Maximizing the Success Probability of Policy Allocations in Online Systems

Artem Betlei[1], **Mariia Vladimirova**[1], **Mehdi Sebbar**[2], **Nicolas Urien**[2], **Thibaud Rahier**[1], **Benjamin Heymann**[1]

[1] Criteo AI Lab, France
[2] Criteo Ad Landscape, France
{a.betlei, m.vladimirova, m.sebbar, n.urien, t.rahier, b.heymann}@criteo.com

Abstract

The effectiveness of advertising in e-commerce largely depends on the ability of merchants to bid on and win impressions for their targeted users. The bidding procedure is highly complex due to various factors such as market competition, user behavior, and the diverse objectives of advertisers. In this paper we consider the problem at the level of user timelines instead of individual bid requests, manipulating full policies (i.e. pre-defined bidding strategies) and not bid values. In order to optimally allocate policies to users, typical multiple treatments allocation methods solve knapsack-like problems which aim at maximizing an expected value under constraints. In the industrial contexts such as online advertising, we argue that optimizing for the probability of success is a more suited objective than expected value maximization, and we introduce the `SuccessProbaMax` algorithm that aims at finding the policy allocation which is the most likely to outperform a fixed reference policy. Finally, we conduct comprehensive experiments both on synthetic and real-world data to evaluate its performance. The results demonstrate that our proposed algorithm outperforms conventional expected-value maximization algorithms in terms of success rate.

Introduction

Optimizing marketing effectiveness relies on using individualized bidding policies, exploiting the fact that each user responds differently. A *policy* may include a set of rules or actions over an extended period of time, e.g., cash bonuses, promotion and display ad shown to consumers on online platforms. Without loss of generality, we take the narrow view of bidding for display advertising in order to ground our research into a real life application. In this context, the task at hand is to specify a full bidding strategy (the policy) on the future advertisement opportunities for each given users during a given time period.

In practice, it is typical to have a fixed budget allocated to a campaign. From an advertising perspective, a bidding strategy must maximize the total expected revenue while ensuring that the expected total cost does not exceed a specified budget.

Usually, this problem is modeled as a multiple choice knapsack problem (Demirović et al. 2019; Zhou et al. 2023) with the objective to select at most one item (bid policy) from each user such that the sum of the weights (expected cost) of selected items does not exceed the capacity (budget) while the total *reward* (expected revenue) is maximized. This problem is known to be NP-hard, although it can be tackled with mixed integer linear programming or through Lagrangian relaxation (Sinha and Zoltners 1979).

From a causal perspective, it is classical to consider every individual ad as a treatment, and the optimization problem goal is to maximize the total causal effect of these treatments by correctly assigning treatments to users. There exist various approaches for individual treatment assignment that differ by the objective function they optimize: learning models to predict either outcomes, causal effects or directly the optimal treatment assignment. Fernández-Loría et al. (2022) compare these approaches analytically and show that the assignment learners optimize the bias-variance tradeoff with respect to decision-making errors.

Optimization at the opportunity –or bid –level, which we refer as *bid by bid* optimization, requires to attribute each observed reward to the action that actually caused it, e.g. each conversion must be attributed to a shown ad. This attribution problem is very complex as there usually are several ads displayed in the few hours preceding each conversion (Bompaire, Gilotte, and Heymann 2021; Bompaire, Désir, and Heymann 2021; Ji and Wang 2017; Dalessandro et al. 2012). It causes fundamental problems in the estimation of the causal effects and makes the bid by bid optimization extremely difficult in practice.

Furthermore, display advertising campaigns, like many other online systems, are operated under several business and technical constraints. In particular, it is typical for an advertising campaign to have a budget constraint. Several algorithms allow adapting bid by bid optimization techniques to such constraints (Castiglioni et al. 2022; Conitzer et al. 2022). While these algorithms have their merits and are largely deployed in practice, they are, however, poorly suited for *causal* bid by bid methods. This is because (a) typical causal methods inherently suppose the absence of causal interaction between the treatment units — such assumption is in general violated when mixing causal method for bid by bid optimization and budget pacing; (b) the overall methodology needs to trade off marginal value and marginal future total cost (Bompaire, Gilotte, and Heymann 2021), which is arguably intractable at the bid level.

Our first idea is to reformulate the problem at the user timeline level (i.e. considering all the bid requests and subsequent events relative to a user along a given time period) which implies to consider entire policies instead of individual bids. With this new formulation, the optimal policy allocation search is framed as a multiple treatment allocation problem, and the causal effects (cost and value) of policies are much easier to estimate than that of individual bids. Our approach is not to be understood as in competition with usual bid by bid design approaches (Moriwaki et al. 2021) but rather complementary. Indeed, any bid by bid design approach could be included as one of the candidate policies we wish to choose from when allocating policies to users with our methodology. If a bid by bid design policy happens to be globally optimal, our method will simply conclude that the optimal policy allocation consists in assigning this policy to every user.

However, we claim that searching for the policy allocation function which maximizes an expected value under an expected cost constraint (which is typically done in treatment allocation problems) is not always the best objective. In a large organization, it is often necessary to have guidelines that allow for consistent decision-making regarding product design and improvements. Without such guidelines, individuals cannot handle trade-offs between different quantities (for example, quality and volume) consistently across the whole organization. One may think about designing medication (which should be efficient but also avoid negative side-effects) or electrical batteries (which should have a big enough capacity while not relying too much on rare materials). This leads to the definition of a *success* across organizations, e.g. in online advertising, it corresponds to increasing generated value *without increasing the cost* with respect to a reference outcome. Taking this as a premise, the (constrained) maximization of a single quantity –such as revenue –is not anymore the right criterion as it does not account for the uncertainty underlying the phenomenon at play, nor does it account for what will be considered a success.

This motivates the focus on finding the policy allocation resulting in the *highest probability of success*. While every metric has its pros and cons, we believe a focus on success probability, with a very flexible notion of *success*, is of particular operational interest, see Fig. 1 for an illustrative example.

In summary, this work presents the following contributions:

- We formally propose the idea of framing the optimization problem at the policy level instead of focusing on bid by bid design, and mathematically formalize both the expected value maximization and the success probability maximization problems.
- We develop a novel customized solution to address the specificities of the success probability optimization problem.
- Finally, we present a series of numerical experiments which were conducted on both synthetic and real-world data, showing that our approach outperforms traditional value maximization methods in terms of success rate guarantees.

