; $\underline{\hat{p}}_{A,k}=\underline{n}_{A,k}/\underline{N}_{A,k}$; and so on.

The sequence of states occupied by a trial forms a *trial history* $h=(s_0,\ldots,s_K)$, where s_0 is always the empty contingency table and s_K always has $\underline{T}_K=N$ observations. We use a subscript to denote the *suffix* of a history. For example, $h_k=(s_k,\ldots,s_K)$ is the sequence of states *after* the kth block of the trial. It is useful to think of a history as a random object, subject to uncertainty in the patient outcomes and the values of p_A, p_B.

Utility function.   We aim to design blocked RAR trials that balance *(i)* statistical power and *(ii)* patient outcomes. We also recognize that each additional block entails a cost in time and other overhead. As such, we wish to avoid an excessive number of blocks. We formalize these goals with the following utility function:

$$U(h) = V(h) \; - \; \lambda_F{\cdot}F(h) \; - \; \lambda_K{\cdot}K(h) \tag{2.1}$$

where $V(h)$ is a proxy for the trial's statistical power; $F(h)$ measures the number of failures; and $K(h)$ is the number of blocks. This utility function promotes a high statistical power while penalizing failures and blocks. The coefficients λ_F and λ_K control the relative importance of patient outcomes and blocks, respectively.

The functions V, F, and K have the following forms:

$$V(h) = \frac{1}{N} \sum_{i=1}^{K} \frac{w_i}{\frac{1}{2}(\underline{\hat{p}}_{A,i}+\underline{\hat{p}}_{B,i})\frac{1}{2}(\underline{\hat{q}}_{A,i}+\underline{\hat{q}}_{B,i})} \qquad F(h) = \frac{1}{N}(\underline{N}_{A,K} - \underline{N}_{B,K})(\underline{\hat{p}}_{B,K} - \underline{\hat{p}}_{A,K})$$

$$K(h) = K$$

Function K simply returns the number of blocks in the trial history. Function F quantifies bad patient outcomes (i.e., failures) as a fraction of all patients. It is a function only of the final state, s_K, and becomes small when the estimates $\hat{\underline{p}}_{A,K}$ and $\hat{\underline{p}}_{B,K}$ are close.

Function V serves as a proxy for the trial's statistical power. It is crafted such that maximizing V also maximizes the power of the Cochran-Mantel-Haenzsel test (Cochran, 1954) to an acceptable approximation. Each $w_i = N_{A,i}N_{B,i}/(N_{A,i}+N_{B,i})$ is the *harmonic mean* of that block's treatment allocations. V takes larger values when the allocations are balanced; and when p_A, p_B are close to each other, and far from $\frac{1}{2}$. The factor $\frac{1}{N}$ makes V consistent across trials with differing sample sizes. See Appendix A of the Supplementary Materials for a more detailed justification of V.

Markov Decision Process formulation. In our effort to maximize the expected utility (Equation 2.1), we find it useful to model a blocked RAR trial as a *Markov Decision Process* (MDP). An MDP is a simple model of sequential decision-making. It consists of an agent navigating a *state space*. At each time-step, the agent chooses an *action*. Given the agent's current state and chosen action, the agent *transitions* to a new state and collects a *reward*. In general the transition is stochastic, governed by a *transition distribution*. One *solves* an MDP by obtaining a *policy* that maximizes the expected total reward. We refer the reader to Chapter 38 of Lattimore and Szepesvari's text for detailed information about MDPs (Lattimore and Szepesvari, 2020).

We model a blocked RAR trial as an MDP with the following components:

- *State space.* In our setting the state space S consists of every possible 2×2 contingency table with $\leqslant N$ observations. We can order the states by their numbers of observations. We let S_i denote the subset of S containing tables with exactly i observations. The trial always begins at the empty contingency table in S_0 and terminates at some table in S_N. The state
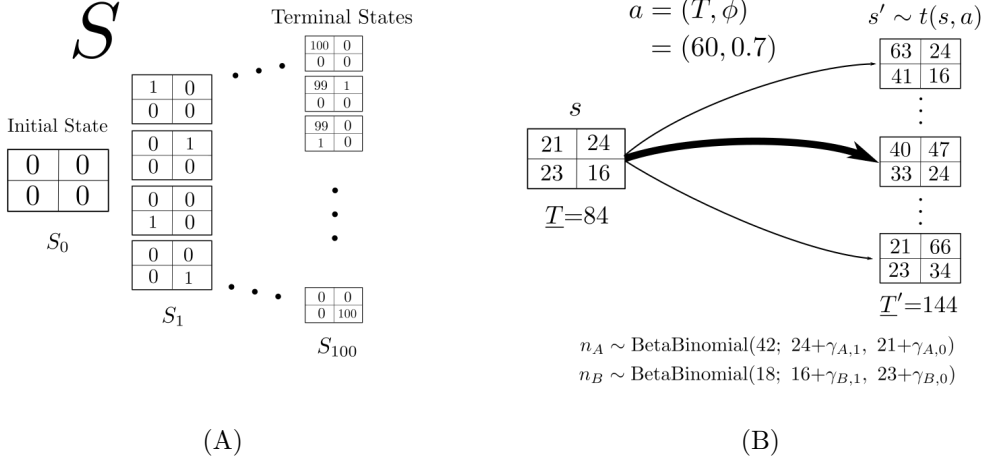
Fig. 2. (A) State space S. At any point, the state of the trial is summarized by a contingency table of all observations. We can order the set of all contingency tables by their numbers of observations, $\underline{T}$. The trial begins with the empty table in S_0; the trial ends when it reaches a state in S_N (in this example $N=100$). (B) Transition distribution. In this example, current state s and action $a=(60, 0.7)$ induce a distribution $s' \sim t(s, a)$ for the next state. The next state necessarily has $\underline{T}'=144=84+60$. Its entries are governed by Beta-Binomial distributions, parameterized by the entries of the current contingency table.

space grows quickly with N, $|S|=O(N^4)$. Figure 2(A) illustrates S for $N=100$.

- *Actions.* With each block of the trial we choose an action $a=(T, \phi)$, the block's *size* and *allocation.* Suppose we have completed k blocks; then T may take any integer value from 1 to $N-\underline{T}_k$. The allocation ϕ is the fraction of patients assigned to treatment A in this block. We constrain ϕ to a finite set of possible values, Φ. For example, $\Phi = \{0.2, 0.3, \ldots, 0.8\}$. Importantly, exactly $T \cdot \phi$ patients (rounded to the nearest integer) are assigned to treatment A. In other words, patients are randomized to treatments "without replacement." Contrast this with other randomized designs—traditional RAR, blocked RAR, etc.—that assign each patient to A with independent probability ϕ. For example, action $(T=60, \phi=0.7)$ implies that the next block will treat 60 patients, assigning exactly $T \cdot \phi=42$ of them to treatment A and 18 to treatment B.

  We let $\mathcal{A}$ denote the set of all actions, and $\mathcal{A}_s$ denote actions available at state s.

