Improved Regret Bounds for Tracking Experts with Memory

James Robinson **Mark Herbster**
Department of Computer Science
University College London
London
United Kingdom
{j.robinson | m.herbster}@cs.ucl.ac.uk

Abstract

We address the problem of sequential *prediction with expert advice* in a non-stationary environment with long-term memory guarantees in the sense of Bousquet and Warmuth [4]. We give a linear-time algorithm that improves on the best known regret bounds [27]. This algorithm incorporates a relative entropy projection step. This projection is advantageous over previous weight-sharing approaches in that weight updates may come with implicit costs as in for example portfolio optimization. We give an algorithm to compute this projection step in linear time, which may be of independent interest.

1 Introduction

We consider the classic problem of online prediction with expert advice [35] in a non-stationary environment. In this model `nature` sequentially generates outcomes which the `learner` attempts to predict. Before making each prediction, the `learner` listens to a set of n experts who each make their own predictions. The `learner` bases its prediction on the advice of the experts. After the prediction is made and the true outcome is revealed by `nature`, the accuracies of the `learner`'s prediction and the expert predictions are measured by a loss function. The `learner` receives information on all expert losses on each trial. We make no distributional assumptions about the outcomes generated, indeed `nature` may be assumed to be adversarial. The goal of the `learner` is to predict well relative to a predetermined comparison class of predictors, in this case the set of experts themselves. Unlike the standard regret model, where the `learner`'s performance is compared to the single best predictor in hindsight, our aim is for the `learner` to predict well relative to a sequence of comparison predictors. That is, "switches" occur in the data sequence and different experts are assumed to predict well at different times.

In this work our focus is on the case when this sequence consists of a few unique predictors relative to the number of switches. Thus most switches return to a previously "good" expert, and a `learner` that can exploit this fact by "remembering" the past can adapt more quickly than a `learner` who has no memory and must re-learn the experts after every switch. The problem of switching with memory in online learning is part of a much broader and fundamental problem in machine learning: how a system can adapt to new information yet retain knowledge of the past. This is an area of research in many fields, including for example, catastrophic forgetting in artificial neural networks [11, 36].

Contributions. In this paper we present an $\mathcal{O}(n)$-time per trial projection-based algorithm for which we prove the best known regret bound for tracking experts with memory. Our projection-based algorithm is intimately related to a more traditional "weight-sharing" algorithm, which we show is a new method for *Mixing Past Posteriors* (MPP) [4]. We show that surprisingly this method

35th Conference on Neural Information Processing Systems (NeurIPS 2021).

corresponds to the algorithm with the previous best known regret bound for this problem [27]. We also give an efficient $\mathcal{O}(n)$-time algorithm for computing exact relative entropy projection onto a simplex with non-uniform (lower) box constraints. Finally, we provide a guarantee which favors projection-based updates over weight-sharing updates when updating weights may incur costs.

The paper is organized as follows. We first introduce the model and discuss related work, giving a detailed overview of the previous results on which we improve. In Section 3 we give our main results, a regret bound which holds for two algorithms, and an algorithm to compute relative entropy projection with non-uniform lower box constraints in linear time. In Section 4 we derive a new "geometric-decay" method for MPP, and show the correspondence to the current best known algorithm [27]. We give a few concluding remarks in Section 5. All proofs are contained in the appendices.

1.1 Preliminaries

We first introduce notation. Let $\Delta_n := \{\boldsymbol{u} \in [0,1]^n : \|\boldsymbol{u}\|_1 = 1\}$ be the $(n-1)$-dimensional probability simplex. Let $\Delta_n^\alpha := \{\boldsymbol{u} \in [0,\alpha]^n : \|\boldsymbol{u}\|_1 = \alpha\}$ be a scaled simplex. Let $\mathbf{1}$ denote the vector $(1,\ldots,1)$ and $\mathbf{0}$ denote the vector $(0,\ldots,0)$. Let $\boldsymbol{e}_i$ denote the i^{th} standard basis vector. We define $D(\boldsymbol{u},\boldsymbol{w}) := \sum_{i=1}^n u_i \log \frac{u_i}{w_i}$ to be the relative entropy between $\boldsymbol{u}$ and $\boldsymbol{w}$. We denote component-wise multiplication as $\boldsymbol{u} \odot \boldsymbol{w} := (u_1 w_1, \ldots, u_n w_n)$. For $p \in [0,1]$ we define $\mathcal{H}(p) := -p \ln p - (1-p) \ln(1-p)$ to be the binary entropy of p, using the convention that $0 \ln 0 = 0$. We define ri S to be the relative interior of the set S. For any positive integer n we define $[n] := \{1,\ldots,n\}$. We overload notation such that $[\texttt{pred}]$ is equal to 1 if the predicate `pred` is true and 0 otherwise. For two vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ we say $\boldsymbol{\alpha} \preceq \boldsymbol{\beta}$ iff $\alpha_i \leq \beta_i$ for all $i = 1,\ldots,n$.

2 Background

In sequential prediction with expert advice `nature` generates elements from an outcome space, $\mathcal{Y}$ while the predictions of the `learner` and the experts are elements from a prediction space, $\mathcal{D}$ (e.g., we may have $\mathcal{Y} = \{0,1\}$ and $\mathcal{D} = [0,1]$). Given a non-negative loss function $\ell : \mathcal{D} \times \mathcal{Y} \to [0,\infty)$, learning proceeds in trials. On each trial $t = 1,\ldots,T$: 1) the `learner` receives the expert predictions $\boldsymbol{x}^t \in \mathcal{D}^n$, 2) the `learner` makes a prediction $\hat{y}^t \in \mathcal{D}$, 3) `nature` reveals the true label $y^t \in \mathcal{Y}$, and 4) the `learner` suffers loss $\ell^t := \ell(\hat{y}^t, y^t)$ and expert i suffers loss $\ell_i^t := \ell(x_i^t, y^t)$ for $i = 1,\ldots,n$. Common to the algorithms we consider in this paper is a weight vector, $\boldsymbol{w}^t \in \Delta_n$, where w_i^t can be interpreted as the algorithm's confidence in expert i on trial t. The `learner` uses a prediction function $\texttt{pred} : \Delta_n \times \mathcal{D}^n \to \mathcal{D}$ to generate its prediction $\hat{y}^t = \texttt{pred}(\boldsymbol{w}^t, \boldsymbol{x}^t)$ on trial t. A classic example is to predict with the weighted average of the expert predictions, that is, $\texttt{pred}(\boldsymbol{w}^t, \boldsymbol{x}^t) = \boldsymbol{w}^t \cdot \boldsymbol{x}^t$, although for some loss functions improved bounds are obtained with different prediction functions (see e.g., [47]). In this paper we assume (c,η)-realizability of ℓ and `pred` [4, 18, 45]. That is, there exists constants $c, \eta > 0$ such that for all $\boldsymbol{w} \in \Delta_n$, $\boldsymbol{x} \in \mathcal{D}^n$, and $y \in \mathcal{Y}$, $\ell(\texttt{pred}(\boldsymbol{w},\boldsymbol{x}), y) \leq -c \ln \sum_{i=1}^n w_i e^{-\eta \ell(x_i, y)}$. This includes η-exp-concave losses when $\texttt{pred}(\boldsymbol{w}^t, \boldsymbol{x}^t) = \boldsymbol{w}^t \cdot \boldsymbol{x}^t$ and $c = \frac{1}{\eta}$. For simplicity we present regret bound guarantees that assume $(c, \frac{1}{c})$-realizability, that is $c\eta = 1$. This includes the log loss with $c = 1$, and the square loss with $c = \frac{1}{2}$ when $\mathcal{D} = \mathcal{Y} = [0,1]$. The absolute loss is *not* (c,η)-realizable. Generalizing these bounds to the Hedge setting [13] is straightforward. For any comparison sequence of experts $i_{1:T} = i_1, \ldots, i_T \in [n]$ the regret of the `learner` with respect to this sequence is defined as

$$\mathcal{R}(i_{1:T}) := \sum_{t=1}^{T} \ell^t - \sum_{t=1}^{T} \ell_{i_t}^t .$$

We consider and derive algorithms which belong to the family of "exponential weights" (EW) algorithms (see e.g., [25, 35, 47]). After receiving the expert losses the EW algorithm applies the following incremental loss update to the expert weights,

$$\dot{w}_i^t = \frac{w_i^t e^{-\eta \ell_i^t}}{\sum_{j=1}^n w_j^t e^{-\eta \ell_j^t}} . \tag{1}$$

Static setting. In the static setting the `learner` competes against a single expert (i.e., $i_1 = \ldots = i_T$). For the static setting the EW algorithm sets $\boldsymbol{w}^{t+1} = \dot{\boldsymbol{w}}^t$ for the next trial, and for $(c, \frac{1}{c})$-realizable losses and prediction functions achieves a static regret bound of $\mathcal{R}(i_{1:T}) \leq c \ln n$.

Switching. In the switching (without memory) setting the `learner` competes against a sequence of experts $i_1, \ldots, i_T$ with $k := \sum_{t=1}^{T-1}[i_t \neq i_{t+1}]$ switches. The well-known Fixed-Share algorithm [23] solves the switching problem with the update

$$\boldsymbol{w}^{t+1} = (1-\alpha)\dot{\boldsymbol{w}}^t + \alpha\frac{\mathbf{1}}{n}\,, \tag{2}$$

by forcing each expert to "share" a fraction of its weight *uniformly* with all experts.[1] The update is parameterized by a "switching" parameter, $\alpha \in [0,1]$. With an optimally-tuned $\alpha = \frac{k}{T-1}$ the regret with respect to the best sequence of experts with k switches is

$$\mathcal{R}(i_{1:T}) \leq c\left((k+1)\ln n + (T-1)\mathcal{H}\left(\frac{k}{T-1}\right)\right) \leq c\left((k+1)\ln n + k\ln\frac{T-1}{k} + k\right). \tag{3}$$

Switching with memory. Freund [12] gave an open problem to improve on the regret bound (3) when the comparison sequence of experts is comprised of a small pool of size $m := |\cup_{t=1}^{T}\{i_t\}| \ll k$. Using counting arguments Freund gave an exponential-time algorithm with the information-theoretic ideal regret bound of $\mathcal{R}(i_{1:T}) \leq c\ln\left(\binom{n}{m}\binom{T-1}{k}m(m-1)^k\right)$, which is upper-bounded by

$$c\left(m\ln n + k\ln\frac{T-1}{k} + (k-m+1)\ln m + k + m\right). \tag{4}$$

The first efficient algorithm solving Freund's problem was presented in the seminal paper [4]. This work introduced the notion of a *mixing scheme*, which is a distribution $\boldsymbol{\gamma}^{t+1}$ with support $\{0,\ldots,t\}$. Given $\boldsymbol{\gamma}^{t+1}$, the algorithm's update on each trial is the *mixture* over all past weight vectors,

$$\boldsymbol{w}^{t+1} = \sum_{q=0}^{t}\gamma_q^{t+1}\dot{\boldsymbol{w}}^q\,, \tag{5}$$

where $\dot{\boldsymbol{w}}^0 := \frac{1}{n}\mathbf{1}$, and $\gamma_0^1 := 1$. Intuitively, by mixing all "past posteriors" (MPP) the weights of previously well-performing experts can be prevented from vanishing and recover quickly. An efficient mixing scheme requiring $\mathcal{O}(n)$-time per trial is the "*uniform*" mixing scheme given by $\gamma_t^{t+1} = 1-\alpha$ and $\gamma_q^{t+1} = \frac{\alpha}{t}$ for $0 \leq q < t$. A better regret bound was proved with a "*decaying*" mixing scheme, given by

$$\gamma_q^{t+1} = \begin{cases} 1-\alpha & q = t \\ \alpha\frac{1}{(t-q)^\rho}\frac{1}{Z_t} & 0 \leq q < t\,, \end{cases} \tag{6}$$

where $Z_t = \sum_{q=0}^{t-1}\frac{1}{(t-q)^\rho}$ is a normalizing factor, and $\rho \geq 0$. With a tuning of $\alpha = \frac{k}{T-1}$ and $\rho = 1$ this mixing scheme achieves a regret bound of[2]

$$\mathcal{R}(i_{1:T}) \leq c\left(m\ln n + 2k\ln\frac{T-1}{k} + k\ln(m-1) + k + k\ln\ln(eT)\right). \tag{7}$$

It appeared that to achieve the best regret bounds, the mixing scheme needed to decay towards the past. Unfortunately, computing (6) exactly requires the storage of all past weights, at a cost of $\mathcal{O}(nt)$-time and space per trial. Observe that these schemes set $\gamma_t^{t+1} = 1-\alpha$, where typically α is small, since intuitively switches are assumed to happen infrequently. All updates using such schemes are of the form

$$\boldsymbol{w}^{t+1} = (1-\alpha)\dot{\boldsymbol{w}}^t + \alpha\boldsymbol{v}^t\,, \tag{8}$$

which we will call the *generalized share update* (see [7]). Fixed-Share is a special case when $\boldsymbol{v}^t = \frac{\mathbf{1}}{n}$ for all t. This generalized share update features heavily in this paper.