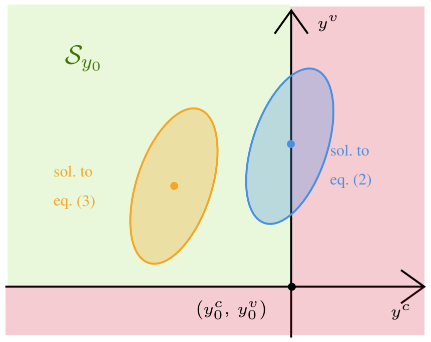

Figure 1: Distributions of the (cost,value) outcome vector $\mathbf{Y} = (Y^c, Y^v)$. In blue for the solution of (2) (maximization of $\mathbb{E}[Y^v]$ under the condition $\mathbb{E}[Y^c] \le y_0^c$) ; and in orange the solution of (3) (maximization of the success probability).

Problem Formulation

Preliminary considerations Throughout this section, we will implicitly refer to a given time period τ of length Δt, i.e. $\tau = [t_0, t_0 + \Delta t]$. We consider a given advertiser Adv who has a fixed budget C to spend over period τ.

Set of candidate policies We assume given a set of K *candidate policies* $\Pi = \{\pi_0, \pi_1, \ldots, \pi_{K-1}\}$ each encapsulating *bidding strategies* that may be applied by Adv to each user **consistently throughout the period** τ. The *reference policy* π_0 is the default bidding strategy used by Adv (typically corresponding to the strategy which is already rolled out in production for this advertiser). This set of policies Π can be thought of as a collection of potential treatments in a *multiple treatment allocation* problem. Note that we do not consider treatments at the level of bidding opportunities here, but at the level of an extended time period, during which we apply policies –or bidding strategies –which each have an integrated way to decide on how to bid on each user for all the opportunities that will arise during period τ.

Random variables and potential outcomes Considering the above setup, and with respect to any given user u targetable by Adv, we define the following random variables:

- $\mathbf{X} \in \mathcal{X} \subset \mathbb{R}^d$ contains a snapshot of the *features* of u captured at time t_0,
- $\mathbf{Y} = (Y^v, Y^c) \in \mathcal{Y} \subset \mathbb{R}_+^2$ contains respectively the *value* generated by u in favor of Adv during period τ and the *cost* Adv spent to advertize to u.

For any $\pi \in \Pi$, we denote $\mathbf{Y}(\pi) = (Y^v(\pi), Y^c(\pi))$ the *potential outcomes* (Rubin 1974) we would have observed **had** π **been applied to** u during τ. In what follows, we consider the tuple $(\mathbf{X}, \mathbf{Y}(\pi_0), \ldots, \mathbf{Y}(\pi_{K-1}))$ has an underlying probability distribution $\mathbb{P}$, which we will overload by simplicity to designate its marginals and conditionals. All expectancy notations $\mathbb{E}$ will refer implicitly to $\mathbb{P}$.

Factuals and counterfactuals Assuming that we apply policy $\pi_u \in \Pi$ to a given user u during τ, we denote $\mathbf{y}_u = (y_u^v, y_u^c)$ the corresponding realization of the outcome variable $\mathbf{Y}$, and $\{\mathbf{y}_u(\pi)\}_{\pi\in\Pi}$ the corresponding realizations of the potential outcomes variables $\{\mathbf{Y}(\pi)\}_{\pi\in\Pi}$. In that case, $\mathbf{y}_u = \mathbf{y}_u(\pi_u)$ is called the observed *factual outcome* and the $\{\mathbf{y}_u(\pi_u)\}_{\pi\in\Pi\setminus\{\pi_u\}}$ are the un-observed *counterfactual outcomes*.

Population random variables Let $\mathcal{U} = \{1, \ldots, N\}$ be the set of users who are targetable by Adv during period τ. We have access to a randomized controlled trial (RCT) –in this case also called an online controlled experiment or A/B test –on population $\mathcal{U}$ during this period, **randomly** assigning to each of those N users the K potential policies in Π.

Formally, let $\{K_u\}_{u\in\mathcal{U}}$ be N i.i.d. uniform categorical variables with values in $\{0, \ldots, K-1\}$. Each $u \in \mathcal{U}$ is assigned to the policy π_{K_u} during τ, resulting in the definition of the collection $\{(\mathbf{X}_u, \mathbf{Y}_u)\}_{u\in\mathcal{U}}$, where the $(\mathbf{X}_u, \mathbf{Y}_u) = (\mathbf{X}_u, \mathbf{Y}_u(\pi_{K_u}))$. We will denote $\mathbb{P}$[1] the probability distribution of $\{(\mathbf{X}_u, \mathbf{Y}_u)\}_{u\in\mathcal{U}}$. Lastly we assume that, in expectation, Adv exactly spends their advertising cost budget C during τ had they assigned the default policy π_0 to every user, i.e. $\mathbb{E}\left[\sum_{u\in\mathcal{U}} Y_u^c(\pi_0)\right] = C$.

Expected Value Maximization Problem

At the user level In the setup we introduced, one aim may be to find an *optimal policy allocation*, i.e. a mapping from $\mathcal{X}$ to policies from Π so that *expected total value* generated in favor of Adv is maximized, while respecting (in expectation) the total budget constraint.

Formally, we are looking for a solution $\phi^* : \mathcal{X} \to \Pi$ to the problem:

$$\max_{\phi\in\Pi^{\mathcal{X}}} \mathbb{E}\left[\sum_{u\in\mathcal{U}} Y_u^v(\phi(\mathbf{X}_u))\right] \text{ s.t. } \mathbb{E}\left[\sum_{u\in\mathcal{U}} Y_u^c(\phi(\mathbf{X}_u))\right] \leq C. \tag{1}$$

In practice, $\Pi^{\mathcal{X}}$ is very large and hard to explore efficiently, making (1) a difficult problem, especially since it involves estimation of K potential outcomes in parallel. A crucial observation is that the problem may be simplified by reducing it to a partition of the space $\mathcal{X}$, which leads to a reparametrization of (1), as explained in the next subsection.