- *Transition distributions.* Given the current contingency table s_i and the chosen block de-

sign $a_i = (T_i, \phi_i)$, the next contingency table s_{i+1} is randomly distributed. This randomness consists of two parts: *(i)* the stochasticity of patient outcomes given the true success probabilities p_A and p_B, and *(ii)* our uncertainty about the values of p_A and p_B. Given the true values for p_A and p_B, the numbers of successes $n_{A,i+1}$ and $n_{B,i+1}$ for this block would have Binomial distributions:

$$n_{A,i+1}|p_A \sim \text{Binomial}\left(T_i{\cdot}\phi_i,\ p_A\right) \qquad n_{B,i+1}|p_B \sim \text{Binomial}\left(T_i{\cdot}(1-\phi_i),\ p_B\right).$$

However, we only have imperfect knowledge of p_A and p_B, encoded in the entries of the current table s_i. We use Beta distributions to describe this uncertainty about p_A and p_B:

$$p_A \sim \text{Beta}(\underline{n}_{A,i}{+}\gamma_{A1},\ \underline{N}_{A,i}{-}\underline{n}_{A,i}{+}\gamma_{A0}) \qquad p_B \sim \text{Beta}(\underline{n}_{B,i}{+}\gamma_{B1},\ \underline{N}_{B,i}{-}\underline{n}_{B,i}{+}\gamma_{B0})$$

where each γ_* is a smoothing hyperparameter typically set to 1. Together, these two sources of randomness assign independent Beta-Binomial probabilities to $n_{A,i+1}$ and $n_{B,i+1}$, which in turn define the distribution for s_{i+1}. See Figure 2(B) for illustration. We sometimes use the notation $s_{i+1} \sim t(s_i, a_i)$ to indicate the *transition distribution* for s_{i+1}, given s_i and a_i.

- *Rewards.* Given the current state s_i and the chosen action a_i, the trial transitions to state s_{i+1} and receives a reward $R(s_i, a_i, s_{i+1})$. In an MDP the goal is to maximize expected total reward. Recall, however, that our ultimate goal is to maximize the expected utility U (Equation 2.1). We craft a reward function R consistent with U, as follows:

$$R(s_i, a_i, s_{i+1}) = \begin{cases} \frac{1}{N} \frac{w_{i+1}}{\frac{1}{2}(\hat{\underline{p}}_{A,i+1}+\hat{\underline{p}}_{B,i+1})\frac{1}{2}(\hat{\underline{q}}_{A,i+1}+\hat{\underline{q}}_{B,i+1})} \ - \ \lambda_K & s_{i+1} \notin S_N \\[2em] \frac{1}{N} \frac{w_{i+1}}{\frac{1}{2}(\hat{\underline{p}}_{A,i+1}+\hat{\underline{p}}_{B,i+1})\frac{1}{2}(\hat{\underline{q}}_{A,i+1}+\hat{\underline{q}}_{B,i+1})} \ - \ \lambda_K \ - \ \lambda_F{\cdot}F(s_{i+1}) & s_{i+1} \in S_N \end{cases}$$

$$(2.2)$$

The total reward for a trial history is identical to the utility (Equation 2.1) of that trial history. With each block, the reward function produces that block's contribution to the total utility. This includes the block's term for V; the block's cost λ_K; and the final failure penalty $F(s_{i+1})$ when s_{i+1} is terminal.

Notice that our particular R is a function only of a_i and s_{i+1}. We sometimes write $R(a_i, s_{i+1})$ for compactness.

- *Policy.* A policy is a function $\pi : S \to \mathcal{A}$ mapping each state in the MDP to an action. In our setting policies are *trial designs*. For each state s_i in the trial, a policy dictates the design of the trial's next block: $\pi(s_i) = (T_{i+1}, \phi_{i+1})$. Our MDP is solved by the *optimal* policy π^* satisfying

$$\pi^*(s_i) = \mathrm{argmax}_\pi \, \mathbb{E}_{h_i|s_i,\pi} \left[U(h_i) \right] \qquad \forall s_i \in S.$$

We let $U^*(s_i) = \mathbb{E}_{h_i|s_i,\pi^*} \left[U(h_i) \right]$ denote the corresponding maximal value at each state $s_i \in S$.

Casting our problem into the MDP framework helps us design algorithmic solutions. Our particular MDP lends itself to a straightforward dynamic programming approach, since there are no cycles in its directed graph of possible transitions.

2.2 *Solution via dynamic programming*

The MDP described in Section 2.1 can be solved by a relatively simple dynamic programming algorithm. This makes our method a close relative of past dynamic programming approaches for trial design (Woodroofe and Hardwick, 1990; Hardwick and Stout, 1995, 1999, 2002). However, our method differs from them in an important respect: we seek to maximize an objective that is a *function of the trial history*, and not just a function of the final state. Concretely, our objective function (Equation 2.1) includes $V(h)$ and $K(h)$, which are functions of block-wise attributes. Formulating the problem as an MDP gives us the flexibility to consider such an objective.

Recurrence relations. Like any dynamic programming algorithm, ours divides the problem at hand into subproblems and solves them in an order that efficiently reuses computation. This dependence between subproblems is defined by a set of recurrence relations. In our case we have

a single recurrence based on the Bellman equation (Lattimore and Szepesvari, 2020):

$$U^*(s) = \begin{cases} \max_a \left\{ \mathbb{E}_{s' \sim t(s,a)} \left[R(s, a, s') + U^*(s') \right] \right\} & s \notin S_N \\ \\ 0 & s \in S_N \end{cases} \qquad (2.3)$$

The algorithm computes this recurrence at every state in S, iterating through the state space in order of *decreasing* $\underline{T}$. In other words the algorithm evaluates the recurrence at each state in S_N, S_{N-1}, and so on, until it finally computes $U^*(s_0)$ for the the empty table $s_0 \in S_0$ and terminates. At each state $s \notin S_N$ the algorithm also tabulates the maximizing action a^*. This table of optimal actions is the algorithm's most important output, as it constitutes π^*, the optimized trial design. Figure 3 illustrates the algorithm in detail with pseudocode.

The trial design (i.e., policy) yielded by this recurrence is guaranteed to maximize the expected utility (subject to the MDP formulation described in Section 2.1), since our optimization problem has the *optimal substructure property*. See Appendix B of the Supplementary Materials for more discussion and a proof of optimal substructure.

Computational expense. At a high level TRIALMDP is a nested loop over every possible state, action, and transition. For each state the algorithm stores a set of values, along with the optimal action. Hence the algorithm uses $O(|S|) = O(N^4)$ space. The number of possible actions and transitions varies between states; summing across all states yields total time cost $O(|\Phi|N^7)$, where Φ is the set of allocation fractions mentioned in Section 2.1.