For a decade it remained an open problem to give the MPP update a Bayesian interpretation. This was finally solved in [27] with the use of *partition specialists*. Here on each trial t, a specialist (first introduced in [14]) is either *awake* and predicts in accordance with a prescribed base expert,

[1] Technically in the original Fixed-Share update each expert shares weight to all *other* experts, i.e., $w_i^{t+1} = (1-\alpha)\dot{w}_i^t + \frac{\alpha}{n-1}\sum_{j\neq i}\dot{w}_j^t$. The two updates achieve essentially the same regret bound and are equivalent up to a scaling of α.

[2] (7) is a simplified upper bound of the bound given in [4, Corollary 9], using $\ln(1+x) \leq x$.

or is *asleep* and abstains from predicting. For n base experts and finite time horizon T there are $n2^T$ partition specialists. For Freund's problem an assembly of m partition specialists can predict exactly as the comparison sequence of experts. The Bayesian interpretation of the MPP update given in [27, Theorem 2] was simple: to define a mixing scheme $\boldsymbol{\gamma}^{t+1}$ was to induce a prior over this set of partition specialists. The authors of [27] proposed a simple Markov chain prior over the set of partition specialists, giving an efficient $\mathcal{O}(n)$-time per trial algorithm with the regret bound

$$\mathcal{R}(i_{1:T}) \leq c\left[m \ln\frac{n}{m} + m\mathcal{H}\left(\frac{1}{m}\right) + (T-1)\mathcal{H}\left(\frac{k}{T-1}\right) + (m-1)(T-1)\mathcal{H}\left(\frac{k}{(m-1)(T-1)}\right)\right] \quad (9)$$

$$\leq c\left(m \ln n + 2k \ln\frac{T-1}{k} + (k-m+1)\ln m + 2(k+1)\right) , \quad (10)$$

which is currently the best known regret bound for Freund's problem. In this work we improve on the bound (9) for tracking experts with memory (Theorem 2). We also show that in fact this Markov prior on partition specialists corresponds to a geometrically-decaying mixing scheme for MPP (Proposition 5). The regret bounds discussed in this paper all rely on optimally tuning one or more parameters, which in practice are usually unknown, and this is true for our regret bound.

Adaptive online learning algorithms with memory have been shown to have better empirical performance than those without memory [15], and to be effective in real-world applications such as intrusion detection systems [39]. While considerable research has been done on switching with memory in online learning (see e.g., [4, 7, 20, 21, 27, 49]), there remain several open problems. Firstly, there remains a gap between the best known regret bound for an efficient algorithm and the information-theoretic ideal bound (4). Present in both bounds (7) and (10) is the factor of 2 in the second term, which does not appear in (4). In [27] this was interpreted as the cost of co-ordination between specialists, essentially one "pays" twice per switch as one specialist falls asleep and another awakens. In this paper we make progress towards closing this gap by avoiding such additional costs the first time each expert is learned by the algorithm. That is, we pay to *remember* but not to *learn*.

Secondly, unless n is very large the current best known bound (9) beats Fixed-Share's bound (3) only when $m \ll k$, but suffers when m is even a moderate fraction of k. A natural question is can we improve on Fixed-Share when we relax the assumption that $m \ll k$, and only a few members of a sequence of experts need remembering (consider for instance, $m > k/2$)? In this paper we prove a regret bound that is not only tighter than (9) for all m, but for sufficiently large n improves on Fixed-Share for all $m \leq k$. See Figure 1 where we show this behavior for several existing regret bounds and our regret bound.

Our regret bound will hold for two algorithms; one utilizes a weight-sharing update in the sense of (8), and the other utilizes a projection update. Why should we consider projections? Consider for example a large model consisting of many weights, and to update these weights costs time and/or money. Alternatively consider the application of regret-bounded adaptive algorithms in online portfolio selection (see e.g., [32, 44]). Here each "expert" corresponds to a single stock and the weight vector $\boldsymbol{w}^t$ corresponds to a (normalized) portfolio. If ℓ_i^t is the negative log return of stock i after day t, then the loss function $\ell^t := -\ln\sum_{i=1}^n w_i^t e^{-\ell_i^t}$ is the negative log return of the portfolio. This loss is $(1,1)$-realizable by definition (although there is no prediction function [1]), and the daily price changes in the market naturally induce the "loss update" (1) by updating the portfolio weights. The algorithm's secondary update (projection or weight-sharing) requires the investor to then actively buy/sell to re-balance the portfolio after each trading period, but doing so may incur transaction costs proportional to the amount bought or sold (see e.g., [2, 32]). Online portfolio selection with transaction costs is an active area of research [9, 30, 32, 33]. In Section 3.3 we motivate the use of projections over weight-sharing in this context, proving that projections are strictly more "efficient".

2.1 Related work

Switching (without memory) in online learning was first introduced in [35] (see also the earlier [34] and independently in the context of universal coding in [43]), and extended with the Fixed-Share algorithm [23]. An extensive literature has built on these works, including but not limited to [1, 4, 7, 8, 16, 17, 21, 22, 24, 27, 29, 38, 42, 49]. Relevant to this work are the results for switching with memory [4, 7, 21, 27, 29, 49]. The first was the seminal work of [4]. The best known result is given in [27], which we improve on. In [49] a reduction of switching with memory to switching without

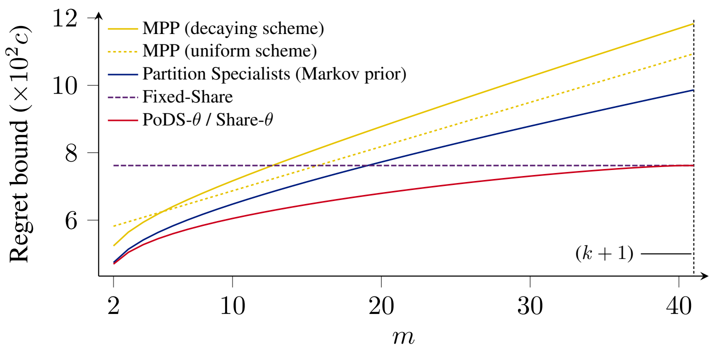

Figure 1: A comparison of the regret bounds discussed in this paper for $m \in [2, k+1]$ with $n = 500000$, $k = 40$, and $T = 4000$. Fixed-Share's bound is constant with respect to m. In this case previous "memory" bounds (blue & yellow) are much worse than Fixed-Share for larger values of m while our bound (red) improves on Fixed-Share for all $m \in [2, k]$.

memory is given, although with a slightly worse regret bound than [4]. Related to the experts model is the *bandits* setting, which was addressed in the memory setting in [49].

In [7] a unified analysis of both Fixed-Share and MPP was given in the context of online convex optimization. They observed the generalized share update (8) and slightly improved the bounds of [4]. Adaptive regret [1, 8, 19, 35] has been used to prove regret bounds for switching but unfortunately does not generalize to the memory setting. This paper primarily builds on the work of [4] with a new geometrically-decaying mixing scheme, and on [24] with a new relative entropy projection algorithm. Related to the problem of prediction with expert advice is that of universal coding in information theory (see e.g., [28, 37, 48] for a discussion). Similarly, related to the problem of tracking experts with memory is the problem of universal coding for switching sources with repeating statistics (see e.g., [40, 41, 43] and references therein).

3 Projection onto dynamic sets

In this section we give a relative entropy projection-based algorithm for tracking experts with memory. Given a non-empty set $\mathcal{C} \subseteq \Delta_n$ and a point $\boldsymbol{w} \in \text{ri } \Delta_n$ we define

$$\mathcal{P}(\boldsymbol{w}; \mathcal{C}) := \arg\min_{\boldsymbol{u} \in \mathcal{C}} D(\boldsymbol{u}, \boldsymbol{w})$$

to be the projection with respect to the relative entropy of $\boldsymbol{w}$ onto $\mathcal{C}$ [6]. Such projections were first introduced for switching (without memory) in online learning in [24], in which after every trial the weight vector $\dot{\boldsymbol{w}}^t$ is projected onto $\mathcal{C} = [\frac{\alpha}{n}, 1]^n \cap \Delta_n$, that is, the simplex with uniform box constraints. For prediction with expert advice this projection algorithm has the regret bound (3) (see [7]). Indeed, we will refer to $\boldsymbol{w}^{t+1} = \mathcal{P}(\dot{\boldsymbol{w}}^t; [\frac{\alpha}{n}, 1]^n \cap \Delta_n)$ as the "projection analogue" of (2). For tracking experts with memory our algorithm will instead project onto a set $\mathcal{C}$ such that each weight does not fall below a certain threshold that is learned for each expert.

Given $\boldsymbol{\beta} \in (0,1)^n$ such that $\|\boldsymbol{\beta}\|_1 \leq 1$, let

$$\mathcal{C}(\boldsymbol{\beta}) := \{\boldsymbol{x} \in \Delta_n : x_i \geq \beta_i, i = 1, \ldots, n\}$$

be a subset of the simplex which is convex and non-empty. Given $\boldsymbol{w} \in \text{ri } \Delta_n$, intuitively $\mathcal{P}(\boldsymbol{w}; \mathcal{C}(\boldsymbol{\beta}))$ is the projection of $\boldsymbol{w}$ onto the simplex with (non-uniform) lower box constraints $\boldsymbol{\beta}$. Relative entropy projection updates for tracking experts with memory were first suggested in [4, Section 5.2]. The authors observed that for any MPP mixing scheme $\boldsymbol{\gamma}^{t+1}$, the update (5) can be replaced with

$$\boldsymbol{w}^{t+1} = \mathcal{P}(\dot{\boldsymbol{w}}^t; \{\boldsymbol{w} \in \Delta_n : \boldsymbol{w} \succeq \gamma_q^{t+1} \dot{\boldsymbol{w}}^q, q = 0, \ldots, t\}), \tag{11}$$

and achieve the same regret bound. We build on this concept in this paper. Observe that for any choice of $\boldsymbol{\gamma}^{t+1}$ the set $\{\boldsymbol{w} \in \Delta_n : \boldsymbol{w} \succeq \gamma_q^{t+1} \dot{\boldsymbol{w}}^q, q = 0, \ldots, t\}$ corresponds to the set $\mathcal{C}(\boldsymbol{\beta})$ where

$$\beta_i = \max_{0 \leq q \leq t} \gamma_q^{t+1} \dot{w}_i^q \qquad i = 1, \ldots, n. \tag{12}$$

In this work we give an algorithm to compute $\mathcal{P}(\boldsymbol{w};\mathcal{C}(\boldsymbol{\beta}))$ exactly for any $\mathcal{C}(\boldsymbol{\beta})$ in $\mathcal{O}(n)$ time (Algorithm 3). With this algorithm and the mapping (12), one immediately obtains the projection analogue of MPP for any mixing scheme $\boldsymbol{\gamma}^{t+1}$ at essentially no additional computational cost. We point out however that for arbitrary mixing schemes computing $\boldsymbol{\beta}$ from (12) takes $\mathcal{O}(nt)$-time on trial t, improving only when some structure of the scheme can be exploited. We therefore propose the following method for tracking experts with memory *efficiently* using projection onto dynamic sets ("PoDS").