Assuming a given partitioning of the user space We consider given a partition function $\gamma : \mathcal{X} \to \mathcal{G}$ where $\mathcal{G} = \{1, \ldots, M\}$ contains the indexes of the partition components (or buckets). Reasoning at a bucket-level instead of the user-level is practical in causal estimation setups since it enables to circumvent the fundamental problem of causal inference (Betlei et al. 2021) and is more compliant with privacy restrictions (Kleber 2019). A reasonable partitioning can be chosen by the domain knowledge or recursively with causal trees through heterogeneous treatment effect estimation (Athey and Imbens 2016; Wager and Athey 2018; Tu et al. 2021; Ai et al. 2022).

Given the partition function γ, we propose to simplify problem (1): instead of searching through all ϕs in $\Pi^{\mathcal{X}}$, we restrict our search to the allocations of the form $\psi \circ \gamma$ where $\psi \in \Pi^{\mathcal{G}}$. In short, we look for allocation functions that assign all users belonging to the same bucket $g \in \mathcal{G}$ to the same policy $\pi \in \Pi$.

[1]We drop the reference to $\mathcal{U}$ in $\mathbb{P}$ for simplicity.

Formally, this leads to the reparametrized problem, where we are looking for a solution $\psi^* : \mathcal{G} \to \Pi$ to the problem:

$$\max_{\psi\in\Pi^{\mathcal{G}}} \mathbb{E}\left[\sum_{u\in\mathcal{U}} Y_u^v(\psi(G_u))\right] \text{ s.t. } \mathbb{E}\left[\sum_{u\in\mathcal{U}} Y_u^c(\psi(G_u))\right] \leq C. \tag{2}$$

where $G_u = \gamma(\mathbf{X}_u)$ for all $u \in \mathcal{U}$.

Solving the expected value maximization problem The value expectation maximization problem formalized in (2) may be solved using mixed integer linear programming or Lagrangian relaxation approaches, which make the problem tractable in practice despite being NP-Hard (Sinha and Zoltners 1979). Nevertheless, the knapsack formulation remains a proxy to the marketing problem and its solution does not always align with the business goal.

Remark 1 –mix of A and B rollout If number of buckets $M = 1$, we assign **all users** to the same policy. This corresponds to a typical rollout decision in an online advertising company: we are A/B testing multiples policies, then depending on the results choosing which one should be rolled out. Our setup allows for a rollout of a **mix** of tested policies, given by function ψ.

Remark 2 –relaxing the allocation space We can relax problems (1) (2) by allowing for *soft allocations*, i.e. mappings from $\mathcal{X}$ (resp. $\mathcal{G}$) to $\Delta = \Delta(\Pi)$ where Δ denotes all categorical distributions with values in Π:

$$\Delta := \left\{ (p(k))_{k\in[\![1,K-1]\!]} \in [0,1]^K \text{ s.t. } \sum_k p(k) = 1 \right\}.$$

For $\psi \in \Delta^{\mathcal{G}}$ and $g \in \mathcal{G}$ and for convenience of notations, we will refer to the kth component of $\psi(g)$ –i.e. the probability for ψ to assign a user in bucket g to policy π_k –as $\psi(g, k)$.

Success Probability Maximization Problem

In this section, we will focus on the case where we are given a partitioning of the user space and consider more general soft allocation setup presented in *Remark 2* at the end of the previous section.

Instead of searching for the allocation that maximizes the expected value under constraint as in (1) and (2), one can also be interested in maximizing their *success probability*, especially in cases where the variance of the variables at play is high. For instance, a policy ψ^* that satisfies (2) might deliver very bad values occasionally. As explained in the introduction, the risk-aversion of industrial players often motivates them to prefer reliable small-increments to uncertain substantial ones.

Instead, we suppose there is an agreement beforehand on the definition of the *success* of a given policy allocation function $\psi : \mathcal{G} \to \Delta$ through the characterization of a convex region $\mathcal{S} \subset \mathcal{Y}$ such that "ψ is successful on a the set of users $\mathcal{U}$" is equivalent to $\sum_{u\in\mathcal{U}} \mathbf{Y}_u(\psi) \in \mathcal{S}$, where for any $\psi \in \Delta^{\mathcal{G}}$ we denote by simplicity $\mathbf{Y}(\psi) := \mathbf{Y}(\psi(\gamma(\mathbf{X}))$.

The success probability maximizing policy ψ^* is therefore a solution to

$$\max_{\psi\in\Delta^{\mathcal{G}}} \mathbb{P}\Big(\sum_{u\in\mathcal{U}} \mathbf{Y}_u(\psi) \in \mathcal{S}\Big) = \max_{\psi\in\Delta^{\mathcal{G}}} \mathbb{E}\left[\mathbb{I}_{\mathcal{S}}\Big(\sum_{u\in\mathcal{U}} \mathbf{Y}_u(\psi)\Big)\right], \quad (3)$$

where $\mathbb{I}_{\mathcal{S}}$ is the indicator function of the success set $\mathcal{S}$.

Example Our problem is defined with respect to any convex success region $\mathcal{S} \subset \mathcal{Y}$. In practice, we will consider success regions relative to a fixed $\mathbf{y}_0 = (y_0^v, y_0^c)$, of the form

$$\mathcal{S}_{\mathbf{y}_0} = \{(y^v, y^c) \in \mathcal{Y} \text{ s.t. } y^v > y_0^v \text{ and } y^c \le y_0^c\}, \quad (4)$$

where $\mathbf{y}_0$ should be interpreted as a *reference outcome*, for example the outcome we observe if we assign the reference policy to every user $\sum_{u\in\mathcal{U}} \mathbf{Y}_u(\pi_0)$. The success region $\mathcal{S}_{\mathbf{y}_0}$ corresponds to all outcomes with an *increased value and decreased cost* with respect to the reference value and cost $\mathbf{y}_0$.

In Figure 1, $\mathcal{S}_{\mathbf{y}_0}$ is displayed in green and its complementary $\bar{\mathcal{S}}_{\mathbf{y}_0}$ in red. We represent the distributions of the outcome $\mathbf{Y}$ for the respective allocations output by (i) the possible solution to (3) (maximization of the success probability) in orange and (ii) the possible solution to (2) (maximization of $\mathbb{E}[Y^v]$ under the condition $\mathbb{E}[Y^c] \le y_0^c$) in blue. The orange outcome has a very high probability to be in $\mathcal{S}_{\mathbf{y}_0}$, even if it generates a bit less value on average than the blue one, which presents a high risk of being outside of the success region (for example by breaking the cost constraint).