These complexities apply if we allow the algorithm to consider every possible state and action. However, there are practical ways to *prune away* states and actions, attaining much lower computational cost without sacrificing much utility. Introducing a *minimum block size* parameter $T_{\min}$ eliminates all of the states in $S_1, \ldots, S_{T_{\min}-1}$ and $S_{N-T_{\min}+1}, \ldots S_{N-1}$; and reduces the number of possible actions at each remaining state. An additional *block increment* parameter κ further constrains the algorithm to states where T is an integer multiple of κ, resulting in a "coarsened" state space. These parameters reduce the algorithm's space and time cost to

---

Algorithm 1 TRIALMDP

1: **procedure** MAINLOOP$(N, \lambda_F, \lambda_K)$
2:     initialize tables U, A
3:     **for** $s \in S_N$ **do**
4:         U[s] = 0
5:     **for** $s \notin S_N$ **do**
6:         U[s] = $-\infty$
7:         **for** $a \in \mathcal{A}_s$ **do**
8:             u = 0
9:             **for** $(p, s') \in t(s, a)$ **do**
10:                 u+=$p \cdot \big\{ R(a, s', N, \lambda_F, \lambda_K)$
11:                                       $+ \text{U}[s'] \big\}$
12:             **if** u > U[s] **then**
13:                 U[s] = u; A[s] = a
14:     **return** U, A

function $R(a, s', N, \lambda_F, \lambda_K)$
    $r = 0$
    $w = a.N_A \cdot a.N_B / (a.N_A + a.N_B)$
    $\hat{p}_A = (s'.n_A + 1)/(s'.N_A + 2)$
    $\hat{p}_B = (s'.n_B + 1)/(s'.N_B + 2)$
    $\hat{q}_A = 1 - \hat{p}_A$
    $\hat{q}_B = 1 - \hat{p}_B$
    $r \mathrel{+}= 4 \cdot w / (N \cdot (\hat{p}_A + \hat{p}_B)(\hat{q}_A + \hat{q}_B))$
    if $s' \in S_N$ **then**
        $r \mathrel{-}= \lambda_F \cdot (s'.N_A - s'.N_B)$
                    $\cdot (\hat{p}_B - \hat{p}_A)/N$
    $r \mathrel{-}= \lambda_K$
    return r

Fig. 3. TRIALMDP algorithm pseudocode. The algorithm populates tables U and A with optimal utilities and actions, respectively. Tables U and A are indexed by states; i.e., U[s] yields the utility for state s. The for-loop on line 5 iterates through states in order of decreasing $\underline{T}$. The for-loop on line 7 iterates through all possible actions for the current state; and the loop on line 9 computes the expectation of U for the current state and action. Function R evaluates the reward function given by Equation 2.2. We use "dot notation" to access the attributes of states and actions; e.g., $s'.N_A$ yields N_A for state s'.

$O((N - T_{\min})^4/\kappa)$ and $O((N - T_{\min})^7/\kappa^2)$, respectively. See Appendix C of the Supplementary Materials for derivations. We typically set $T_{\min} = N/8$ and $\kappa = 2$. Unless specified otherwise, we use $\Phi = \{0.2, 0.3, \ldots, 0.8\}$. These settings yielded trials with competitive characteristics, without incurring undue computational expense during the evaluations of Section 3.

Empirically, we observe a time cost of 5; 2,300; and 23,000 seconds for trials with 40, 100, and 140 patients respectively. These measurements used a single-threaded implementation of TRIALMDP, on a laptop with Intel 1.1GHz CPUs.

3. EVALUATION

3.1 *Simulation study*

We performed a simulation study to compare TRIALMDP against established trial designs. At each point in a grid of values for $\lambda_F, \lambda_K, p_A$, and p_B, we ran 10,000 simulated trials using

TRIALMDP and a suite of baseline designs. The baselines included *(i)* a 1:1, fixed randomization design; *(ii)* a traditional Response-Adaptive Randomized (RAR) design; and *(iii)* a blocked RAR design.

For null scenarios with $p_A = p_B$, we chose an arbitrary sample size of $N=100$. For alternative scenarios with $p_A > p_B$, we chose N large enough for a 1:1 design to attain a power of 0.8. See Tables 1 and 2 for the exact values of N, p_A, and p_B used in our simulated scenarios.

The traditional RAR baseline used a 1:1 randomization ratio for the first 25% of patients, and adaptive randomization thereafter according to the procedure used by Rosenberger *and others* (2001). That is, the kth patient was assigned to treatment A with probability

$$\xi_k = \frac{\sqrt{\hat{\underline{p}}_{A,k-1}}}{\sqrt{\hat{\underline{p}}_{A,k-1}} + \sqrt{\hat{\underline{p}}_{B,k-1}}}. \tag{3.4}$$

The blocked RAR baseline used two blocks of equal size. The first block used a 1:1 randomization ratio; the second block used the same randomization given by Equation 3.4. This agrees with the blocked RAR procedure described by Chandereng and Chappell (2019).

We used TRIALMDP to generate trial designs over a grid of parameter settings: $(\lambda_F, \lambda_K) \in \{2, 3, 4, 5\} \times \{0.01, 0.025, 0.05, 0.1\}$. Each parameter setting implies a different balance between statistical power, patient outcomes, and the number of blocks.

We simulated 10,000 trials for every scenario p_A, p_B, for each baseline design, and for each TRIALMDP parameter setting. As an initial sanity check we visualized some trial histories to see whether the designs behaved as expected. Figure 4 shows some examples. TRIALMDP always chose 1:1 allocation for the first block, increasing the allocation to A in subsequent blocks when $p_A > p_B$. As λ_F increased, the designs reliably increased allocation to A, in agreement with our expectations. The baseline designs also yielded trial histories that agreed with our expectations.

Recall that TRIALMDP is supposed to optimize the utility function (Equation 2.1) in expectation. If this were true, we would expect our designs to attain higher utility than the others,

averaged over the simulated histories. To verify this we computed the utility for every simulated history and for every design, and tabulated the resulting averages.

Table 1 shows some representative results from the alternative scenarios. These results employed TRIALMDP with λ_F=0.4 and λ_K=0.01. Under these particular parameter settings TRIALMDP attained slightly lower power than the other designs, but its superior patient outcomes gave it the greatest utility across all scenarios. Indeed, we found that our algorithm does typically achieve higher average utility than the baseline designs *(i)* under the alternative hypothesis and *(ii) as long as λ_F is sufficiently large.* When λ_F is not large enough, our designs have highest utility *among the adaptive designs*, but the 1:1 design is mathematically guaranteed to attain highest utility. We show this in Appendix D of the Supplementary Materials.

We highlight the fact that TRIALMDP assigned many more patients to the superior treatment on average, in all the scenarios of Table 1. Furthermore, it did so reliably. The 5%-ile for $N_A - N_B$ is higher for TRIALMDP than for any other adaptive design, in every alternative scenario.

It is also important to note that TRIALMDP's design yielded slightly biased estimates of the effect size in the alternative scenarios. We hypothesize that this bias—on the order of 0.01— stems from the rapidly changing randomization ratio prescribed by TRIALMDP. The user ought to weigh this against other matters, such as vastly improved patient outcomes, when considering TRIALMDP.