Just as (8) generalizes the Fixed-Share update (2), we propose PoDS as the analogous generalization of the update $\boldsymbol{w}^{t+1}=\mathcal{P}(\dot{\boldsymbol{w}}^t;\mathcal{C}(\alpha\frac{\mathbf{1}}{n}))$ (the projection analogue of Fixed-Share). PoDS maintains a vector $\boldsymbol{\beta}^t \in \Delta_n^\alpha$, and on each trial updates the weights by setting $\boldsymbol{w}^{t+1}=\mathcal{P}(\dot{\boldsymbol{w}}^t;\mathcal{C}(\boldsymbol{\beta}^t))$. Intuitively PoDS is the projection analogue of (8) with $\boldsymbol{\beta}^t$ corresponding simply to $\alpha\boldsymbol{v}^t$. In some cases $\boldsymbol{\beta}^t=\alpha\boldsymbol{v}^t$ for all t (e.g., for Fixed-Share), but in general equality may not hold since $\boldsymbol{\beta}^t$ and $\boldsymbol{v}^t$ can be functions of past weights, which may differ for weight-sharing and projection algorithms. Recall that (8) describes all MPP mixing schemes that set $\gamma_t^{t+1}=1-\alpha$. PoDS implicitly captures all such mixing schemes. This simple formulation of PoDS allows us to define new updates, which will correspond to new mixing schemes. In the following section we give a simple update for PoDS and prove the best known regret bound. In Section 3.2 we discuss Algorithm 3 and the efficient computation of $\mathcal{P}(\boldsymbol{w};\mathcal{C}(\boldsymbol{\beta}))$.

3.1 A simple update rule for PoDS

We now suggest a simple update rule for $\boldsymbol{\beta}^t$ in PoDS for tracking experts with memory. The regret bound for this algorithm is given in Theorem 2. We first set $\boldsymbol{\beta}^1 = \alpha\frac{\mathbf{1}}{n}$ to be uniform, and with a parameter $0 \leq \theta \leq 1$ update $\boldsymbol{\beta}^t$ on subsequent trials by setting

$$\boldsymbol{\beta}^{t+1} = (1-\theta)\boldsymbol{\beta}^t + \theta\alpha\dot{\boldsymbol{w}}^t \,. \tag{13}$$

We refer to PoDS with this update as PoDS-θ. Intuitively the constraint vector $\boldsymbol{\beta}^t$ is updated in (13) by mixing in a small amount of the current weight vector, $\dot{\boldsymbol{w}}^t$, scaled such that $\|\boldsymbol{\beta}^{t+1}\|_1 = \alpha$. If expert i predicted well in the past, then its constraint β_i^t will be relatively large, preventing the weight from vanishing even if that expert suffers large losses locally. Using Algorithm 3 in its projection step, PoDS-θ has $\mathcal{O}(n)$ per-trial time complexity.

As discussed, the vector $\boldsymbol{\beta}^t$ of PoDS is conceptually equivalent to the vector $\alpha\boldsymbol{v}^t$ of the generalized share update (8). If PoDS has a simple update rule such as (13) then it is straightforward to recover the weight-sharing equivalent by simply "pretending" equality holds on all trials. We now do this for PoDS-θ. Clearly we have $\boldsymbol{v}^1 = \frac{\mathbf{1}}{n}$, and if $\boldsymbol{\beta}^t = \alpha\boldsymbol{v}^t$ and $\boldsymbol{\beta}^{t+1} = \alpha\boldsymbol{v}^{t+1}$, then $\boldsymbol{v}^{t+1}=\frac{1}{\alpha}\boldsymbol{\beta}^{t+1}=\frac{1}{\alpha}(1-\theta)\boldsymbol{\beta}^t + \theta\dot{\boldsymbol{w}}^t=(1-\theta)\boldsymbol{v}^t + \theta\dot{\boldsymbol{w}}^t$. This then leads to an efficient sharing algorithm, which we call Share-θ. In Section 4 we show this algorithm is in fact a new MPP mixing scheme, which surprisingly corresponds to the previous best known algorithm for this problem. Both PoDS-θ and Share-θ use the same parameters (α and θ), differing only in the final update (see Algorithms 1&2).

Our regret bound for PoDS-θ and Share-θ is given in Theorem 2, the proof of which will use the following corollary.

Corollary 1. *Let $0 < \alpha < 1$. Then for any $\boldsymbol{u} \in \Delta_n$, $\boldsymbol{w} \in \operatorname{ri}\Delta_n$, and $\boldsymbol{\beta} \in \operatorname{ri}\Delta_n^\alpha$, let $\boldsymbol{p} = \mathcal{P}(\boldsymbol{w};\mathcal{C}(\boldsymbol{\beta}))$. Then,*

$$D(\boldsymbol{u},\boldsymbol{w}) - D(\boldsymbol{u},\boldsymbol{p}) \geq \ln(1-\alpha) \,. \tag{14}$$

We now give the regret bound which holds for both algorithms.

Theorem 2. *For any comparison sequence $i_1,\ldots,i_T$ containing k switches and consisting of m unique experts from a set of size n, if $\alpha = \frac{k}{T-1}$ and $\theta = \frac{k-m+1}{(m-1)(T-2)}$, the regret of both PoDS-θ and Share-θ with any prediction function and loss function which are $(c, \frac{1}{c})$-realizable is*

$$\mathcal{R}(i_{1:T}) \leq c\left(m\ln n + (T-1)\mathcal{H}\left(\frac{k}{T-1}\right) + (m-1)(T-2)\mathcal{H}\left(\frac{k-m+1}{(m-1)(T-2)}\right)\right). \tag{15}$$

The regret bound (15) is at least $c((m-1)\ln\frac{T-1}{k} - (k-m+1)\ln\frac{k}{k-m+1})$ tighter than the currently best known bound (9). Thus if $m \ll k$ then the improvement is $\approx cm\ln\frac{T}{k}$, and as $m \to k+1$ then

Algorithms 1&2 PoDS-θ / Share-θ

Input: $n > 0$, $\eta = \frac{1}{c} > 0$, $\alpha \in [0,1]$, $\theta \in [0,1]$

▷ `PoDS-`θ

init: $\boldsymbol{w}^1 \leftarrow \frac{\mathbf{1}}{n}$; $\boldsymbol{\beta}^1 \leftarrow \alpha\frac{\mathbf{1}}{n}$

▷ `Share-`θ

init: $\boldsymbol{w}^1 \leftarrow \frac{\mathbf{1}}{n}$; $\boldsymbol{v}^1 \leftarrow \frac{\mathbf{1}}{n}$

▷ `PoDS-`θ `& Share-`θ

for $t \leftarrow 1$ to T **do**
- **receive** $\boldsymbol{x}^t \in \mathcal{D}^n$
- **predict** $\hat{y}^t = \mathtt{pred}(\boldsymbol{w}^t, \boldsymbol{x}^t)$
- **receive** $y^t \in \mathcal{Y}$
- **for** $i \leftarrow 1$ to n **do**
  - $\dot{w}_i^t \leftarrow \frac{w_i^t e^{-\eta \ell_i^t}}{\sum_{j=1}^n w_j^t e^{-\eta \ell_j^t}}$

▷ `PoDS-`θ

$$\boldsymbol{w}^{t+1} \leftarrow \mathcal{P}(\dot{\boldsymbol{w}}^t; \mathcal{C}(\boldsymbol{\beta}^t)) \tag{16}$$
$$\boldsymbol{\beta}^{t+1} \leftarrow (1-\theta)\boldsymbol{\beta}^t + \theta\alpha\dot{\boldsymbol{w}}^t$$

▷ `Share-`θ

$$\boldsymbol{w}^{t+1} \leftarrow (1-\alpha)\dot{\boldsymbol{w}}^t + \alpha\boldsymbol{v}^t \tag{17}$$
$$\boldsymbol{v}^{t+1} \leftarrow (1-\theta)\boldsymbol{v}^t + \theta\dot{\boldsymbol{w}}^t \tag{18}$$

Algorithm 3 $\mathcal{P}(\boldsymbol{w}; \mathcal{C}(\boldsymbol{\beta}))$ in $\mathcal{O}(n)$ time

Input: $\boldsymbol{w} \in \text{ri } \Delta_n$; $\boldsymbol{\beta} \in (0,1)^n$ s.t. $\|\boldsymbol{\beta}\|_1 \leq 1$

Output: $\boldsymbol{w}' = \mathcal{P}(\boldsymbol{w}; \mathcal{C}(\boldsymbol{\beta}))$

init: $\mathcal{W} \leftarrow [n]$; $\boldsymbol{r} \leftarrow \boldsymbol{w} \odot \frac{1}{\boldsymbol{\beta}}$; $S_{\boldsymbol{w}} \leftarrow 0$; $S_{\boldsymbol{\beta}} \leftarrow 0$

while $\mathcal{W} \neq \emptyset$ **do**
- $\phi \leftarrow \mathtt{median}(\{r_i : i \in \mathcal{W}\})$
- $\mathcal{L} \leftarrow \{i \in \mathcal{W} : r_i < \phi\}$
- $L_{\boldsymbol{\beta}} \leftarrow \sum_{i \in \mathcal{L}} \beta_i$; $L_{\boldsymbol{w}} \leftarrow \sum_{i \in \mathcal{L}} w_i$
- $\mathcal{M} \leftarrow \{i \in \mathcal{W} : r_i = \phi\}$
- $M_{\boldsymbol{\beta}} \leftarrow \sum_{i \in \mathcal{M}} \beta_i$; $M_{\boldsymbol{w}} \leftarrow \sum_{i \in \mathcal{M}} w_i$
- $\mathcal{H} \leftarrow \{i \in \mathcal{W} : r_i > \phi\}$
- $\lambda \leftarrow \frac{1 - S_{\boldsymbol{\beta}} - L_{\boldsymbol{\beta}}}{1 - S_{\boldsymbol{w}} - L_{\boldsymbol{w}}}$
- **if** $\phi\lambda < 1$ **then**
  - $S_{\boldsymbol{w}} \leftarrow S_{\boldsymbol{w}} + L_{\boldsymbol{w}} + M_{\boldsymbol{w}}$
  - $S_{\boldsymbol{\beta}} \leftarrow S_{\boldsymbol{\beta}} + L_{\boldsymbol{\beta}} + M_{\boldsymbol{\beta}}$
  - **if** $\mathcal{H} = \emptyset$ **then**
    - $\phi \leftarrow \mathtt{min}(\{r_i : r_i > \phi, i \in [n]\})$
  - $\mathcal{W} \leftarrow \mathcal{H}$
- **else**
  - $\mathcal{W} \leftarrow \mathcal{L}$

$\lambda \leftarrow \frac{1 - S_{\boldsymbol{\beta}}}{1 - S_{\boldsymbol{w}}}$

$\forall i : 1, \ldots, n : w_i' \leftarrow \begin{cases} \beta_i & r_i < \phi \\ \lambda w_i & r_i \geq \phi \end{cases}$

the improvement is $\approx ck \ln \frac{T}{k}$. Additionally note that if $m = k+1$ (i.e., every switch we track a *new* expert) the optimal tuning of θ is zero, and PoDS-θ reduces to setting $\boldsymbol{\beta}^t = \alpha\frac{\mathbf{1}}{n}$ on every trial. That is, we recover the projection analogue of Fixed-Share. This is also reflected in the regret bound since (15) reduces to (3). Since $x\mathcal{H}(\frac{y}{x}) \leq y \ln(\frac{x}{y}) + y$, the regret bound (15) is upper-bounded by

$$\mathcal{R}(i_{1:T}) \leq c\left[m \ln n + k \ln \frac{T-1}{k} + (k-m+1) \ln \frac{T-2}{k-m+1} + (k-m+1) \ln(m-1) + 2k - m + 1\right].$$

Comparing this to (10), we see that instead of paying $c \ln \frac{T-1}{k}$ *twice* on every switch, we pay $c \ln \frac{T-1}{k}$ once per switch and $c \ln \frac{T-2}{k-m+1}$ for every switch we *remember* an old expert ($k - m + 1$ times). Unlike previous results for tracking experts with memory, PoDS-θ and its regret bound (15) smoothly interpolate between the two switching settings. That is, it is capable of exploiting memory when necessary and on the other hand does not suffer when memory is not necessary (see Figure 1).