The `SuccessProbaMax` Algorithm

In this section, we present solutions for the problems (1) and (2). We will focus on the bucket-level versions of these problems, and therefore assume given a fixed partitioning $\gamma : \mathcal{X} \to \mathcal{G} = \{1, \dots, M\}$ of the feature space. This function could have been given by an expert or learned by machine learning algorithm, but it is not the focus of this work.

Gaussian Parametrization of the Problem

In this subsection, we introduce a novel method to solve the success probability maximization problem. This optimization problem, stated in (3), presents several non-trivial difficulties: (a) the indicator function which expectancy we are maximizing is not continuous on $\Delta^{\mathcal{G}}$ and (b) the criteria we wish to maximize is non-concave. We use

$$\mathbf{Y}_{g,k} = \sum_{u\in\mathcal{U}} \mathbf{Y}_u(\pi_k)\mathbb{I}\left(\gamma(\mathbf{X}_u) = g\right),$$

as a compact notation for the total expected outcome from users in bucket g, had they been allocated to policy π_k. Assuming that the buckets in $\mathcal{G}$ are approximately balanced in size (each containing $\approx N/M$ data points), we observe around $N/(MK)$ i.i.d. realizations to estimate each $\mathbf{Y}_{g,k}$. Let $\boldsymbol{\mu}_{k,g}$ and $\boldsymbol{\Sigma}_{k,g}$ be the mean and covariance matrix of the potential outcomes which contain value and cost.

For any soft allocation $\psi : \mathcal{G} \to \Delta$ –which maps all buckets in $\mathcal{G}$ to a stochastic mix of policies in Π–the total expected outcome under allocation ψ

$$\mathbf{Y}(\psi) = \sum_k \sum_g \psi(g,k)\mathbf{Y}_{g,k}, \quad \sum_k \psi(g,k) = 1.$$

The distributions $\psi(g,k)\mathbf{Y}_{g,k}$ are independent, therefore, we can use the Lyapunov central limit theorem and approximate the total expected outcome by a Gaussian distribution

$$\mathbf{Y}(\psi) \sim \mathcal{N}\left(\boldsymbol{\mu}(\psi), \boldsymbol{\Sigma}(\psi)\right),$$

where $\boldsymbol{\mu}(\psi) = \sum_k \sum_g \psi(g,k)\boldsymbol{\mu}_{g,k}$ and $\boldsymbol{\Sigma}(\psi) = \mathrm{Var}\left[\sum_k \sum_g \psi(g,k)\mathbf{Y}_{g,k}\right]$. Depending on assumptions, $\boldsymbol{\Sigma}(\psi)$ can be a linear or quadratic function of $\psi(g,k)$ due to different sources of randomness which lead to different variances. We assume that $\boldsymbol{\Sigma}(\psi) = \sum_k \sum_g \psi(g,k)\boldsymbol{\Sigma}_{g,k}$ (more details in Supplementary).

Parameters Estimation

When we do not have a direct access to parameters $\boldsymbol{\mu}_{g,k}$ and $\boldsymbol{\Sigma}_{g,k}$, we need to estimate them. In practice, the parameters are estimated on a **randomized control trial (RCT) dataset** $\mathcal{D} = \{(\mathbf{x}_u, \mathbf{y}_u)\}_{u\in\mathcal{U}}$ –realization of the collection $\{(\mathbf{X}_u, \mathbf{Y}_u)\}_{u\in\mathcal{U}}$ introduced in the last section. More precisely, $(\mathbf{x}_u, \mathbf{y}_u) = (\mathbf{x}_u, \mathbf{y}_u(\pi_{k_u}))$ are i.i.d. realizations of $(\mathbf{X}, \mathbf{Y}(\pi_{k_u}))$, where $\{k_u\}_{u\in\mathcal{U}}$ are i.i.d. realization of a uniform categorical variable on $\{0, \dots, K-1\}$. For $k \in [\![0, K-1]\!]$ and $g \in \mathcal{G}$, we will refer to the restrictions of $\mathcal{D}$ to points $u \in \mathcal{U}$ for which $\gamma(\mathbf{x}_u) = g$ and $k_u = k$ as $\mathcal{D}_{g,k}$.

To estimate parameters, we choose **mean and variance estimation methods** (e.g. bootstrapping Efron (1979)) which take as input a dataset $\mathcal{D}_{g,k}$ containing realizations of $\mathbf{Y}$ for a given bucket g and policy k and return respectively its mean $\{\hat{\boldsymbol{\mu}}_{g,k}\}$ and variances $\{\hat{\boldsymbol{\Sigma}}_{g,k}\}$.

Gradient Computation

In the following, for $\psi \in \Delta^{\mathcal{G}}$, we will denote for clarity purposes $\mathcal{C}(\psi) = \mathbb{P}\left(\sum_{u\in\mathcal{U}} \mathbf{Y}_u(\psi) \in \mathcal{S}\right) = \mathbb{E}\left[\mathbb{I}_{\mathcal{S}}\left(\sum_{u\in\mathcal{U}} \mathbf{Y}_u(\psi)\right)\right]$ the criterion we wish to optimize. The indicator function is discontinuous on the border of $\mathcal{S}$. It prevents us from directly using a stochastic gradient method (Shapiro, Dentcheva, and Ruszczynski 2021). The next lemma [2] provides an explicit expression for the gradient of the criteria.

Lemma 1. *The gradient of $\mathcal{C}$ at ψ satisfies*

$$[\nabla\mathcal{C}(\psi)]_{g,k} = \mathbb{E}\Big[\mathbb{I}_{\mathcal{S}}(\mathbf{Y})\Big((\mathbf{Y}-\boldsymbol{\mu}(\psi))^{\mathsf{T}}\boldsymbol{\Sigma}(\psi)^{-1}\cdot\boldsymbol{\mu}_{g,k} - \frac{1}{2}\big(\boldsymbol{\Sigma}(\psi) - (\mathbf{Y}-\boldsymbol{\mu}(\psi))(\mathbf{Y}-\boldsymbol{\mu}(\psi))^{\mathsf{T}}\big)\cdot\boldsymbol{\Sigma}(\psi)^{-1}\boldsymbol{\Sigma}_{g,k}\boldsymbol{\Sigma}(\psi)^{-1}\Big)\Big].$$

[2]The proof of Lemma 1 uses classical arguments from the policy learning literature (Williams 1992; Sutton and Barto 2018) and further relies on the chain rule with a few relations for multivariate Gaussian variables. We defer the proof to the Supplementary.