Table 2 shows the corresponding results for null scenarios. Notice that in some cases TRIALMDP's designs showed somewhat inflated type-I error. The percentiles of $N_A - N_B$ show that TRIALMDP is more prone to creating an imbalanced allocation under the null hypothesis. Another salient observation is the relative decrease in utility for *all* of the adaptive designs. This has a simple explanation. Under the null hypothesis, a 1:1 design always has optimal utility. A 1:1 design attains maximal $V(h)$ and minimal $K(h)$; and under the null hypothesis, $F(h)=0$ for any design. Hence, every adaptive design will yield lower utility than the 1:1 design.
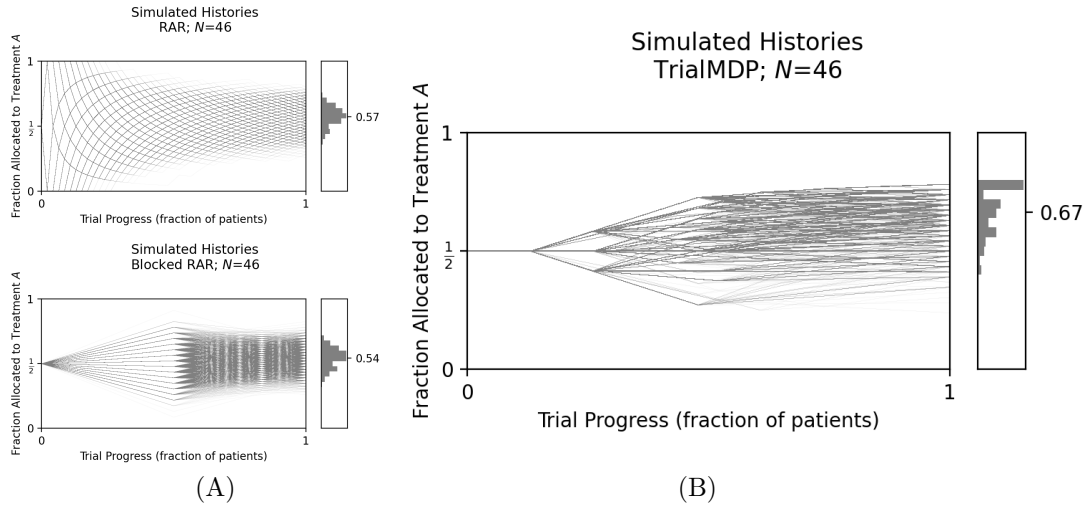
Fig. 4. Simulated trial histories. Each plot traces the treatment allocation of 10,000 simulated trials. Histograms on the right give distributions of final allocations and report the mean. For each of these plots, $(p_A, p_B)=(0.4, 0.1)$. (A) Histories for RAR and blocked RAR trial designs. (B) Histories for a TRIALMDP design, with parameter settings $\lambda_F=4.0$ and $\lambda_K=0.01$. Under these specific settings TRIALMDP allocates many more patients on average to the superior treatment.

p_A	p_B	N	Power (CMH test) 1:1	RAR	BRAR	MDP	Effect Bias RAR	BRAR	MDP	N_A-N_B (5%, 95%) RAR	BRAR	MDP	$K(h)$ MDP	Utility Z-score RAR	BRAR	MDP
0.3	0.1	94	0.79	0.78	0.79	0.78	0.00	0.00	0.01	16.94 (-4, 36)	11.22 (-8, 30)	19.88 (-2, 50)	3.26	-60.25	6.23	8.40
0.4	0.1	46	0.78	0.75	0.77	0.74	0.00	0.00	0.01	6.80 (-6, 18)	3.98 (-8, 16)	15.26 (0, 26)	3.87	-9.18	2.65	8.17
	0.2	124	0.80	0.78	0.79	0.77	0.00	0.00	0.01	17.03 (-6, 42)	11.18 (-10, 32)	31.77 (-4, 66)	3.50	-104.45	6.10	11.64
0.5	0.3	144	0.80	0.80	0.78	0.76	0.00	0.00	0.01	14.20 (-10, 38)	9.54 (-12, 32)	42.23 (-6, 76)	3.41	-157.52	5.11	14.80
0.7	0.4	62	0.79	0.77	0.77	0.73	0.00	0.00	0.01	6.96 (-8, 22)	4.60 (-10, 18)	23.03 (0, 34)	3.85	-17.25	2.89	11.22
	0.5	144	0.81	0.79	0.79	0.77	0.00	0.00	0.01	9.22 (-12, 30)	6.13 (-14, 28)	42.33 (-6, 76)	3.34	-174.98	3.16	14.78
0.9	0.6	46	0.79	0.78	0.78	0.76	0.00	0.00	0.01	3.72 (-8, 16)	2.55 (-8, 14)	13.43 (0, 26)	3.50	-9.56	1.62	6.89
	0.7	94	0.80	0.80	0.80	0.79	0.00	0.00	0.00	4.53 (-12, 20)	3.08 (-14, 20)	19.17 (-2, 50)	3.22	-73.27	1.32	7.80

Table 1. Simulation study alternative scenarios. Labels RAR, BRAR, and MDP refer to adaptive trials designed by traditional RAR, blocked RAR, and TRIALMDP, respectively. The label 1:1 refers to a fixed randomization trial with one-to-one allocation. The "Effect Bias" multicolumn reports the average difference between estimated effect size and true effect size. The "N_A-N_B (5%, 95%)" multicolumn shows the difference in patient allocation between treatments; it reports the mean, with the 5%-ile and 95%-ile in parentheses. $K(h)$ shows the average number of blocks. It only varies for TRIALMDP; $K(h)=N$ for RAR and $K(h)=2$ for BRAR in all scenarios. The "Utility Z-score" multicolumn reports gain in utility relative to the 1:1 trial design, computed as $Z=(\mu_1 - \mu_2)/\sqrt{(\sigma_1^2 + \sigma_2^2)/10,000}$. For these results, TRIALMDP used parameter settings $\lambda_F=4.0$ and $\lambda_K=0.01$.

$p_A=p_B$	N	Size (CMH test) 1:1	RAR	BRAR	MDP	Effect Bias RAR	BRAR	MDP	N_A-N_B (5%, 95%) RAR	BRAR	MDP	$K(h)$ MDP	Utility Z-score RAR	BRAR	MDP
0.1	100	0.05	0.05	0.05	0.05	0.00	0.00	0.00	0.25 (-22, 24)	0.14 (-20, 22)	0.13 (-18, 18)	2.77	-542.40	-5.26	-9.15
0.3	100	0.05	0.05	0.05	0.05	0.00	0.00	0.00	-0.18 (-22, 20)	-0.07 (-20, 20)	-0.07 (-32, 32)	3.77	-505.00	-5.14	-13.45
0.5	100	0.05	0.05	0.05	0.06	0.00	0.00	0.00	0.05 (-18, 18)	0.12 (-18, 18)	0.06 (-46, 46)	3.88	-490.50	-4.96	-13.36
0.6	100	0.05	0.05	0.04	0.05	0.00	0.00	0.00	0.03 (-18, 18)	0.07 (-18, 18)	-0.04 (-42, 42)	3.85	-491.29	-4.98	-13.31
0.7	100	0.05	0.05	0.05	0.06	0.00	0.00	0.00	0.11 (-18, 18)	-0.01 (-16, 16)	0.32 (-36, 36)	3.68	-499.43	-5.12	-12.64
0.9	100	0.05	0.05	0.05	0.05	0.00	0.00	0.00	-0.10 (-16, 16)	-0.01 (-16, 16)	-0.02 (-18, 18)	2.77	-511.70	-5.05	-9.03