3.2 Computing $\mathcal{P}(\boldsymbol{w}; \mathcal{C}(\boldsymbol{\beta}))$

Before we consider PoDS-θ and Share-θ further, we briefly discuss the computation of $\mathcal{P}(\boldsymbol{w}; \mathcal{C}(\boldsymbol{\beta}))$. In [24] the authors showed that computing relative entropy projection onto the simplex with *uniform* box constraints is non-trivial, but gave an algorithm to compute it in $\mathcal{O}(n)$ time. We give a generalization of their algorithm to compute $\mathcal{P}(\boldsymbol{w}; \mathcal{C}(\boldsymbol{\beta}))$ exactly for any non-empty set $\mathcal{C}(\boldsymbol{\beta})$ in $\mathcal{O}(n)$ time. As far as we are aware our method to compute exact relative entropy projection onto the simplex with non-uniform (lower) box constraints in linear time is the first, and may be of independent interest (see e.g., [31]).

We first develop intuition by sketching out the form that $\mathcal{P}(\boldsymbol{w}; \mathcal{C}(\boldsymbol{\beta}))$ must take, and then describe how Algorithm 3 computes this projection efficiently. This is stated formally in Theorem 3, the proof of which is given in Appendix C. Firstly consider the case that $\boldsymbol{w} \in \mathcal{C}(\boldsymbol{\beta})$, then trivially $\mathcal{P}(\boldsymbol{w}; \mathcal{C}(\boldsymbol{\beta})) = \boldsymbol{w}$, due to the non-negativity of $D(\boldsymbol{u}, \boldsymbol{w})$ and the fact that $D(\boldsymbol{u}, \boldsymbol{w}) = 0$ iff $\boldsymbol{u} = \boldsymbol{w}$ [6]. For the case that $\boldsymbol{w} \notin \mathcal{C}(\boldsymbol{\beta})$, this implies that the set $\{i \in [n] : w_i < \beta_i\}$ is non-empty. For each index i in this set the projection of $\boldsymbol{w}$ onto $\mathcal{C}(\boldsymbol{\beta})$ must set the component w_i to its corresponding constraint value β_i. The remaining components are then normalized, such that $\sum_{i=1}^n w_i = 1$. However, doing so may cause one (or more) of these components w_j to drop below its constraint β_j. In Appendix C we prove that the projection algorithm must find the set of components Ψ of least cardinality to set to their constraint values such that when the remaining components are normalized, no component lies below its constraint, and that this can be done in linear time.

Consider the following inefficient approach to finding Ψ. Given $\boldsymbol{w}$ and $\mathcal{C}(\boldsymbol{\beta})$, let $\boldsymbol{r} = \boldsymbol{w} \odot \frac{1}{\boldsymbol{\beta}}$ be a "ratio vector". First sort $\boldsymbol{r}$ in ascending order, and then sort $\boldsymbol{w}$ and $\boldsymbol{\beta}$ according to the ordering of $\boldsymbol{r}$. If $r_1 \geq 1$ then $\Psi = \emptyset$ and we are done ($\Rightarrow \boldsymbol{w} \in \mathcal{C}(\boldsymbol{\beta})$). Otherwise for each $a = 1, \ldots, n$: 1) let the candidate set $\Psi^{'} = [a]$, 2) let $\boldsymbol{w}' = \boldsymbol{w}$ except for each $i \in \Psi^{'}$ set $w_i' = \beta_i$, 3) re-normalize the remaining components of $\boldsymbol{w}'$, and 4) let $\boldsymbol{r}' = \boldsymbol{w}' \odot \frac{1}{\boldsymbol{\beta}}$. The set Ψ is then the candidate set $\Psi^{'}$ of least cardinality such that $\boldsymbol{r}' \succeq \mathbf{1}$. This approach requires sorting $\boldsymbol{r}$ and therefore even an efficient implementation takes $\mathcal{O}(n \log n)$ time. Algorithm 3 finds Ψ without having to sort $\boldsymbol{r}$. It instead specifies Ψ uniquely with a threshold, ϕ, such that $\Psi = \{i : r_i < \phi\}$. Algorithm 3 finds ϕ through repeatedly bisecting the set $\mathcal{W} = [n]$ by finding the median of the set $\{r_i : i \in \mathcal{W}\}$ (which can be done in $\mathcal{O}(|\mathcal{W}|)$ time [3]), and efficiently testing this value as the candidate threshold on each iteration. The smallest valid threshold then specifies the set Ψ. The following theorem states the time complexity of the algorithm and the form of the projection, which is used in the proof of Theorem 2 (the proof of Theorem 3 is in Appendix C, where we also give a more detailed description of Algorithm 3).

Theorem 3. *For any $\boldsymbol{\beta} \in (0,1)^n$ such that $\|\boldsymbol{\beta}\|_1 \leq 1$, and for any $\boldsymbol{w} \in \operatorname{ri} \Delta_n$, let $\boldsymbol{p} = \mathcal{P}(\boldsymbol{w}; \mathcal{C}(\boldsymbol{\beta}))$, where $\mathcal{C}(\boldsymbol{\beta}) = \{\boldsymbol{x} \in \Delta_n : x_i \geq \beta_i, i = 1, \ldots, n\}$. Then $\boldsymbol{p}$ is such that for all $i = 1, \ldots, n$,*

$$p_i = \max \left\{ \beta_i ; \frac{1 - \sum_{j \in \Psi} \beta_j}{1 - \sum_{j \in \Psi} w_j} w_i \right\} , \tag{19}$$

where $\Psi := \{i \in [n] : p_i = \beta_i\}$. Furthermore, Algorithm 3 computes $\boldsymbol{p}$ in $\mathcal{O}(n)$ time.

3.3 Projection vs. sharing in online learning

We now briefly consider the two types of updates discussed in this paper (projection and weight-sharing) when updating weights may incur costs. Recall the motivating example introduced in Section 2 was in online portfolio selection with transaction costs. It is straightforward to show that in this model transaction costs are proportional to the 1-norm of the difference in the weight vectors before and after re-balancing. In Theorem 4 we give a result which in this context guarantees the "cost" of projecting is less than that of weight-sharing.

To compare the update of PoDS and the generalized share update (8), we must consider for a set of weights $\dot{\boldsymbol{w}}^t$, the point $\mathcal{P}(\dot{\boldsymbol{w}}^t; \mathcal{C}(\boldsymbol{\beta}^t))$ and the point $(1-\alpha)\dot{\boldsymbol{w}}^t + \alpha \boldsymbol{v}^t$. However these points depend on $\boldsymbol{\beta}^t$ and $\boldsymbol{v}^t$ respectively, which may themselves be functions of previous weight vectors $\dot{\boldsymbol{w}}^1, \ldots, \dot{\boldsymbol{w}}^{t-1}$, which as discussed are generally not the same for each of the two algorithms. To compare the two updates equally we therefore assume that the current weights are the same (i.e., they must both update the same weights $\dot{\boldsymbol{w}}^t$), and additionally that $\boldsymbol{\beta}^t = \alpha \boldsymbol{v}^t$. The following theorem states that under mild conditions, PoDS is strictly less "expensive" than its weight-sharing counterpart.

Theorem 4. *Let $0 < \alpha < 1$. Then for any $\boldsymbol{v} \in \operatorname{ri} \Delta_n$, let $\boldsymbol{\beta} = \alpha \boldsymbol{v}$, and for any $\boldsymbol{w} \in \operatorname{ri} \Delta_n$ such that $\boldsymbol{w} \neq \boldsymbol{v}$, let $\boldsymbol{w}' = (1-\alpha)\boldsymbol{w} + \alpha \boldsymbol{v}$. Then,*

$$\|\mathcal{P}(\boldsymbol{w}; \mathcal{C}(\boldsymbol{\beta})) - \boldsymbol{w}\|_1 < \|\boldsymbol{w}' - \boldsymbol{w}\|_1 \ .$$

Thus if one has to pay to update weights, projection is the economical choice.

4 A geometrically-decaying mixing scheme for MPP

In this section we look more closely at Share-θ. We show that it is in fact a new type of *decaying* MPP mixing scheme which corresponds to the partition specialist algorithm with Markov prior.

Recall that the previous best known mixing scheme for MPP is the decaying scheme (6). Observe that in (6) the decay (with the "distance" to the current trial t) follows a power-law, and that computing (6) exactly takes $\mathcal{O}(nt)$ time per trial. We now derive an explicit MPP mixing scheme from the updates (17) and (18) of Share-θ. Observe that if we define $\dot{\boldsymbol{w}}^0 := \frac{\mathbf{1}}{n}$, then an iterative expansion of (18) on any trial t gives $\boldsymbol{v}^t = \sum_{q=0}^{t-1} \theta^{[q\neq 0]}(1-\theta)^{t-q-1}\dot{\boldsymbol{w}}^q$, from which (17) implies $\boldsymbol{w}^{t+1} = (1-\alpha)\dot{\boldsymbol{w}}^t + \alpha\boldsymbol{v}^t = \sum_{q=0}^{t} \gamma_q^{t+1}\dot{\boldsymbol{w}}^q$, where

$$\gamma_q^{t+1} = \begin{cases} 1-\alpha & q=t \\ \theta(1-\theta)^{t-q-1}\alpha & 1 \le q < t \\ (1-\theta)^{t-1}\alpha & q=0\,. \end{cases} \tag{20}$$

Note that (20) is a valid mixing scheme since for all t, $\sum_{q=0}^{t} \gamma_q^{t+1} = 1$. The Share-$\theta$ update is therefore a new kind of decaying mixing scheme. In this new scheme the decay is *geometric*, and can therefore be computed efficiently, requiring only $\mathcal{O}(n)$ time and space per trial as we have shown. Furthermore MPP with this scheme has the improved regret bound (15).

Another interesting difference between the decaying schemes (20) and (6) is that when θ is small then (20) keeps γ_0^{t+1} relatively large initially and slowly decays this value as t increases. Intuitively by heavily weighting the initial uniform vector $\dot{\boldsymbol{w}}^0$ on each trial early on, the algorithm can "pick up" the weights of new experts easily. Finally as in the case of PoDS-θ, if $m = k+1$, then with the optimal tuning of $\theta = 0$, this update reduces to the Fixed-Share update (2).

Revisiting partition specialists. We now turn our attention to the previous best known result for tracking experts with memory (the partition specialists algorithm with a Markov prior [27]).