Optimization

Here, we present our optimization algorithm `SuccessProbaMax` (Algorithm 1) to solve (3) which takes as input

(a) **success region** $\mathcal{S} \subset \mathcal{Y}$ to define a criteria $\mathcal{C}$ introduced in (3). We typically consider success regions relative to a reference outcome (y_0^v, y_0^c): it might be defined as all outcomes corresponding to increased value and decreased cost with respect to the reference value and cost;

(b) **estimated mean and variance values** $\{\hat{\boldsymbol{\mu}}_{g,k}\}$ and $\{\hat{\boldsymbol{\Sigma}}_{g,k}\}$ for all pairwise couples of groups g and candidate policies k. There is a particular case when the exact values of mean and variances are known and do not require estimation;

(c) **some hyperparameters** such as an initial policy allocation function $\psi_0 \in \Delta^{\mathcal{G}}$, number of steps n_{st} and learning rate η.

```
Input: $\mathcal{S}$, $\{\hat{\boldsymbol{\mu}}_{g,k}\}$, $\{\hat{\boldsymbol{\Sigma}}_{g,k}\}$, $\psi_0$, $n_{\text{st}} > 0$, $\eta > 0$
$\psi \leftarrow \psi_0$
for $t = 0$ to $n_{st}$ do
    $\hat{\boldsymbol{\mu}} \leftarrow \sum_{k,g} \psi(g,k)\hat{\boldsymbol{\mu}}_{g,k}$, $\hat{\boldsymbol{\Sigma}} \leftarrow \sum_{k,g} \psi(g,k)\hat{\boldsymbol{\Sigma}}_{g,k}$
    $\nabla \leftarrow \hat{\nabla}\mathcal{C}(\psi)$
    $\psi \leftarrow \psi + \eta\nabla$
    Project $\psi$ onto $\Delta^M$
end
Return $\psi$
```

Algorithm 1: `SuccessProbaMax`

The algorithm performs a gradient ascent $\nabla \leftarrow \hat{\nabla}\mathcal{C}(\psi)$ which can be computed using the formula from Lemma 1 and a numerical integration method for computing the expectation $\mathbb{E}_\psi$, e.g. a Monte-Carlo approach. The updated gradient is, then, projected onto the space of metapolicies $\Delta^{\mathcal{G}}$ to produce a solution candidate for (3) using a method from (Duchi et al. 2008). We provide several possible improvements of Algorithm 1 in Supplementary.

Remark As the computation of the gradient through the closed-form expression requires a matrix inversion, it is not always the best option computationally.

This is the case for the success region (4), for which we observe that the criterion rewrites

$$\mathbb{E}\left[\mathbb{I}_{\mathcal{S}}\left(\sum_{u\in\mathcal{U}} \mathbf{Y}_u(\psi)\right)\right] = \mathrm{cdf}_{Y^c}(y_0^c) - \mathrm{cdf}_{\mathbf{Y}}(\mathbf{y}_0),$$

where $\mathrm{cdf}_{Y_c}(y_0^c)$ is a (univariate) c.d.f. of Y_c in y_0^c and $\mathrm{cdf}_{\mathbf{Y}}(\mathbf{y}_0)$ is a (bivariate) c.d.f. of $\mathbf{Y}$ in $\mathbf{y}_0$. (see appendix for the definition of bivariate c.d.f.). To speed up the algorithm, we rely on an approximation of the bivariate c.d.f. based on the error function (Tsay, Ke et al. 2011) to estimate $\mathrm{cdf}_{\mathbf{Y}}(\mathbf{y}_0)$, and then implement it in JAX – this way we can directly use the automatic differentiation in JAX to numerically approximate the gradient of $\mathrm{cdf}_{Y^c}(y_0^c) - \mathrm{cdf}_{\mathbf{Y}}(\mathbf{y}_0)$.

Experimental Results

For all experiments below we use JAX framework (Bradbury et al. 2018) for the numerical estimation of the criterion's gradient, utilizing automatic differentiation within JAX instead of explicitly calculating the gradient and integrating it over an outcome. Hyperparameters used for the methods are provided in Supplementary material and source code[3] is published to reproduce all the empirical results.

Datasets

Besides the synthetic setups, which will be described below, we test algorithm on two large-scale, real world datasets.

- **CRITEO-UPLIFT v2** (Diemert et al. 2021) is provided by the AdTech company Criteo. Data contains 13.9 million samples which are collected from several incremental A/B tests. It includes 12 features, 1 binary treatment and 2 binary outcome labels (”visit” and ”conversion”). Following (Zhou et al. 2023), we use ”visit” label as proxy of the cost and ”conversion” as the value. For the buckets, we used quantile bins of the ”f0” feature. Finally, we randomly partitioned dataset into two equal parts for train and test. Preprocessing details are in Supplementary.
- **Private dataset** is constructed from a large-scale real-time bidding RCT. One feature was chosen based on an expert knowledge, buckets were created then as quantile-based projections of the feature. Dataset is aggregated over 70 days and consists of 9 buckets, 3 bidding policies (with reference) and 100 bootstraps of values and associated costs for each pair (bucket, policy).

  Remaining details along with aggregated datasets for one- and two-dimensional outcome cases are available in Supplementary material.

One-dimensional Outcome

Here we assume an outcome $\mathcal{Y} \in \mathbb{R}$. Problem is parameterized by a difficulty level r so that $\mathcal{S} = \{(r, +\infty)\}$. We present here results for synthetic data. Private data results are in Supplementary.

Baselines `SuccessProbaMax` is compared to several baselines searching for the optimal policy allocation:

- `Bruteforce`$(\{\boldsymbol{\mu}_{g,k}\}, \{\boldsymbol{\Sigma}_{g,k}\}, \mathcal{S})$ method that compares all possible *hard* allocations and for a given difficulty level returns allocation that maximises criterion;
- `Greedy1D`$(\{\boldsymbol{\mu}_{g,k}\})$ algorithm that returns the policy with the maximum mean value per bucket.