Table 2. Simulation study null scenarios. We report trial size rather than power; all other columns have the same meaning as in Table 1. For these results, TRIALMDP used $\lambda_F=4.0$ and $\lambda_K=0.01$.

p_A	p_B	N	Power/Size (CMH test) 1:1	RAR	BRAR	MDP	Effect Bias RAR	BRAR	MDP	N_A-N_B (5%, 95%) RAR	BRAR	MDP	$K(h)$ MDP	Utility Z-score RAR	BRAR	MDP
0.4	0.4	20	0.06	0.05	0.05	0.05	0.00	0.00	0.00	-0.04 (-8, 8)	-0.03 (-8, 8)	0.01 (-10, 10)	2.57	-45.59	-2.48	-3.57
0.8	0.4	20	0.61	0.60	0.54	0.56	0.00	0.00	0.02	2.13 (-6, 10)	1.47 (-6, 8)	5.09 (-6, 10)	2.28	-9.94	0.24	2.50

Table 3. Results of trial redesign. We report the same quantities as in the simulation study. For these results, TRIALMDP used $\lambda_F=3.0$ and $\lambda_K=0.05$.
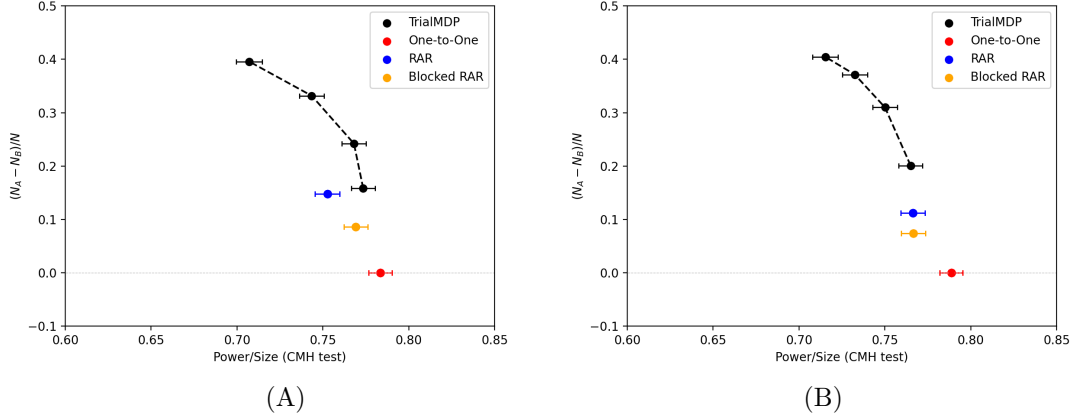
Fig. 5. Frontier plots. (A) Trial design characteristics for scenario $p_A{=}0.4, p_B{=}0.1$. (B) Trial design characteristics for scenario $p_A{=}0.7, p_B{=}0.4$ In each plot the curve traced by TRIALMDP corresponds to $\lambda_F{\in}\{2,3,4,5\}$. TRIALMDP used $\lambda_K{=}0.01$ in both plots. The points represent *mean* power and patient allocations; the error bars show symmetric 90% confidence intervals for the means. Note that there are error bars for the vertical direction, but they are too compact to be seen.

Beyond a one-dimensional comparison of utility, it is useful to compare the designs in two dimensions: statistical power and patient outcomes. As we vary the parameter λ_F, TRIALMDP designs trials that balance these quantities differently. We visualize this with *frontier plots*; trial designs are shown as points in two dimensions, with statistical power on the horizontal axis and allocation to A on the vertical axis. Higher and to the right is better. Figure 5 gives examples. In some scenarios, TRIALMDP's designs *dominate* the other adaptive designs, attaining higher power *and* assigning more patients to treatment A. Figure 5(A) is one such case. In other scenarios, TRIALMDP's designs do not dominate the others. However, Figure 5(B) shows that even in those cases, TRIALMDP still provides a useful way to control the balance between power and patient outcomes. For example, the user is free to choose a trial design with much better patient outcomes at the cost of slightly lower power, by selecting larger values of λ_F.

3.2   *Trial redesign*

We demonstrate TRIALMDP's practical usage by applying it to a historical trial. We chose the phase-II thymoglobulin trial described by Bashir *and others* (2012) because it *(i)* had two arms, *(ii)* had a small sample size (N=20), and *(iii)* the trial designers saw fit to use an adaptive design, for ethical reasons. This combination made the trial well-suited for testing our algorithm.

We redesigned the trial in two phases: "parameter tuning" and "testing." In the parameter tuning phase we swept through the same grid of $\lambda_F, \lambda_K, p_A, p_B$ values used in our simulation study, but with the sample size fixed at N=20. We ran our algorithm and simulated 10,000 trials at each grid point, and generated frontier plots similar to those in Figure 5. Visual inspection suggested that TRIALMDP with λ_F=3.0 and λ_K=0.05 would yield reasonable power and patient outcomes for a variety of p_A, p_B scenarios.

In the testing phase we simulated the thymoglobulin trial by computing point estimates of p_A=0.8 and p_B=0.4 from the original trial's results. We simulated two scenarios: a null scenario where p_A=p_B=0.4, and an alternative where p_A=0.8 and p_B=0.4. Using the design from TRIALMDP with "tuned" parameter values λ_F=3.0 and λ_K=0.05, we simulated 10,000 trials for each scenario. The results are aggregated in Table 3. Under the alternative scenario we found that TRIALMDP's design, on average, assigned significantly more patients to treatment A with a slightly decreased power of 0.557. Note also that in the null scenario, TRIALMDP's design had a somewhat inflated type-I error of 0.055.

4. DISCUSSION

Key takeaways.   We presented TRIALMDP, an algorithm for designing blocked RAR trials. TRIALMDP represents a blocked RAR trial as a Markov Decision Process, and solves for the optimal design via dynamic programming. The resulting design dictates the size and treatment allocation of the next block, given the results observed thus far.

Our algorithm allows users to choose the relative importance of *(i)* statistical power and *(ii)* patient outcomes. The trial designs generated by TRIALMDP consistently attain superior utility against a suite of baselines when *(i)* the effect size is large and *(ii)* patient outcomes are given sufficient importance. The simulation study in Section 3.1 demonstrates this.

TRIALMDP has some shortcomings worth keeping in mind. It is currently restricted to a narrow class of trials: two-armed trials with binary outcomes. All outcomes for past blocks must be observed before the next block can begin. The MDP formulation assumes a single statistical test (one-sided CMH) is performed at the end of the trial. While interim analyses may be used in trials governed by the current version of TRIALMDP, we provide no guarantees of optimality in that case. TRIALMDP's computational cost grows quickly with the number of patients, and becomes impractical for $N > 200$. Setting large values for the *minimum block size* and *block increment* parameters ($T_{\min}$ and κ) can ameliorate some of this expense. Simulations showed that in some scenarios, TRIALMDP's designs have modestly inflated type-I error, and may yield a slightly biased estimate of effect size. These weaknesses should be weighed against the vastly superior patient outcomes TRIALMDP can deliver.