For sleep/wake patterns $(\chi_1 \ldots \chi_T)$ the Markov prior is a Markov chain on states $\{w, s\}$, defined by the initial distribution $\boldsymbol{\pi} = (\pi_w, \pi_s)$ and transition probabilities $P_{ij} := P(\chi_{t+1} = j | \chi_t = i)$ for $i, j \in \{w, s\}$. The algorithm with these inputs efficiently collapses one weight per specialist down to two weights per expert. These two weight vectors, which we denote $\boldsymbol{a}_t$ and $\boldsymbol{s}_t$, represent the total weight of all awake and sleeping specialists associated with each expert, respectively. Note that the vectors $\boldsymbol{a}_t$ and $\boldsymbol{s}_t$ are not in Δ_n, but rather the vector $(\boldsymbol{a}_t, \boldsymbol{s}_t) \in \Delta_{2n}$ and the "awake vector" $\boldsymbol{a}_t$ gets normalized upon prediction. The weights are initialized by setting $\boldsymbol{a}_1 = \pi_w \frac{\mathbf{1}}{n}$, and $\boldsymbol{s}_1 = \pi_s \frac{\mathbf{1}}{n}$. The update[3] of these weights after receiving the true label y^t is given by

$$a_i^{t+1} = P_{ww} \frac{a_i^t e^{-\eta \ell_i^t} (\sum_{j=1}^n a_j^t)}{\sum_{j=1}^n a_j^t e^{-\eta \ell_j^t}} + P_{sw} s_i^t, \qquad \text{and} \qquad s_i^{t+1} = P_{ws} \frac{a_i^t e^{-\eta \ell_i^t} (\sum_{j=1}^n a_j^t)}{\sum_{j=1}^n a_j^t e^{-\eta \ell_j^t}} + P_{ss} s_i^t$$

for $i = 1, \ldots, n$. Recall that the authors of [27] proved that an MPP mixing scheme implicitly induces a prior over partition specialists. The following states that the Markov prior is induced by (20).

Proposition 5. *Let $0 < \alpha < 1$, and $0 < \theta < 1$. Then the partition specialists algorithm with Markov prior parameterized with $P_{sw} = \theta$, $P_{ws} = \alpha$, $\pi_w = \frac{\theta}{\alpha+\theta}$, and $\pi_s = \frac{\alpha}{\alpha+\theta}$ is equivalent to Share-θ parameterized with α and θ.*

The proof (given in Appendix E) amounts to showing for all t that $\frac{\boldsymbol{a}_t}{\pi_w} = \boldsymbol{w}^t$ and $\frac{\boldsymbol{s}_t}{\pi_s} = \boldsymbol{v}^t$. The Markov prior on partition specialists therefore corresponds to a geometrically-decaying MPP mixing scheme! Note however that we have proved a better regret bound for this algorithm in Theorem 2.

5 Discussion

We gave an efficient projection-based algorithm for tracking experts with memory for which we proved the best known regret bound. We also gave an algorithm to compute relative entropy projection onto the simplex with non-uniform (lower) box constraints exactly in $\mathcal{O}(n)$ time, which may be of independent interest. We showed that the weight-sharing equivalent of our projection-based algorithm is in fact a geometrically-decaying mixing scheme for *Mixing Past Posteriors* [4]. Furthermore we showed that this mixing scheme corresponds exactly to the previous best known result (the partition specialists algorithm with Markov prior [27]), and we therefore improved their bound. We proved a guarantee favoring projection updates over weight-sharing when updating weights may incur costs, such as in online portfolio selection with proportional transaction costs. We are currently applying PoDS-θ to this problem, primarily extending the work of [44] in the sense of incorporating both the assumption of "memory" and transaction costs.

[3] In [27] the algorithm is presented in terms of probabilities with the log loss. Here we give the update generalized to (c, η)-realizable losses.

In this work we focused on proving good regret bounds, which naturally required optimally-tuned parameters. A limitation of our work is that in practice the optimal parameters are unknown. This is a common issue in online learning, and one may employ standard techniques to address this such as the "doubling trick", or by using a Bayesian mixture over parameters [46]. For a prominent recent result in this area see [26].

Finally, the work of [27] gave a Bayesian interpretation to MPP, however this is lost when one uses the projection update of PoDS. We ask: Is there also a Bayesian interpretation to these projection-based updates?

Ethical considerations. While the scope of applicability of online learning algorithms is wide, this research in regret-bounded online learning is foundational in nature and we therefore cannot foresee the extent of any societal impacts (positive or negative) this research may have.

Acknowledgments and Disclosure of Funding

The authors would like to thank the anonymous reviewers for their feedback, insights, and discussion. The authors would also like to thank Dmitry Adamskiy for valuable discussions. This work was supported by the UK Engineering and Physical Sciences Research Council (EPSRC) grant EP/N509577/1.

References

[1] D. Adamskiy, W. M. Koolen, A. Chernov, and V. Vovk. A closer look at adaptive regret. *The Journal of Machine Learning Research*, 17(1):706–726, 2016.

[2] A. Blum and A. Kalai. Universal portfolios with and without transaction costs. *Machine Learning*, 35(3):193–205, 1999.

[3] M. Blum, R. W. Floyd, V. R. Pratt, R. L. Rivest, and R. E. Tarjan. Time bounds for selection. *J. Comput. Syst. Sci.*, 7(4):448–461, 1973.

[4] O. Bousquet and M. K. Warmuth. Tracking a small set of experts by mixing past posteriors. *Journal of Machine Learning Research*, 3(Nov):363–396, 2002.

[5] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[6] L. M. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR computational mathematics and mathematical physics*, 7(3):200–217, 1967.

[7] N. Cesa-Bianchi, P. Gaillard, G. Lugosi, and G. Stoltz. Mirror descent meets fixed share (and feels no regret). In *Conference on Neural Information Processing Systems*, volume 2, pages 989–997, 2012.

[8] A. Daniely, A. Gonen, and S. Shalev-Shwartz. Strongly adaptive online learning. In *International Conference on Machine Learning*, pages 1405–1411. PMLR, 2015.

[9] P. Das, N. Johnson, and A. Banerjee. Online lazy updates for portfolio selection with transaction costs. In *AAAI*. Citeseer, 2013.

[10] R. W. Floyd and R. L. Rivest. Expected time bounds for selection. *Communications of the ACM*, 18(3):165–172, 1975.

[11] R. M. French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

[12] Y. Freund. Private Communication, 2000. Also posted on `http://www.learning-theory.org/`.

[13] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

[14] Y. Freund, R. E. Schapire, Y. Singer, and M. K. Warmuth. Using and combining predictors that specialize. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 334–343, 1997.

[15] R. B. Gramacy, M. K. Warmuth, S. Brandt, and I. Ari. Adaptive caching by refetching. *Advances in Neural Information Processing Systems*, 15:1489–1496, 2002.

[16] A. György, T. Linder, and G. Lugosi. Tracking the best of many experts. In *International Conference on Computational Learning Theory*, pages 204–216. Springer, 2005.

[17] A. Gyorgy, T. Linder, and G. Lugosi. Efficient tracking of large classes of experts. *IEEE Transactions on Information Theory*, 58(11):6709–6725, 2012.

[18] D. Haussler, J. Kivinen, and M. K. Warmuth. Sequential prediction of individual sequences under general loss functions. *IEEE Transactions on Information Theory*, 44(5):1906–1925, 1998.

[19] E. Hazan and C. Seshadhri. Efficient learning algorithms for changing environments. In *Proceedings of the 26th annual international conference on machine learning*, pages 393–400, 2009.

[20] M. Herbster, S. Pasteris, and M. Pontil. Predicting a switching sequence of graph labelings. *J. Mach. Learn. Res.*, 16:2003–2022, 2015.

[21] M. Herbster, S. Pasteris, and L. Tse. Online multitask learning with long-term memory. In *Advances in Neural Information Processing Systems*, volume 33, pages 17779–17791, 2020.

[22] M. Herbster and J. Robinson. Online prediction of switching graph labelings with cluster specialists. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[23] M. Herbster and M. K. Warmuth. Tracking the best expert. *Machine learning*, 32(2):151–178, 1998.

[24] M. Herbster and M. K. Warmuth. Tracking the best linear predictor. *Journal of Machine Learning Research*, 1(Sep):281–309, 2001.

[25] D. Hoeven, T. Erven, and W. Kotłowski. The many faces of exponential weights in online learning. In *Conference On Learning Theory*, pages 2067–2092. PMLR, 2018.

[26] K.-S. Jun, F. Orabona, S. Wright, and R. Willett. Improved strongly adaptive online learning using coin betting. In *Artificial Intelligence and Statistics*, pages 943–951. PMLR, 2017.

[27] W. M. Koolen, D. Adamskiy, and M. K. Warmuth. Putting bayes to sleep. In *NIPS*, pages 135–143, 2012.

[28] W. M. Koolen and S. de Rooij. Universal codes from switching strategies. *IEEE Transactions on Information Theory*, 59(11):7168–7185, 2013.

[29] W. M. Koolen and T. van Erven. Freezing and sleeping: Tracking experts that learn by evolving past posteriors. *CoRR*, abs/1008.4654, 2010.

[30] S. S. Kozat and A. C. Singer. Universal switching portfolios under transaction costs. In *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5404–5407. IEEE, 2008.

[31] W. Krichene, S. Krichene, and A. Bayen. Efficient bregman projections onto the simplex. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 3291–3298. IEEE, 2015.

[32] B. Li and S. C. Hoi. Online portfolio selection: A survey. *ACM Computing Surveys (CSUR)*, 46(3):1–36, 2014.

[33] B. Li, J. Wang, D. Huang, and S. C. Hoi. Transaction cost optimization for online portfolio selection. *Quantitative Finance*, 18(8):1411–1424, 2018.

[34] N. Littlestone and M. Warmuth. The weighted majority algorithm. In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science*, pages 256–261, 1989.

[35] N. Littlestone and M. K. Warmuth. The weighted majority algorithm. *Information and computation*, 108(2):212–261, 1994.

[36] M. McCloskey and N. J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

[37] N. Merhav and M. Feder. Universal prediction. *IEEE Transactions on Information Theory*, 44(6):2124–2147, 1998.

[38] J. Mourtada and O.-A. Maillard. Efficient tracking of a growing number of experts. In *International Conference on Algorithmic Learning Theory*, pages 517–539. PMLR, 2017.

[39] H. T. Nguyen and K. Franke. Adaptive intrusion detection system via online machine learning. In *2012 12th International Conference on Hybrid Intelligent Systems (HIS)*, pages 271–277. IEEE, 2012.

[40] G. I. Shamir. On asymptotically optimal sequential lossless coding for memoryless switching sources. In *Proceedings IEEE International Symposium on Information Theory,*, page 124. IEEE, 2002.

[41] G. I. Shamir and D. J. Costello. Universal lossless coding for sources with repeating statistics. *IEEE transactions on information theory*, 50(8):1620–1635, 2004.

[42] D. Sharma, M.-F. Balcan, and T. Dick. Learning piecewise lipschitz functions in changing environments. In *International Conference on Artificial Intelligence and Statistics*, pages 3567–3577. PMLR, 2020.

[43] Y. M. Shtarkov. Switching discrete sources and its universal encoding. *Probl. Inform. Transm.*, 28(3):95–111, 1992.

[44] Y. Singer. Switching portfolios. *International Journal of Neural Systems*, 8(04):445–455, 1997.

[45] V. Vovk. A game of prediction with expert advice. *Journal of Computer and System Sciences*, 56(2):153–173, 1998.

[46] V. Vovk. Derandomizing stochastic prediction strategies. *Machine Learning*, 35(3):247–282, 1999.

[47] V. G. Vovk. Aggregating strategies. *Proc. of Computational Learning Theory, 1990*, 1990.

[48] Q. Xie and A. R. Barron. Asymptotic minimax regret for data compression, gambling, and prediction. *IEEE Transactions on Information Theory*, 46(2):431–445, 2000.