Synthetic data generation We generate Gaussian distributions for two cases: (i) ”large variance” and (ii) ”small variance”, the same setup but the relative difference between the variances is much smaller – we expect the latter problem be harder than the former for the algorithms that take into account the variance. See Table 1 for precise parameters of distributions (data construction details and illustration of policy distributions per bucket are provided in Supplementary).

[3]https://github.com/criteo-research/success-proba-max

Example	$[\mu_{g,k}]$	$[\Sigma_{g,k}]$
Large variance	$\begin{bmatrix} 2 & 1.9 & 0 \\ 2 & 1 & 0 \\ 2 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 9 & 1 & 9 \\ 9 & 1 & 9 \\ 1 & 1 & 1 \end{bmatrix}$
Small variance	$\begin{bmatrix} 2 & 1.9 & 0 \\ 2 & 1 & 0 \\ 2 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 9 & 1 & 9 \\ 9 & 1 & 9 \\ 1 & 1 & 1 \end{bmatrix} \cdot 0.01$

Table 1: Gaussian distribution parameters for synthetic data generation with three buckets ($M = 3$), three policies ($K = 3$) and one outcome ($Y \in \mathbb{R}$).

Results We firstly fix $\mu_{g,k}$, $\Sigma_{g,k}$ and use them directly in the algorithm to avoid a source of randomness arising from parameters estimation, we provide the results below. Then we generate normal distributions with parameters $\mu_{g,k}$, $\Sigma_{g,k}$ and use *estimations* $\hat{\mu}_{g,k}$, $\hat{\Sigma}_{g,k}$ in the algorithm – corresponding results are presented in Supplementary.

On Fig. 2 (left) we show how our method performs for easier problem with large variance with varying difficulty level r. SuccessProbaMax starts from uniform allocation ψ_0^{unif} and performs same as Bruteforce. The key reasons why our algorithm beats Greedy1D is that we i) directly optimize metric of interest and that we ii) effectively incorporate variance into optimization, while Greedy1D only operates with means.

Fig. 2 (right) shows results for the small variance case. Firstly, note that the gain over Greedy1D (in the region $r \in [5, 5.5]$) is drastically smaller than in the previous case. Then, at difficulty level $r > 6$ performance of our algorithm drops down - as criterion value $\mathcal{C}(\psi_0^{unif})$ becomes 0 and the gradient is not updated. To overcome the problem, we can either "warm-start" from baseline policy (e.g. from Greedy1D one) or to explore, by estimating the criterion for several random initial allocations (ψ_0^{expl} on the figure).

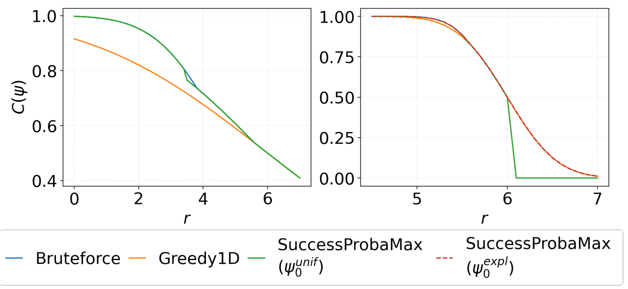

Figure 2: Results for different difficulty level r on synthetic setup with one-dimensional outcome for large (left) and small (right) variance cases.

Two-dimensional Outcome

In this case we consider an outcome $\mathcal{Y} \in \mathbb{R}^2$, so $\mathbf{Y} = (Y^v, Y^c)$. Problem is parameterized by two-dimensional difficulty level $\mathbf{r} = (r_v, r_c)$ so that $\mathcal{S} = \{(r_v, +\infty), (-\infty, r_c]\}$.

Baselines In addition to Bruteforce, SuccessProbaMax is compared with two other baselines that search for the optimal policy allocation:

- LinProg($\{\boldsymbol{\mu}_{g,k}\}, r_c$) algorithm (linear programming) that solves the fractional knapsack problem and returns a policy (soft allocation) with the maximum mean value per bucket;
- MixedInt($\{\boldsymbol{\mu}_{g,k}\}, r_c$) algorithm (mixed-integer linear programming) that solves the 0/1 knapsack problem and returns a policy (hard allocation) with the maximum mean value per bucket.

Synthetic data generation We generate bivariate Gaussian distributions for cases (i) and (ii), see Table 2 for precise parameters of distributions (an illustration of policy distributions is provided in Supplementary).

Example	$[\mu^v_{g,k}]$	$[\Sigma^v_{g,k}]$	$[\mu^c_{g,k}]$	$[\Sigma^c_{g,k}]$	ρ
μ_2^c and Σ_1^c larger	$[2, 1]$	$[9, 1]$	$[1, 1.5]$	$[4, 1]$	0.5
μ_2^c and Σ_1^c smaller	$[2, 1]$	$[9, 1]$	$[1, 0.5]$	$[1, 1]$	0.5

Table 2: Bivariate Gaussian distribution parameters for synthetic data generation with one bucket ($M = 1$), two policies ($K = 2$) and two-dimensional outcome ($\mathbf{Y} = (Y^v, Y^c) \in \mathbb{R}^2$).

We firstly fix $\mu^v_{g,k}, \Sigma^v_{g,k}$ and $\mu^c_{g,k}, \Sigma^c_{g,k}$, and use them directly in the algorithm to avoid a source of randomness arising from parameters estimation (results for the case with parameters estimation are presented in Supplementary).

Two-dimensional outcome: results. For the experiment, we fix $r_v = 0$ and vary r_c only. Fig. 3 shows that for both cases, SuccessProbaMax started from uniform allocation ψ_0^{unif} reaches the same performance as Bruteforce.

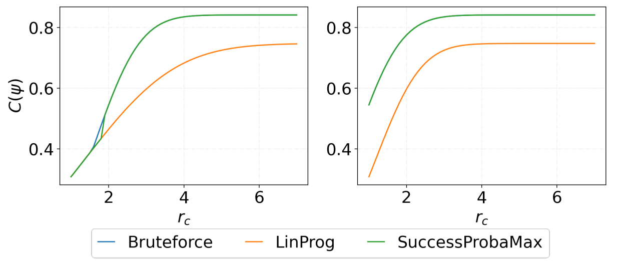

Figure 3: Result for different r_c with fixed $r_v = 0$ on synthetic setup with two-dimensional outcome for cases (i) (left) and (ii) (right).