Practical recommendations. The user of TRIALMDP immediately faces a question: *what values of λ_F and λ_K should be used?* Consider the terms of Equation 2.1. Since $V(h)$ is only a proxy for the statistical power, there isn't a clear way to assign practical meaning to λ_F, λ_K. For example, we cannot interpret λ_F as a literal "conversion rate" between units of failure and units of statistical power. This makes it difficult to set λ_F, λ_K in a principled way. Instead we recommend *tuning* λ_F and λ_K through a process like the one demonstrated in Section 3.2: *(i)* use the algorithm to design trials for a grid of λ_F, λ_K values; *(ii)* simulate trials for each design, for a set of scenarios p_A, p_B; *(iii)* examine the simulation results and choose λ_F, λ_K that yield acceptable power and patient outcomes across scenarios. As a starting point, $\lambda_F = 3.0$, $\lambda_K = 0.01$ yielded reasonable characteristics across all the scenarios in this paper.

Future improvements. Although TRIALMDP's current implementation is single-threaded, it is highly parallelizable and would have a speedup roughly linear in the number of threads. A multi-threaded parallel implementation is a natural next step.

There are multiple ways that TRIALMDP could be extended to a broader class of trials. For instance, it could permit more than two arms and more than two outcomes. This would incur exponentially greater computational expense, but may be useful for some very small trials.

The current MDP formulation assumes that the trial terminates after all patients have been treated. A more sophisticated MDP could incorporate interim analyses, accounting for the possibility of early termination for success or futility.

5. SOFTWARE

We implemented TRIALMDP in C++ and provide it as an R package on GitHub: `https://github.com/dpmerrell/TrialMDP`. We also provide the code for our Section 3 evaluations in another repository: `https://github.com/dpmerrell/TrialMDP-analyses`. This includes a Snakemake workflow (Mölder *and others*, 2021) that reproduces all results in this paper.

6. SUPPLEMENTARY MATERIAL

Our Supplementary Material contains four appendices. Appendix A gives our justification for using the function $V(h)$. Appendix B shows that our optimization problem has the optimal substructure property (and hence TRIALMDP yields an optimal policy with respect to our MDP assumptions). Appendix C derives the computational complexities given in Section 2.2. Appendix D shows that λ_F must be sufficiently large for an adaptive trial to attain higher utility than a single-block trial.

REFERENCES

BASHIR, QAISER, MUNSELL, MARK F., GIRALT, SERGIO, SILVA, LEANDRO DE PADUA, SHARMA, MANISH, COURIEL, DANIEL, CHIATTONE, ALEXANDRE, POPAT, UDAY, QAZIL-BASH, MUZAFFAR H., FERNANDEZ-VINA, MARCELO, CHAMPLIN, RICHARD E. *and others*. (2012, May). Randomized phase II trial comparing two dose levels of thymoglobulin in patients undergoing unrelated donor hematopoietic cell transplant. *Leukemia & Lymphoma* **53**(5), 915–919. Publisher: Taylor & Francis _eprint: https://doi.org/10.3109/10428194.2011.634039.

CHANDERENG, THEVAA AND CHAPPELL, RICK. (2019, September). Robust Blocked Response-Adaptive Randomization Designs. *arXiv:1904.07758 [stat]*. arXiv: 1904.07758.

CHANDERENG, THEVAA AND CHAPPELL, RICK. (2020). How to do response-adaptive randomization (rar) if you really must. *Clinical Infectious Diseases*.

COCHRAN, WILLIAM G. (1954). Some Methods for Strengthening the Common chi-squared Tests. *Biometrics* **10**(4), 417–451. Publisher: [Wiley, International Biometric Society].

HARDWICK, JANIS AND STOUT, QUENTIN F. (2002, May). Optimal few-stage designs. *Journal of Statistical Planning and Inference* **104**(1), 121–145.

HARDWICK, JANIS P. AND STOUT, QUENTIN F. (1995). Exact Computational Analyses for Adaptive Designs. *Lecture Notes-Monograph Series* **25**, 223–237. Publisher: Institute of Mathematical Statistics.

HARDWICK, JANIS P. AND STOUT, QUENTIN F. (1999, January). Using Path Induction to Evaluate Sequential Allocation Procedures. *SIAM Journal on Scientific Computing* **21**(1), 67–87.

KARRISON, THEODORE G, HUO, DEZHENG AND CHAPPELL, RICK. (2003). A group sequential, response-adaptive design for randomized clinical trials. *Controlled Clinical Trials* **24**(5), 506–522.

LATTIMORE, TOR AND SZEPESVARI, CSABA. (2020, July). *Bandit Algorithms*. Cambridge University Press.

LIU, SUYU AND LEE, J JACK. (2015). An overview of the design and conduct of the battle trials. *Chinese clinical oncology* **4**(3), 33–33.

MÖLDER, FELIX, JABLONSKI, KIM PHILIPP, LETCHER, BRICE, HALL, MICHAEL B., TOMKINS-TINCH, CHRISTOPHER H., SOCHAT, VANESSA, FORSTER, JAN, LEE, SOOHYUN, TWARDZIOK, SVEN O., KANITZ, ALEXANDER, WILM, ANDREAS, HOLTGREWE, MANUEL, RAHMANN, SVEN, NAHNSEN, SVEN *and others*. (2021, January). Sustainable data analysis with Snakemake. *F1000Research* **10**, 33.

PROSCHAN, MICHAEL AND EVANS, SCOTT. (2020). Resist the temptation of response-adaptive randomization. *Clinical Infectious Diseases* **71**(11), 3002–3004.

ROSENBERGER, WILLIAM F., STALLARD, NIGEL, IVANOVA, ANASTASIA, HARPER, CHERICE N. AND RICKS, MICHELLE L. (2001). Optimal Adaptive Designs for Binary Response Trials. *Biometrics* **57**(3), 909–913. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.0006-341X.2001.00909.x.

THALL, PETER F AND WATHEN, J KYLE. (2007). Practical bayesian adaptive randomisation in clinical trials. *European Journal of Cancer* **43**(5), 859–866.

VILLAR, SOFÍA S., BOWDEN, JACK AND WASON, JAMES. (2015*a*, May). Multi-armed Bandit Models for the Optimal Design of Clinical Trials: Benefits and Challenges. *Statistical Science* **30**(2), 199–215.

VILLAR, SOFÍA S., BOWDEN, JACK AND WASON, JAMES. (2018). Response-adaptive designs for binary responses: How to offer patient benefit while being robust to time trends? *Pharmaceutical Statistics* **17**(2), 182–197.

VILLAR, SOFÍA S., WASON, JAMES AND BOWDEN, JACK. (2015*b*, December). Response-Adaptive Randomization for Multi-arm Clinical Trials Using the Forward Looking Gittins Index Rule. *Biometrics* **71**(4), 969–978.

WOODROOFE, MICHAEL AND HARDWICK, JANIS. (1990, September). Sequential Allocation for an Estimation Problem with Ethical Costs. *The Annals of Statistics* **18**(3), 1358–1377.

APPENDIX

A. DERIVATION OF V

Equation 2.1 uses the following function, $V(h)$, as a proxy for a trial's statistical power:

$$V(h) = \frac{1}{N} \sum_i \frac{w_i}{\frac{1}{2}(\underline{p}_{A,i} + \underline{p}_{B,i})\frac{1}{2}(\underline{q}_{A,i} + \underline{q}_{B,i})}$$

where

$$w_i = N_{A,i} \cdot N_{B,i}/(N_{A,i} + N_{B,i}).$$

This appendix provides some justification for $V(h)$.