[49] K. Zheng, H. Luo, I. Diakonikolas, and L. Wang. Equipping experts/bandits with long-term memory. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

A Proof of Corollary 1

Proof. Let $\Psi := \{i \in [n] : p_i = \beta_i\}$. Recall from Theorem 3 that the projected vector $\boldsymbol{p}$ takes the form (19). Expanding the relative entropy terms of (14) then gives the following,

$$\begin{aligned}
D(\boldsymbol{u}, \boldsymbol{w}) - D(\boldsymbol{u}, \boldsymbol{p}) &= \sum_{i=1}^{n} u_i \ln\left(\frac{p_i}{w_i}\right) \\
&\geq \sum_{i=1}^{n} u_i \ln\left(\frac{\left(1 - \sum_{j\in\Psi} \beta_j\right) w_i}{\left(1 - \sum_{j\in\Psi} w_j\right) w_i}\right) \\
&= \ln\left(\frac{1 - \sum_{j\in\Psi} \beta_j}{1 - \sum_{j\in\Psi} w_j}\right) \\
&\geq \ln(1-\alpha)\,,
\end{aligned}$$

where the first inequality follows from the definition of p_i in (19) and the fact that $\max\{a, b\} \geq b$. The second inequality follows from the fact that $\sum_{j\in\Psi} w_j \geq 0$ and $\sum_{j\in\Psi} \beta_j \leq \alpha$. $\square$

B Proof of Theorem 2

Proof. We first prove the bound for PoDS-θ, and then prove that Share-θ has the same bound. We use the relative entropy $D(\boldsymbol{u}^t, \boldsymbol{w}^t)$ as a measure of progress of the algorithm, where $\boldsymbol{u}^t$ is a comparator vector which we take to be a basis vector $\boldsymbol{e}_i$ for some $i \in [n]$ corresponding to the locally best expert i_t in hindsight on trial t. Recall that the comparator sequence $i_1, \ldots, i_T$ is partitioned with k *switches* into $k+1$ *segments*, where a segment is defined as a sequence of trials where the comparator is unchanged, i.e. $i_a = \ldots = i_b$ for some $a < b$.

Recall that pred and ℓ are assumed to be $(c, \frac{1}{c})$-realizable. That is, for any $\boldsymbol{w}^t \in \Delta_n$, $\boldsymbol{x}^t \in \mathcal{D}^n$, and $y^t \in \mathcal{Y}$, there exists $\eta > 0$ such that

$$\ell(\texttt{pred}(\boldsymbol{w}, \boldsymbol{x}), y) \leq -c \ln \sum_{i=1}^{n} w_i e^{-\eta \ell(x_i, y)} \tag{21}$$

holds with $c\eta = 1$.

We first establish that

$$\ell^t - \ell^t_{i_t} \leq c\left(D(\boldsymbol{u}^t, \boldsymbol{w}^t) - D(\boldsymbol{u}^t, \dot{\boldsymbol{w}}^t)\right) \tag{22}$$

holds for all t. Expanding the relative entropy terms gives

$$\begin{aligned}
D(\boldsymbol{u}^t, \boldsymbol{w}^t) - D(\boldsymbol{u}^t, \dot{\boldsymbol{w}}^t) &= \sum_{i=1}^{n} u_i^t \ln \frac{\dot{w}_i^t}{w_i^t} \\
&= \sum_{i=1}^{n} u_i^t \ln \frac{w_i^t e^{-\eta \ell_i^t}}{w_i^t \sum_{j=1}^{n} w_j^t e^{-\eta \ell_j^t}} \\
&= -\eta \sum_{i=1}^{n} u_i^t \ell_i^t - \ln \sum_{j=1}^{n} w_j^t e^{-\eta \ell_j^t} \\
&\geq -\eta \ell^t_{i_t} + \frac{1}{c}\ell^t\,,
\end{aligned}$$

where the inequality follows from (21). Multiplying both sides by c gives (22).

We now find lower bounds, δ, for $D(\boldsymbol{u}^t, \dot{\boldsymbol{w}}^t) - D(\boldsymbol{u}^{t+1}, \boldsymbol{w}^{t+1})$ to give non-negative terms of the form $D(\boldsymbol{u}^t, \dot{\boldsymbol{w}}^t) - D(\boldsymbol{u}^{t+1}, \boldsymbol{w}^{t+1}) - \delta \geq 0$, which we will multiply by c and add to (22) to give a telescoping sum of relative entropy terms. We consider three distinct cases for the different values of $\boldsymbol{u}^t$ over the T trials.

For the first case, we consider when there is no switch immediately after trial t (i.e., $\boldsymbol{u}^t = \boldsymbol{u}^{t+1}$). We use Corollary 1 with $\boldsymbol{u} = \boldsymbol{u}^t$, $\boldsymbol{w} = \dot{\boldsymbol{w}}^t$, and $\boldsymbol{\beta} = \boldsymbol{\beta}^t$. It follows then by definition that $\boldsymbol{p} = \boldsymbol{w}^{t+1}$ and we obtain

$$D(\boldsymbol{u}^t, \dot{\boldsymbol{w}}^t) - D(\boldsymbol{u}^{t+1}, \boldsymbol{w}^{t+1}) \geq \ln(1-\alpha)\,, \tag{23}$$

which gives a telescoping sum of relative entropy terms within in each segment, paying $c\ln(1/(1-\alpha))$ for every trial where $\boldsymbol{u}^t = \boldsymbol{u}^{t+1}$.

For the two remaining cases, we will consider the segment boundaries, that is, the case when there is a switch and $\boldsymbol{u}^t \neq \boldsymbol{u}^{t+1}$. W.l.o.g let $\boldsymbol{u}^t = \boldsymbol{e}_j$ and let $\boldsymbol{u}^{t+1} = \boldsymbol{e}_k$ for any $j \neq k$ (that is we switch from expert "j" to expert "k" after trial t). We then have the following

$$D(\boldsymbol{u}^t, \dot{\boldsymbol{w}}^t) - D(\boldsymbol{u}^{t+1}, \boldsymbol{w}^{t+1}) = \sum_{i=1}^{n} u_i^t \ln \frac{u_i^t}{\dot{w}_i^t} - \sum_{i=1}^{n} u_i^{t+1} \ln \frac{u_i^{t+1}}{w_i^{t+1}} = \ln \frac{1}{\dot{w}_j^t} + \ln w_k^{t+1}\,, \tag{24}$$

thus we collect a $\ln(1/\dot{w}_j^t)$ term from the *last* trial of the segment of expert j and a $\ln(w_k^{t+1})$ term from the *first* trial of the new segment of expert k. We now consider the remaining two cases: when trial $t+1$ is the first time expert k predicts well, and when trial $t+1$ is a trial on which we "re-visit" expert k.

For the first of these two cases, we consider the first time expert k starts to predict well. We then use (16) and (13) to give

$$\ln w_k^{t+1} \geq \ln \beta_k^t \geq \ln\left((1-\theta)^{t-1}\beta_k^1\right) = \ln\left((1-\theta)^{t-1}\frac{\alpha}{n}\right). \tag{25}$$

Substituting (25) into (24), we therefore pay $-c\ln\left((1-\theta)^{t-1}\frac{\alpha}{n}\right)$ to switch to a new expert for the first time on trial $t+1$.

Finally for the second of these two cases, we consider when expert k has predicted well before. Let trial $q < t$ denote the *last* trial of expert k's most recent "segment". We then have the following (again using (16) and (13)),

$$\ln w_k^{t+1} \geq \ln \beta_k^t \geq \ln\left((1-\theta)^{t-q-1}\beta_k^{q+1}\right) \geq \ln\left((1-\theta)^{t-q-1}\alpha\theta\dot{w}_k^q\right). \tag{26}$$

By substituting (26) into (24) for each segment boundary, and summing over these boundaries, we therefore pay $-c\ln\left((1-\theta)^{t-q-1}\alpha\theta\right)$ in order to telescope the $\ln(\dot{w}_k^q)$ term with the $\ln(1/\dot{w}_k^q)$ term from the end of expert k's most recent segment ending on trial q.

Putting these together we thus pay $c\ln(1/(1-\alpha))$ for every trial on which we don't switch (from Corollary 1), we pay $c\ln(1/(1-\theta))$ for every expert in our pool that *isn't* predicting well or involved in a switch on every trial (i.e., $m-1$ times, on non-switch trials, and $m-2$ times on switch trials, from (25) and (26)), and finally when we switch to an expert k before trial $t+1$ we pay $c\ln(n/\alpha)$ if it is the first time to track expert k (there are $m-1$ such trials), and $c\ln(1/\alpha\theta)$ otherwise (there are $k-m+1$ such trials).

Summing over all trials, and using $D(\boldsymbol{u}^1, \boldsymbol{w}^1) \leq \ln n$ then gives

$$\begin{aligned}
\sum_{t=1}^{T} \ell^t - \sum_{t=1}^{T} \ell_{i_t}^t &\leq \sum_{t=1}^{T} c\left(D(\boldsymbol{u}^t, \boldsymbol{w}^t) - D(\boldsymbol{u}^t, \dot{\boldsymbol{w}}^t) + D(\boldsymbol{u}^t, \dot{\boldsymbol{w}}^t) - D(\boldsymbol{u}^{t+1}, \boldsymbol{w}^{t+1})\right) \\
&\leq cD(\boldsymbol{u}^1, \boldsymbol{w}^1) + c(T-k-1)\ln\left(\frac{1}{1-\alpha}\right) + c(m-1)\ln\left(\frac{n}{\alpha}\right) \\
&\quad + c((m-1)(T-1)-k)\ln\left(\frac{1}{1-\theta}\right) + c(k-m+1)\ln\left(\frac{1}{\alpha\theta}\right) \\
&\leq cm\ln n + c(T-k-1)\ln\left(\frac{1}{1-\alpha}\right) + ck\ln\left(\frac{1}{\alpha}\right) \\
&\quad + c((m-1)(T-1)-k)\ln\left(\frac{1}{1-\theta}\right) + c(k-m+1)\ln\left(\frac{1}{\theta}\right).
\end{aligned} \tag{27}$$

The optimal tuning of α and θ that minimizes (27) is given by $\alpha = \frac{k}{T-1}$ and $\theta = \frac{k-m+1}{(m-1)(T-2)}$. Substituting these values into (27) gives a bound of

$$cm\ln n + c(T-1)\mathcal{H}\left(\frac{k}{T-1}\right) + c(m-1)(T-2)\mathcal{H}\left(\frac{k-m+1}{(m-1)(T-2)}\right),$$

which completes the proof for PoDS-θ.

We now prove that Share-θ has the same bound with an almost identical argument as the proof just given for PoDS-θ. Firstly observe that (24) is independent of the algorithm update and therefore holds for both algorithms. Additionally, observe that the proof for PoDS-θ relies on the inequalities (22), (23), (25), and (26). We now prove that these inequalities hold for Share-θ, and thus the two algorithms share the same bound.

Firstly we observe that inequality (22) holds since both algorithms use the same loss update, and we assume that the prediction function and loss function are $(c, \frac{1}{c})$-realizable.