Private dataset In our first experiment, we fix $r_c = 0$ and vary r_v only - so we check if we can increase total value while having same total cost as for reference. We then repeat computations, but now we fix $r_v = 0$ and vary r_c only - in this case we wonder how often we can reach at least total value of the reference policy while changing total cost (this case is described in Supplementary).

Results Fig. 4 describes results on the private dataset with two-dimensional outcome for the range of Gain r_v while $r_c = 0$ for train (left) and test (right) splits. Our algorithm, initialized with ψ_0 from exploration, reach the Gain of 0.01

in value (1% over the reference) with probability 0.7 for train and 0.4 for test, while for `MixedInt` respective probabilities are 0.35 and 0.1. Note that `Bruteforce` might not be the best here as some soft allocation may outperform hard ones for particular (r_v, r_c).

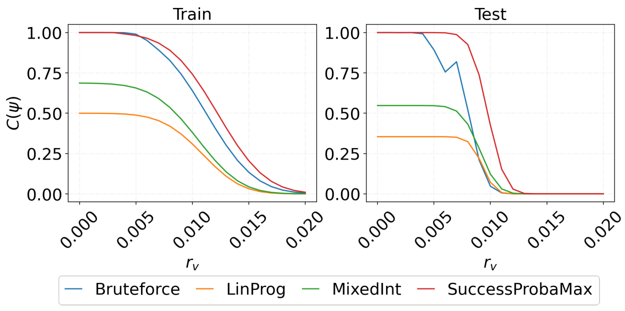

Figure 4: Results for different Gain r_v while $r_c = 0$ on private dataset with two-dimensional outcome for train (left) and test (right) splits.

CRITEO-UPLIFT v2 Data contains 2 policies including reference (”control”), so value can be increased only by increasing the cost. Thus, now we vary both r_v and r_c from 0 to 0.2, and a trade-off between value and cost is expected.

Results Fig. 5 depicts differences in $\mathcal{C}(\psi)$ between our algorithm and best baseline `Bruteforce` (absolute values are provided in Supplementary). Firstly, there is indeed a trade-off - for increasing cost by $x\%$ value increases by roughly $2x\%$. In addition, our algorithm reach higher $\mathcal{C}(\psi)$ in several regions (e.g. where $r_c \in [0.03, 0.04]$ and $r_v \in [0.04, 0.08]$ or where $r_c \in [0.08, 0.1]$ and $r_v \in [0.1, 0.16]$).

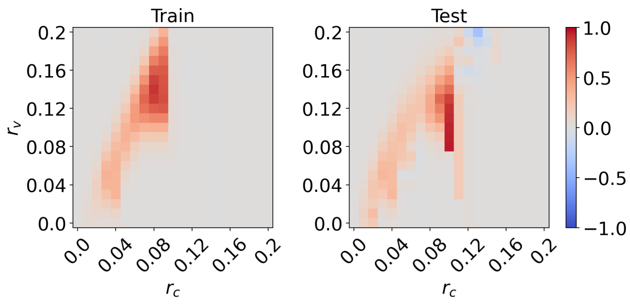

Figure 5: $C(SuccessProbaMax) - C(Bruteforce)$ results for different r_v and r_c on CRITEO-UPLIFT v2 with two-dimensional outcome for train (left) and test (right) splits.

The results correspond to the sketch provided in Fig. 1. It represents the distributions of the outcome $\mathbf{Y}$ for the respective allocations output by (i) `SuccessProbaMax` in orange and (ii) `Greedy` in blue. `SuccessProbaMax` outputs a solution for which the outcome has a very high probability to be in $\mathcal{S}_{\mathbf{y}_0}$ with $\mathbf{y}_0 = (y_0^v, y_0^c) = (r_v, r_c)$, even if generates a bit less value on average than `Greedy`, which presents a high risk of being outside of the success region (for example by breaking the cost constraint).

Related Work

Some recent papers address multiple treatment allocation problem under budget constraints from different perspectives. The standard two-stage method firstly estimates treatment effects to predict value and cost for each user, then solves a knapsack problem (Ai et al. 2022; Albert and Goldenberg 2022; Tu et al. 2021; Zhao et al. 2019). Nevertheless, the goals of two-stage approaches and real-world scenarios do not perfectly align. Yan et al. (2023) proposes a two-stage method with an addition regularizer to the knapsack problem loss to address a business goal. However, the regularizer requires a mathematically well-defined function (such as expected outcome metric) and its gradients estimation.

Applying the decision-focused framework for marketing problems under budget constraints, Du, Lee, and Ghaffarizadeh (2019) propose a rank method by comparing learned ratios between values and costs for the aggregated targeted treatment effect to improve user retention problem. However, Zhou et al. (2023) show that the suggested loss function cannot converge to a stable extreme point in theory and improve the framework. Authors limit the treatments to different levels of one treatment, e.g. different levels of discount of some products. Further, they develop an algorithm equivalent to the Lagrange dual method (’greedy’ approach) but based on learning to rank decision factors for multiple choice knapsack problem solutions. In our current context, our focus is solely on the top-ranked action, rather than the complete ranking itself. Moreover, as we discussed earlier, the knapsack formulation remains a proxy to our problem, so finding efficiently the best decision factors is still not equivalent to finding the best solution to the final business goal.

The closest to our work, Tu et al. (2021) suggest to reformulate the treatment allocation problem as a stochastic optimization task assuming normally distributed outcomes of bucket-level objective and constraints, however, the final problem remains in the knapsack form.

Conclusion and Future Works

We suggested a new formulation of the policy allocation problem that is better adapted to some downstream tasks when the success region is clearly identified. Compared to greedy approaches, our algorithm directly optimizes metric of interest and effectively utilizes variance in the optimization, while greedy ones only operate with means. Moreover, the proposed method can be efficiently applied to improve the given baseline policy.