We're interested in blocked RAR trials where the final analysis uses a Cochran-Mantel-Haenzsel (CMH) superiority test. Recall that the CMH statistic takes this form:

$$\text{CMH}(h) = \frac{\sum_i w_i d_i}{\sqrt{\sum_i w_i \hat{p}_i \hat{q}_i}}$$

where

$$d_i = p_{A,i} - p_{B,i} \qquad \hat{p}_i = \frac{N_{A,i} \cdot p_{A,i} + N_{B,i} \cdot p_{B,i}}{T_i} \qquad \hat{q}_i = 1 - \hat{p}_i$$

Under the null hypothesis, CMH $\sim \mathcal{N}(0,1)$ asymptotically. Intuitively, we maximize the power of the test by choosing $N_{A,i}, N_{B,i}$ such that when $p_A \neq p_B$, the distribution of CMH has large mean without inflated variance. Our goal is to find an objective function V that, when maximized, yields trial designs with those characteristics.

As a first candidate we may try maximizing the expected value of of CMH:

$$\mathbb{E}_h\left[\text{CMH}(h)\right] \simeq \frac{\sum_i w_i(p_A - p_B)}{\sqrt{\sum_i w_i \left(\frac{N_{A,i} \cdot p_A + N_{B,i} \cdot p_B}{T_i}\right)\left(\frac{N_{A,i} \cdot q_A + N_{B,i} \cdot q_B}{T_i}\right)}}$$

$$= (p_A - p_B)\frac{\sum_i w_i}{\sqrt{\sum_i w_i \left(\phi p_A + (1-\phi)p_B\right)\left(\phi q_A + (1-\phi)q_B\right)}}$$

where $\phi_i = N_{A,i}/T_i$ is the *fraction* of block i's patients allocated to A. The trial designer has no control over p_A or p_B. So if they wish to maximize this quantity then they may ignore the factor

$(p_A - p_B)$, yielding

$$\frac{\sum_i w_i}{\sqrt{\sum_i w_i \left(\phi p_A + (1 - \phi)p_B\right)\left(\phi q_A + (1 - \phi)q_B\right)}} \tag{A.1}$$

as a proxy objective for maximizing power. It's important to note, however, a subtle property of Expression A.1. The denominator is minimized when more patients are allocated to the treatment with more *extreme* success probability—i.e., success probability closer to 0 or 1. As a result, the maximizer of Expression A.1 exhibits a preference toward that treatment. This preference manifested itself in earlier versions of the algorithm, which would do well when $\frac{1}{2} < p_B < p_A$, but would do worse when $p_B < p_A < \frac{1}{2}$.

As a second candidate, we may try maximizing the the related quantity

$$\frac{\mathbb{E}\left[\sum_i w_i d_i\right]}{\sqrt{\mathrm{Var}\left[\sum_i w_i d_i\right]}} = (p_A - p_B)\frac{\sum_i w_i}{\sqrt{\sum_i w_i^2 \left(\frac{p_A q_A}{N_{A,i}} + \frac{p_B q_B}{N_{B,i}}\right)}} \tag{A.2}$$

$$= (p_A - p_B)\frac{\sum_i w_i}{\sqrt{\sum_i w_i \left((1-\phi_i)p_A q_A + \phi_i p_B q_B\right)}} \tag{A.3}$$

$$\propto \frac{\sum_i w_i}{\sqrt{\sum_i w_i \left((1-\phi_i)p_A q_A + \phi_i p_B q_B\right)}} \tag{A.4}$$

Cochran uses Expression A.2 as a proxy for the power of a CMH test in his original justifications for the CMH statistic (Cochran, 1954). Like Expression A.1, the new Expression A.4 also exhibits a preference based on *extremality* of the success probabilities. However, it instead favors the treatment with *less* extreme success probability, i.e., probability nearer $\frac{1}{2}$. Versions of the algorithm based on Expression A.4 would manifest this preference during simulations. The algorithm would attain superior utility when $p_B < p_A < \frac{1}{2}$, but would do worse when $\frac{1}{2} < p_B < p_A$.

Note the similarity between Expression A.4 and Expression A.1. They have identical numerators, and both denominators have the form $\sqrt{\sum_i w_i \tilde{pq}}$ where $\tilde{pq}$ is some "combined variance" computed from p_A, p_B. They differ precisely in how they compute $\tilde{pq}$. This in turn produces their different preferences (toward the treatment with less-extreme and more-extreme success probability, respectively). Neither of these preferences are favorable. We would like a proxy for power

that has simpler dependence on p_A and p_B, which are unknown. To that end we propose our final candidate:

$$\frac{\sum_i w_i}{\sqrt{\sum_i w_i \frac{1}{2}(p_A + p_B)\frac{1}{2}(q_A + q_B)}} = \frac{\sum_i w_i}{\sqrt{\frac{1}{2}(p_A + p_B)\frac{1}{2}(q_A + q_B)}\sqrt{\sum_i w_i}}$$

$$= \sqrt{\frac{\sum_i w_i}{\frac{1}{2}(p_A + p_B)\frac{1}{2}(q_A + q_B)}}$$

or, after squaring,

$$\frac{\sum_i w_i}{\frac{1}{2}(p_A + p_B)\frac{1}{2}(q_A + q_B)}$$

This new quantity lets $\tilde{p}q = \frac{1}{2}(p_A + p_B)\frac{1}{2}(q_A + q_B)$, which has neither of the preferences exhibited by Expressions A.1 or A.4. Of course in practice we don't know p_A or p_B, so we substitute their MAP estimates at each block:

$$V(h) = \sum_i \frac{w_i}{\frac{1}{2}(\hat{\underline{p}}_{A,i} + \hat{\underline{p}}_{B,i})\frac{1}{2}(\hat{\underline{q}}_{A,i} + \hat{\underline{q}}_{B,i})}, \tag{A.5}$$

which is the expression for V used in Section 2.1 (up to a factor of $\frac{1}{N}$).

B. Optimal Substructure

We show that our optimization problem—maximizing expected utility—possesses the *optimal substructure* property. In other words, we prove that the recurrence relation (Equation 2.3) correctly decomposes the problem into subproblems, and reuses their solutions to solve the original problem.

Suppose the algorithm is evaluating $U^*(s_i)$ for some state s_i, and that it's already evaluated $U^*(s_{i+1})$ for every possible successor state s_{i+1} of s_i. Let π^* denote the optimal policy, i.e., the one yielding U^*. Then optimal substructure follows from the linearity of our utility function.