Secondly, it follows directly from the update (17) that (23) holds for Share-θ when $\boldsymbol{u}^t = \boldsymbol{u}^{t+1}$, since $\boldsymbol{w}^{t+1} \geq (1-\alpha)\dot{\boldsymbol{w}}^t$ and therefore

$$D(\boldsymbol{u}^t, \dot{\boldsymbol{w}}^t) - D(\boldsymbol{u}^{t+1}, \boldsymbol{w}^{t+1}) = \sum_{i=1}^n u_i^t \ln \frac{w_i^{t+1}}{\dot{w}_i^t} \geq \sum_{i=1}^n u_i^t \ln \frac{(1-\alpha)\dot{w}_i^t}{\dot{w}_i^t} = \ln(1-\alpha)\,.$$

The proof that (25) holds follows directly from the updates (17) and (18) and the fact $\boldsymbol{v}^1 = \frac{\mathbf{1}}{n}$. That is, for the first time expert "k" appears on trial $t+1$,

$$\ln w_k^{t+1} \geq \ln(\alpha v_k^t) \geq \ln((1-\theta)^{t-1}\alpha v_k^1) = \ln\left((1-\theta)^{t-1}\frac{\alpha}{n}\right).$$

Similarly, the proof that (26) holds follows directly from the updates (17) and (18). That is, when we return to expert "k" on trial $t+1$,

$$\ln w_k^{t+1} \geq \ln(\alpha v_k^t) \geq \ln((1-\theta)^{t-q-1}\alpha v_k^{q+1}) \geq \ln((1-\theta)^{t-q-1}\alpha\theta\dot{w}_k^q)\,.$$

Having shown that the inequalities (22), (23), (25), and (26) hold for Share-θ, the remainder of the proof follows exactly as the proof for PoDS-θ. $\square$

C Proof of Theorem 3

A note on the proof: The proof of the theorem follows very closely to the proof of Theorem 7 in [24] (including Claims 1, 2, and 3). There the problem is concerned with uniform constraints, whereas we consider non-uniform constraints. In particular Claims 6 and 7 given below are generalizations of Claims 2 and 3 of [24]. The proof of the second statement of Theorem 3 is almost identical to the proof of Theorem 7 in [24]. We first give a sketch of the proof of the two statements of Theorem 3.

For the first statement, recall that $\Psi := \{i \in [n] : p_i = \beta_i\}$ is the set of indexes of components which must be set to their constraint values. To prove the first statement we will show that given $\boldsymbol{w}$ and $\mathcal{C}(\boldsymbol{\beta})$, each component of the point $\mathcal{P}(\boldsymbol{w}; \mathcal{C}(\boldsymbol{\beta}))$ either takes the value of its lower box constraint, β_i, or is equal to w_i multiplied by a factor λ, with

$$\lambda = \frac{1 - \sum_{i\in\Psi}\beta_i}{1 - \sum_{i\in\Psi} w_i}\,.$$

We then argue that each component $p_i = \max\{\beta_i; \lambda w_i\}$ for $i = 1, \ldots, n$.

For the second statement, we first show that Ψ, which uniquely specifies $\mathcal{P}(\boldsymbol{w}; \mathcal{C}(\boldsymbol{\beta}))$, is the set of minimum cardinality such that when all other components are re-normalized, no component lies below its constraint value, and then show that Algorithm 3 finds this set in $\mathcal{O}(n)$ time.

Proof of the first statement of Theorem 3. Recall the first statement of the theorem: that $\mathcal{P}(\boldsymbol{w}; \mathcal{C}(\boldsymbol{\beta}))$ takes the form (19). Given $\boldsymbol{w}$ and the non-empty set $\mathcal{C}(\boldsymbol{\beta})$, the point $\mathcal{P}(\boldsymbol{w}; \mathcal{C}(\boldsymbol{\beta}))$ is the minimizer of the following convex optimization problem

$$\begin{aligned} \min_{\boldsymbol{u}} \quad & D(\boldsymbol{u}, \boldsymbol{w}) \\ \text{s.t.} \quad & \beta_i - u_i \leq 0, \quad i = 1, \ldots, n \\ & \mathbf{1}\cdot\boldsymbol{u} - 1 = 0 \quad . \end{aligned} \tag{28}$$

Since $D(\boldsymbol{u}, \boldsymbol{w})$ is convex in its first argument, and $\mathcal{C}(\boldsymbol{\beta})$ is a convex set, then (28) has a unique minimizer, which we denote by $\boldsymbol{p}$.

Constructing the Lagrangian of (28) with Lagrange multipliers $\boldsymbol{\xi} \succeq \mathbf{0}, \nu \in \mathbb{R}$,

$$\mathcal{L}(\boldsymbol{u}, \boldsymbol{\xi}, \nu) = \sum_{i=1}^{n} u_i \ln \frac{u_i}{w_i} + \boldsymbol{\xi}^\top(\boldsymbol{\beta} - \boldsymbol{u}) + \nu(\mathbf{1} \cdot \boldsymbol{u} - 1)\,,$$

and setting $\nabla_{\boldsymbol{u}} \mathcal{L}(\boldsymbol{u}, \boldsymbol{\xi}, \nu) = \mathbf{0}$ gives for $i = 1, \ldots, n$,

$$\frac{\partial \mathcal{L}}{\partial u_i} = \ln \frac{u_i}{w_i} + 1 - \xi_i + \nu = 0\,.$$

This then gives for $i = 1, \ldots, n$,

$$p_i = w_i e^{\xi_i - 1 - \nu}\,.$$

Since $D(\boldsymbol{u}, \boldsymbol{w})$ is convex in its first argument, and (28) has only linear constraints then strong duality holds and we may exploit the complementary slackness Karush-Kuhn-Tucker necessary condition of the optimal solution (see e.g., [5, Chapter 5]). That is, $\xi_i(\beta_i - p_i) = 0$ for all $i = 1, \ldots, n$. Therefore for any i such that $p_i > \beta_i$, the corresponding Lagrange multiplier is zero, and we have

$$p_i = w_i e^{-1-\nu}\,.$$

Recall $\Psi = \{i : p_i = \beta_i\}$, we then have

$$1 = \sum_{i=1}^{n} p_i = \sum_{i \in \Psi} p_i + \sum_{i \in [n] \setminus \Psi} p_i = \sum_{i \in \Psi} \beta_i + \sum_{i \in [n] \setminus \Psi} w_i e^{-1-\nu}\,.$$

Re-arranging gives

$$e^{-1-\nu} = \frac{1 - \sum_{i \in \Psi} \beta_i}{\sum_{i \in [n] \setminus \Psi} w_i} = \frac{1 - \sum_{i \in \Psi} \beta_i}{1 - \sum_{i \in \Psi} w_i}\,.$$

Therefore for each index $i \in [n]$, either i is in Ψ which implies $p_i = \beta_i$, or $i \notin \Psi$ and therefore $p_i = \lambda w_i$, where

$$\lambda = \frac{1 - \sum_{j \in \Psi} \beta_j}{1 - \sum_{j \in \Psi} w_j}\,.$$

We now establish that $p_i = \max\{\beta_i; \lambda w_i\}$ for all $i = 1, \ldots, n$. Observe that if $i \in \Psi$, then $p_i = w_i e^{\xi_i - 1 - \nu} = \beta_i$, and since the Lagrange multiplier $\xi_i \geq 0$ then $p_i \geq w_i e^{-1-\nu} = \lambda w_i$.

For $i \notin \Psi$, then this implies $p_i = \lambda w_i > \beta_i$, since if $p_i = \beta_i$ then $i \in \Psi$, and if $p_i < \beta_i$ then we have a contradiction since $\boldsymbol{p}$ is not a feasible solution to (28). We therefore conclude that $\boldsymbol{p}$ is such that for all $i = 1, \ldots, n$,

$$p_i = \max\left\{\beta_i; \frac{1 - \sum_{j \in \Psi} \beta_j}{1 - \sum_{j \in \Psi} w_j} w_i\right\}\,,$$

which completes the proof of the first statement of the Theorem. ☐

The proof of the second statement of Theorem 3 will rely on the following two claims.

Claim 6. *Given $\boldsymbol{w}$ and $\boldsymbol{\beta}$, let $\boldsymbol{r} := \boldsymbol{w} \odot \frac{1}{\boldsymbol{\beta}}$. Without loss of generality, for $i < j$ assume $r_i \leq r_j$. Let $\lambda = \frac{1 - \sum_{i \in \Psi} \beta_i}{1 - \sum_{i \in \Psi} w_i}$, then*

$$\boldsymbol{p} = \left(\beta_1, \ldots, \beta_{|\Psi|}, \lambda w_{|\Psi|+1}, \ldots, \lambda w_n\right)\,. \tag{29}$$

Proof. In the proof of the first statement of Theorem 3 we established that $\boldsymbol{p}$ is a permutation of (29), that is, either $p_i = \beta_i$ or $p_i = \lambda w_i$ for $i = 1, \ldots, n$. We also established that $p_i = \max\{\beta_i; \lambda w_i\}$ for $i = 1, \ldots, n$.

Suppose $\boldsymbol{p}$ is not in the form of (29). Then there exists $a < b$ such that $p_a = \lambda w_a$ and $p_b = \beta_b$ (that is, $b \in \Psi$ and $a \notin \Psi$).

If $p_a = \lambda w_a$ then by the first statement of Theorem 3 we have $\lambda w_a > \beta_a$. However since $r_a \leq r_b$, and $\lambda > 0$, this implies $\frac{\lambda w_a}{\beta_a} \leq \frac{\lambda w_b}{\beta_b}$. We then have $1 < \frac{\lambda w_a}{\beta_a} \leq \frac{\lambda w_b}{\beta_b}$, which implies $\lambda w_b > \beta_b$. However we necessarily assumed that $p_b = \beta_b$. This violates the first statement of Theorem 3 that $p_b = \max\{\lambda w_b, \beta_b\}$, and thus contradicts our assumption that $\boldsymbol{p}$ is the minimizer of (28). Hence our supposition that $\boldsymbol{p}$ is not in the form of (29) is false. ☐

Claim 7. *Let* $\Psi' = \{1, \ldots, k\}$*, and* $\Psi'' = \{1, \ldots, k+1\}$*, and let* $\lambda' = \frac{1 - \sum_{i \in \Psi'} \beta_i}{1 - \sum_{i \in \Psi'} w_i}$*, and* $\lambda'' = \frac{1 - \sum_{i \in \Psi''} \beta_i}{1 - \sum_{i \in \Psi''} w_i}$*. Then let*

$$\boldsymbol{u}' = \left(\overbrace{\beta_1, \ldots, \beta_{|\Psi'|}}^{k}, \lambda' w_{|\Psi'|+1}, \ldots, \lambda' w_n \right) ,$$

and

$$\boldsymbol{u}'' = \left(\overbrace{\beta_1, \ldots, \beta_{|\Psi''|}}^{k+1}, \lambda'' w_{|\Psi''|+1}, \ldots, \lambda'' w_n \right) ,$$

then $D(\boldsymbol{u}', \boldsymbol{w}) \leq D(\boldsymbol{u}'', \boldsymbol{w})$*.*

Proof. Consider the following convex optimization problem for some $\boldsymbol{w} \in \operatorname{ri} \Delta_n$,

$$\begin{array}{ll} \min\limits_{\boldsymbol{u}} & D(\boldsymbol{u}, \boldsymbol{w}) \\ \text{s.t.} & \beta_i - u_i = 0, \quad i = 1, \ldots, k \\ & \mathbf{1} \cdot \boldsymbol{u} - 1 = 0 \quad . \end{array} \tag{30}$$

The point $\boldsymbol{u}'$ is the unique minimizer of (30), while $\boldsymbol{u}''$ clearly also satisifies the constraints of (30) and is therefore a feasible solution. This implies that $D(\boldsymbol{u}', \boldsymbol{w}) \leq D(\boldsymbol{u}'', \boldsymbol{w})$. □

Proof of the second statement of Theorem 3. Recall the second statement of the theorem: that Algorithm 3 computes $\mathcal{P}(\boldsymbol{w}; \mathcal{C}(\boldsymbol{\beta}))$ in linear time. We prove this statement by first showing that the set Ψ corresponding to this projection is the set of components of minimal cardinality to set to their constraint values such that when the other components are normalized, no component lies below its constraint value. We then prove that Algorithm 3 computes the projection by finding this set in linear time.

In the proof of the first statement of the theorem we proved that $\boldsymbol{p}$ has the form (19). Thus $\boldsymbol{p}$ is uniquely specified by the set $\Psi = \{i \in [n] : p_i = \beta_i\} \subseteq \{1, \ldots, n\}$. There are therefore 2^n possible solutions. Claim 6 proves that the magnitude of the ratio of a component and its constraint is smaller for a component to be set to its constraint value than a component to be normalized. That is, if $i \in \Psi$ and $j \notin \Psi$, then $\frac{w_i}{\beta_i} \leq \frac{w_j}{\beta_j}$. This reduces the number of feasible solutions to n.