Further works include a theoretical analysis of the algorithm. In particular, how it behaves numerically when the dimension of the outcome increases. Also, it is important to understand the relationship between the means and variances of potential outcomes that makes the proposed method outperform the greedy approaches. In addition to several suggested improvements of Algorithm 1, promising direction should be to couple the choice of user partitioning and the policy allocation problem into one master problem. Last, given that outcomes on different user segments may correlate, adapting the framework for Bayesian learning seems a pragmatic avenue for further research.

Acknowledgments

We would like to thank David Rohde and Eustache Diemert for their feedback and ideas during the project.

References

Ai, M.; Li, B.; Gong, H.; Yu, Q.; Xue, S.; Zhang, Y.; Zhang, Y.; and Jiang, P. 2022. LBCF: A Large-Scale Budget-Constrained Causal Forest Algorithm. In *ACM Web Conference*.

Albert, J.; and Goldenberg, D. 2022. E-Commerce Promotions Personalization via Online Multiple-Choice Knapsack with Uplift Modeling. In *ACM International Conference on Information & Knowledge Management*.

Athey, S.; and Imbens, G. 2016. Recursive partitioning for heterogeneous causal effects. *National Academy of Sciences*, 113(27): 7353–7360.

Betlei, A.; Gregoir, T.; Rahier, T.; Bissuel, A.; Diemert, E.; and Amini, M.-R. 2021. Differentially Private Individual Treatment Effect Estimation from Aggregated Data. *PPML Workshop*.

Bompaire, M.; Désir, A.; and Heymann, B. 2021. Robust label attribution for real-time bidding. *arXiv preprint arXiv:2012.01767*.

Bompaire, M.; Gilotte, A.; and Heymann, B. 2021. Causal Models for Real Time Bidding with Repeated User Interactions. In *ACM SIGKDD Conference on Knowledge Discovery & Data Mining*.

Bradbury, J.; Frostig, R.; Hawkins, P.; Johnson, M. J.; Leary, C.; Maclaurin, D.; Necula, G.; Paszke, A.; VanderPlas, J.; Wanderman-Milne, S.; and Zhang, Q. 2018. JAX: composable transformations of Python+NumPy programs.

Castiglioni, M.; Celli, A.; Marchesi, A.; Romano, G.; and Gatti, N. 2022. A Unifying Framework for Online Optimization with Long-Term Constraints. *arXiv preprint arXiv:2209.07454*.

Conitzer, V.; Kroer, C.; Sodomka, E.; and Stier-Moses, N. E. 2022. Multiplicative Pacing Equilibria in Auction Markets. *Operations Research*, 70(2): 963–989.

Dalessandro, B.; Perlich, C.; Stitelman, O.; and Provost, F. 2012. Causally Motivated Attribution for Online Advertising. In *International Workshop on Data Mining for Online Advertising and Internet Economy (AdKDD)*.

Demirović, E.; Stuckey, P. J.; Bailey, J.; Chan, J.; Leckie, C.; Ramamohanarao, K.; and Guns, T. 2019. An investigation into prediction+ optimisation for the knapsack problem. *Integration of Constraint Programming, Artificial Intelligence, and Operations Research*.

Diemert, E.; Betlei, A.; Renaudin, C.; Amini, M.-R.; Gregoir, T.; and Rahier, T. 2021. A large scale benchmark for individual treatment effect prediction and uplift modeling. *arXiv preprint arXiv:2111.10106*.

Du, S.; Lee, J.; and Ghaffarizadeh, F. 2019. Improve User Retention with Causal Learning. In *ACM SIGKDD Workshop on Causal Discovery*.

Duchi, J.; Shalev-Shwartz, S.; Singer, Y.; and Chandra, T. 2008. Efficient projections onto the l1-ball for learning in high dimensions. In *International Conference on Machine Learning*.

Efron, B. 1979. Bootstrap Methods: Another Look at the Jackknife. *The Annals of Statistics*, 7(1): 1 – 26.

Fernández-Loría, C.; Provost, F.; Anderton, J.; Carterette, B.; and Chandar, P. 2022. A comparison of methods for treatment assignment with an application to playlist generation. *Information Systems Research*.

Ji, W.; and Wang, X. 2017. Additional Multi-Touch Attribution for Online Advertising. *AAAI Conference on Artificial Intelligence*.

Kleber, M. 2019. Turtledove.

Moriwaki, D.; Hayakawa, Y.; Matsui, A.; Saito, Y.; Munemasa, I.; and Shibata, M. 2021. A Real-World Implementation of Unbiased Lift-based Bidding System. In *IEEE International Conference on Big Data*.

Rubin, D. B. 1974. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of educational Psychology*, 66(5): 688.

Shapiro, A.; Dentcheva, D.; and Ruszczynski, A. 2021. *Lectures on stochastic programming: modeling and theory*. Society for Industrial and Applied Mathematics.

Sinha, P.; and Zoltners, A. A. 1979. The Multiple-Choice Knapsack Problem. *Operations Research*, 27(3): 503–515.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement learning: An introduction*. MIT press.

Tsay, W.-J.; Ke, P.-H.; et al. 2011. A simple approximation for bivariate normal integral based on error function and its application on probit model with binary endogenous regressor. Technical report, Institute of Economics, Academia Sinica, Taipei, Taiwan.

Tu, Y.; Basu, K.; DiCiccio, C.; Bansal, R.; Nandy, P.; Jaikumar, P.; and Chatterjee, S. 2021. Personalized treatment selection using causal heterogeneity. In *ACM Web Conference*.

Wager, S.; and Athey, S. 2018. Estimation and inference of heterogeneous treatment effects using random forests. *Journal of the American Statistical Association*, 113(523): 1228–1242.

Williams, R. J. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Reinforcement learning*, 5–32.

Yan, Z.; Wang, S.; Zhou, G.; Lin, J.; and Jiang, P. 2023. An End-to-End Framework for Marketing Effectiveness Optimization under Budget Constraint. *arXiv preprint arXiv:2302.04477*.

Zhao, K.; Hua, J.; Yan, L.; Zhang, Q.; Xu, H.; and Yang, C. 2019. A unified framework for marketing budget allocation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1820–1830.

Zhou, H.; Li, S.; Jiang, G.; Zheng, J.; and Wang, D. 2023. Direct Heterogeneous Causal Learning for Resource Allocation Problems in Marketing. *AAAI Conference on Artificial Intelligence*.