Assuming s_{i+1} is not terminal:

$$U^*(s_i) = \mathbb{E}_{h_{i+1}|\pi^*,s_i} \left[\frac{1}{N} V(h_{i+1}) - \lambda_F \cdot F(h_{i+1}) - \lambda_K \cdot K(h_{i+1}) \right]$$

$$= \mathbb{E}_{(s_{i+1},h_{i+2})|\pi^*,s_i} \left[\frac{w(a^*)}{N\tilde{p}\tilde{q}(s_{i+1})} + \frac{1}{N} V(h_{i+2}) - \lambda_F \cdot F(h_{i+2}) - \lambda_K \cdot (1 + K(h_{i+2})) \right]$$

$$= \mathbb{E}_{s_{i+1}|\pi^*,s_i} \left[\mathbb{E}_{h_{i+2}|\pi^*,s_{i+1}} \left[\frac{w(a^*)}{N\tilde{p}\tilde{q}(s_{i+1})} + \frac{1}{N} V(h_{i+2}) - \lambda_F \cdot F(h_{i+2}) - \lambda_K \cdot (1 + K(h_{i+2})) \right] \right]$$

$$= \mathbb{E}_{s_{i+1}|\pi^*,s_i} \left[\frac{w(a^*)}{N\tilde{p}\tilde{q}(s_{i+1})} - \lambda_K + \mathbb{E}_{h_{i+2}|\pi^*,s_{i+1}} \left[\frac{1}{N} V(h_{i+2}) - \lambda_F \cdot F(h_{i+2}) - \lambda_K \cdot K(h_{i+2}) \right] \right]$$

$$= \mathbb{E}_{s_{i+1}|\pi^*,s_i} \left[R(a^*, s_{i+1}) + U^*(s_{i+1}) \right]$$

$$= \max_a \left\{ \mathbb{E}_{s_{i+1}|s_i} \left[R(a, s_{i+1}) + U^*(s_{i+1}) \right] \right\}$$

which agrees exactly with the recurrence in Equation 2.3. A similar computation covers the case when s_{i+1} is terminal.

Put another way, our dynamic program's recurrence relation computes $U^*(s)$ correctly at each state, and will yield the optimal policy π^*.

C. Computational complexity

We derive the space and time complexities given in Section 2.2.

Let S_i denote the set of all 2×2 contingency tables containing i observations. Define $[T_{\min} : \kappa : N - T_{\min}] = \{T_{\min}, T_{\min} + \kappa, \ldots, N - T_{\min}\}$, the set of integers ranging from $T_{\min}$ to $N - T_{\min}$ in increments of κ. Then $|S_i| = O(i^3)$, and the size of the full state space is

$$|S| = \sum_{i \in [T_{\min}:\kappa:N-T_{\min}]} |S_i| = \sum_{i=T_{\min}}^{N-T_{\min}} O\left(\frac{i^3}{\kappa}\right) = O\left(\frac{(N - T_{\min})^4}{\kappa}\right).$$

The algorithm stores data proportional to $|S|$, so this gives the space complexity.

The time complexity results from a nested sum over states, actions, and transitions:

$$
\begin{aligned}
T(N, |\Phi|) &= \sum_{i\in[T_{\min}:\kappa:N-T_{\min}]} \sum_{s\in S_i} \sum_{j\in[T_{\min}:\kappa:N-T_{\min}-i]} \sum_{\phi\in\Phi} \phi{\cdot}j(1-\phi){\cdot}j \\
&= \sum_{i\in[T_{\min}:\kappa:N-T_{\min}]} \sum_{s\in S_i} \sum_{j\in[T_{\min}:\kappa:N-T_{\min}-i]} j^2 \sum_{\phi\in\Phi} \phi(1-\phi) \\
&= \sum_{i\in[T_{\min}:\kappa:N-T_{\min}]} \sum_{s\in S_i} \sum_{j\in[T_{\min}:\kappa:N-T_{\min}-i]} j^2{\cdot}O(|\Phi|) \\
&= O(|\Phi|)\cdot \sum_{i\in[T_{\min}:\kappa:N-T_{\min}]} \sum_{s\in S_i} \sum_{j\in[T_{\min}:\kappa:N-T_{\min}-i]} j^2 \\
&= O(|\Phi|)\cdot \sum_{i\in[T_{\min}:\kappa:N-T_{\min}]} O(i^3) \sum_{j\in[T_{\min}:\kappa:N-T_{\min}-i]} j^2 \\
&= O(|\Phi|)\cdot \sum_{i\in[T_{\min}:\kappa:N-T_{\min}]} O(i^3)O\left(\frac{(N-T_{\min}-i)^3}{\kappa}\right) \\
&= O(|\Phi|)\cdot O\left(\frac{(N-T_{\min})^3}{\kappa}\right)\cdot \sum_{i\in[T_{\min}:\kappa:N-T_{\min}]} O(i^3) \\
&= O\left(|\Phi|\cdot \frac{(N-T_{\min})^7}{\kappa^2}\right).
\end{aligned}
$$

D. The utility of single-block vs. multi-block trials

It is not always possible for a trial with multiple blocks to attain higher utility (Equation 2.1) than a trial with one block. The cost λ_K of an additional block outweighs any improvements in patient outcomes, unless λ_F is large enough. In this appendix we find conditions on λ_F, λ_K that determine when a two-block adaptive trial can attain higher utility than a single-block (i.e., fixed randomization) trial.

Given the true values of p_A and p_B, we can compute the utility of a single-block trial in closed form:

$$
U_{\text{single}} = \frac{1}{(p_A + p_B)(q_A + q_B)} - \lambda_K
$$

We can likewise compute the utility of a two-block trial in closed form. Assume the first block of the trial treats T patients, assigning half to each treatment. In the second block, assume $\phi \cdot (N-T)$ patients are assigned to treatment A and $(1-\phi) \cdot (N-T)$ are assigned to treatment B. Then the two-block trial has this utility:

$$U_{\text{two-block}} = \frac{T + (N-T)\phi(1-\phi)}{N(p_A + p_B)(q_a + q_B)} + \frac{\lambda_F}{N}(2\phi - 1)(N-T)(p_A - p_B) - 2\lambda_K$$

We want to find conditions where $U_{\text{two-block}} - U_{\text{single}} \geqslant 0$. Some algebra yields this condition:

$$-\frac{1}{(p_A + p_B)(q_A + q_B)}\hat{\phi}^2 + \lambda_F(p_A - p_B)\hat{\phi} - \lambda_B \frac{N}{N - T} \geqslant 0$$

where $\hat{\phi} = (2\phi - 1)$ is a convenient shorthand. The LHS of this inequality is a concave quadratic in $\hat{\phi}$. It has real roots (and hence, a feasible region) iff

$$\lambda_F^2(p_A - p_B)^2(p_A + p_B)^2(q_A + q_B)^2 - 4\frac{\lambda_B(p_A + p_B)(q_A + q_B)N}{N - T} \geqslant 0.$$

Rearranging gives the following condition on λ_F

$$\lambda_F \geqslant \frac{2}{(p_A - p_B)}\sqrt{\frac{N\lambda_K}{(N - T)(p_A + p_B)(q_A + q_B)}}.$$

The key takeaway is that λ_F must be sufficiently large before *any* RAR design can possibly attain higher utility than a single-block, fixed-randomization trial.

This analysis does not account for the uncertainty in p_A and p_B. TRIALMDP operates under this uncertainty, and will therefore not generally choose a single-block design when it is truly optimal. In practice, the algorithm only chooses a single-block design when λ_F is small relative to λ_K.