Given these n possible solutions, claim 7 shows that if $\Psi' \subseteq \Psi''$ with corresponding candidate projection vectors $\boldsymbol{u}'$ and $\boldsymbol{u}''$ respectively, then $D(\boldsymbol{u}', \boldsymbol{w}) \leq D(\boldsymbol{u}'', \boldsymbol{w})$. Thus to compute the projection, one must find the set Ψ of minimum cardinality whose corresponding candidate projection vector is in $\mathcal{C}(\boldsymbol{\beta})$.

Observe that this "minimal" set Ψ is specified uniquely by a threshold, ϕ, such that $\Psi = \{i \in [n] : r_i < \phi\}$, where $r_i = \frac{w_i}{\beta_i}$, for $i = 1, \ldots, n$. Algorithm 3 finds Ψ by finding this threshold. The algorithm initially computes the vector $\boldsymbol{r} = \boldsymbol{w} \odot \frac{1}{\boldsymbol{\beta}}$ and when ϕ has been found, the algorithm sets all components of w_i where $r_i < \phi$ to their thresholds β_i, and normalizes the remaining components.

We now discuss how the algorithm finds ϕ in linear time. On each iteration a candidate threshold is examined. These candidate thresholds are determined from an index set $\mathcal{W}$, which is initially set to $\{1, \ldots, n\}$. On each iteration the threshold ϕ is chosen as the median of the ratios in the set $\{r_i : i \in \mathcal{W}\}$ (line 3). This can be done in $\mathcal{O}(|\mathcal{W}|)$ time [3]. The approach used is a divide and conquer method, however from a practical perspective this could also be replaced with a randomized median-finding algorithm with average time complexity $\mathcal{O}(|\mathcal{W}|)$ [10]. If $|\mathcal{W}|$ is even, then the algorithm can choose between the $\frac{|\mathcal{W}|}{2}$ and the $\frac{|\mathcal{W}|+1}{2}$ largest element arbitrarily. The set $\mathcal{W}$ is then sorted into two sets, $\mathcal{L}$ and $\mathcal{H}$, where $\mathcal{L} = \{i \in \mathcal{W} : r_i < \phi\}$ and $\mathcal{H} = \{i \in \mathcal{W} : r_i > \phi\}$.

The normalizing constant λ is then computed (line 9). If $\lambda\phi < 1$, then by Claims 6 and 7 the true threshold must be larger than the current candidate threshold ϕ, and must therefore correspond to r_i for an index i contained in $\mathcal{H}$. Otherwise the true threshold must be either equal to the current candidate threshold, or must correspond to r_i for an index i contained in $\mathcal{L}$.

Since ϕ was taken to be the median, then the algorithm iterates this procedure, setting $\mathcal{W} = \mathcal{L}$ or $\mathcal{W} = \mathcal{H}$ as appropriate. Additionally, since ϕ was taken to be the median, then $\max\{|\mathcal{L}|; |\mathcal{H}|\} \leq \frac{1}{2}|\mathcal{W}|$. When $\mathcal{W} = \emptyset$, then the algorithm has found ϕ, and the projection is computed.

There are a maximum of $\lceil \log n + 1 \rceil$ iterations of lines 2-17, with the i^{th} iteration taking $\mathcal{O}(\frac{n}{2^i})$ time. The algorithm therefore takes $\mathcal{O}(n)$ time to find ϕ, and the time complexity of the algorithm is therefore $\mathcal{O}(n)$. $\square$

D Proof of Theorem 4

Before proving Theorem 4, we introduce some additional notation. Let $\boldsymbol{p} := \mathcal{P}(\boldsymbol{w}; \mathcal{C}(\boldsymbol{\beta}))$, and recall the definition of $\boldsymbol{w}' = (1-\alpha)\boldsymbol{w} + \alpha \boldsymbol{v}$. We then define the following sets,

$$\mathcal{P}_{inc} := \{i \in [n] : p_i > w_i\}, \quad \mathcal{P}_{dec} := \{i \in [n] : p_i \leq w_i\},$$
$$\mathcal{S}_{inc} := \{i \in [n] : w'_i > w_i\}, \quad \mathcal{S}_{dec} := \{i \in [n] : w'_i \leq w_i\}.$$

The subscripts inc and dec correspond to the relative change in the weights before and after the corresponding update - whether they *increase* or *decrease*, respectively.

We first require the following corollary, which follows naturally from Theorem 3.

Corollary 8. *If $i \in \mathcal{P}_{inc}$ then $p_i = \beta_i$.*

Proof. Recall that Theorem 3 states that $\boldsymbol{p}$ is such that for $i = 1, \dots, n$,

$$p_i = \max\{\beta_i; \lambda w_i\},$$

where $\lambda = \frac{1-\sum_{j\in\Psi}\beta_j}{1-\sum_{j\in\Psi} w_j}$ is a normalizing constant. We first establish that $\lambda \leq 1$. Suppose $\lambda > 1$, then this implies $\sum_{i\in\Psi} w_i > \sum_{i\in\Psi}\beta_i$. In this case there must exist $i \in \Psi$ such that $w_i > \beta_i$. However if $\lambda > 1$ then $\lambda w_i > w_i > \beta_i$, but since $i \in \Psi$ then $p_i = \beta_i$, which must be greater than λw_i by Theorem 3. This leads to a contradiction and thus our supposition that $\lambda > 1$ is false.

The form of $\boldsymbol{p}$ implies that $i \in \mathcal{P}_{inc}$ iff $w_i < \beta_i$, since if $w_i \geq \beta_i$ then this implies that either $p_i = \beta_i \leq w_i$ or $p_i = \lambda w_i \leq w_i$, and in both of these cases i must be in $\mathcal{P}_{dec}$. It then follows that if $i \in \mathcal{P}_{inc}$ then $p_i = \beta_i$ since otherwise $p_i = \lambda w_i \leq w_i < \beta_i$ which is a contradiction. $\square$

Recall that is it assumed that $\boldsymbol{w} \neq \boldsymbol{v}$ and thus the definition of $\boldsymbol{w}'$ implies that $\mathcal{S}_{inc}$ is non-empty. We use this fact in the following two lemmas. The first states that if a weight w_i were to increase after the projection update, then it would always increase after the weight-sharing update.

Lemma 9. $\mathcal{P}_{inc} \subseteq \mathcal{S}_{inc}$.

Proof. For any $i \in [n]$ we have

$$w'_i - w_i = (1-\alpha)\, w_i + \alpha v_i - w_i = \alpha\,(v_i - w_i)\ ,$$

and it follows that $i \in \mathcal{S}_{inc}$ iff $w_i < v_i$. Using Corollary 8 we conclude that if $i \in \mathcal{P}_{inc}$, then $w_i < p_i = \beta_i = \alpha v_i < v_i$ and then i must also be in $\mathcal{S}_{inc}$. $\square$

Lemma 10. $\|\boldsymbol{p} - \boldsymbol{w}\|_1 = 2\sum_{i\in\mathcal{P}_{inc}}(p_i - w_i)$*, and* $\|\boldsymbol{w}' - \boldsymbol{w}\|_1 = 2\sum_{i\in\mathcal{S}_{inc}}(w'_i - w_i)$.

Proof. We prove the first equality by observing that

$$\|\boldsymbol{p} - \boldsymbol{w}\|_1 = \sum_{i=1}^{n} |p_i - w_i| = \sum_{i\in\mathcal{P}_{inc}} (p_i - w_i) + \sum_{i\in\mathcal{P}_{dec}} (w_i - p_i)\,,$$

and since the total weight does not change after an update (i.e., $\sum_{i=1}^n p_i = \sum_{i=1}^n w_i$), necessarily we have $\sum_{i\in\mathcal{P}_{inc}}(p_i - w_i) = \sum_{i\in\mathcal{P}_{dec}}(w_i - p_i)$. Since $\sum_{i=1}^n w'_i = \sum_{i=1}^n w_i$, the same argument can be used to prove the second claim. $\square$

Proof of Theorem 4. Using Corollary 8, and the definition of $\boldsymbol{w}'$, we have for $i \in \mathcal{P}_{inc}$,

$$w_i' - w_i = (1-\alpha)w_i + \alpha v_i - w_i = \alpha(v_i - w_i) = \beta_i - \alpha w_i = p_i - \alpha w_i > p_i - w_i\,, \qquad (31)$$

where the inequality arises from the fact that $\alpha < 1$. Finally combining this inequality with Lemmas 9 and 10 gives

$$\begin{aligned}
\|\boldsymbol{p} - \boldsymbol{w}\|_1 &= 2 \sum_{i \in \mathcal{P}_{inc}} (p_i - w_i) && \text{(Lemma 10)} \\
&< 2 \sum_{i \in \mathcal{P}_{inc}} (w_i' - w_i) && \text{(Equation 31)} \\
&\leq 2 \sum_{i \in \mathcal{S}_{inc}} (w_i' - w_i) && \text{(Lemma 9)} \\
&= \|\boldsymbol{w}' - \boldsymbol{w}\|_1\,. && \text{(Lemma 10)}
\end{aligned}$$

□

E Proof of Proposition 5

Proof. It suffices to show that

$$\frac{a_i^t}{\sum_{j=1}^n a_j^t} = w_i^t\,, \qquad (32)$$

and

$$\frac{s_i^t}{\sum_{j=1}^n s_j^t} = v_i^t \qquad (33)$$

for all t. Since the initial distribution, $\boldsymbol{\pi}$, of the Markov chain prior is taken to be the stationary distribution, the detailed balance equation, $P_{ws}\pi_w = P_{sw}\pi_s$, holds for all trials.

It is therefore straightforward to show that $\sum_{i=1}^n a_i^t = \pi_w$ and $\sum_{i=1}^n s_i^t = \pi_s$ for all t. Letting $\alpha = P_{ws}$, and $\theta = P_{sw}$, we proceed to prove that (32) and (33) hold simultaneously for all t by induction. The case for $t = 1$ is trivial. Then by induction on t for $t \geq 1$,

$$\begin{aligned}
\frac{a_i^{t+1}}{\pi_w} &= P_{ww} \frac{a_i^t e^{-\eta \ell_i^t}}{\sum_{j=1}^n a_j^t e^{-\eta \ell_j^t}} + \frac{P_{sw}}{\pi_w} s_i^t \\
&= P_{ww} \frac{a_i^t e^{-\eta \ell_i^t}}{\sum_{j=1}^n a_j^t e^{-\eta \ell_j^t}} + \frac{P_{ws}}{\pi_s} s_i^t \\
&= P_{ww} \dot{w}_i^t + P_{ws} v_i^t && \text{(induction)} \\
&= (1-\alpha)\dot{w}_i^t + \alpha v_i^t \\
&= w_i^{t+1}\,,
\end{aligned}$$

and similarly

$$\begin{aligned}
\frac{s_i^{t+1}}{\pi_s} &= \frac{P_{ws}\pi_w}{\pi_s} \frac{a_i^t e^{-\eta \ell_i^t}}{\sum_{j=1}^n a_j^t e^{-\eta \ell_j^t}} + P_{ss} \frac{s_i^t}{\pi_s} \\
&= P_{sw} \frac{a_i^t e^{-\eta \ell_i^t}}{\sum_{j=1}^n a_j^t e^{-\eta \ell_j^t}} + P_{ss} \frac{s_i^t}{\pi_s} \\
&= P_{sw} \dot{w}_i^t + P_{ss} v_i^t && \text{(induction)} \\
&= \theta \dot{w}_i^t + (1-\theta) v_i^t \\
&= v_i^{t+1}\,.
\end{aligned}$$

We therefore conclude by the inductive argument that (32) and (33) hold for all $t \geq 1$. □