OPT2023: 15th Annual Workshop on Optimization for Machine Learning

High-Dimensional Unbiased Prediction for Sequential Decision Making

Georgy Noarov GNOAROV@SEAS.UPENN.EDU
Ramya Ramalingam RAMYA23@SEAS.UPENN.EDU
Aaron Roth AAROTH@CIS.UPENN.EDU
Stephan Xie STEPHANX@SEAS.UPENN.EDU
University of Pennsylvania

Abstract

We study the problem of making predictions of an adversarially chosen high-dimensional state that are *unbiased* subject to an arbitrary collection of conditioning events, with the goal of tailoring these events to downstream decision makers. We give efficient algorithms for solving this problem, along with several applications that stem from choosing an appropriate set of conditioning events.

For example, we can efficiently produce predictions targeted at any polynomial number of decision makers, such that if they best respond to our predictions, each of them will have diminishing swap regret at the optimal rate. We then generalize this to the online combinatorial optimization problem, where the decision makers have large action spaces corresponding to structured subsets of a set of base actions: We give the first algorithms that can guarantee (to any polynomial number of decision makers) no regret to the best fixed action, not just overall, but on any polynomial number of *subsequences* that can depend on the actions chosen as well as any external context. We show how playing in an extensive-form game can be cast into this framework, and use these results to give efficient algorithms for obtaining no *subsequence regret* in extensive-form games. This gives a new family of regret guarantees that captures and generalizes previously studied notions such as regret to informed causal deviations, and is generally incomparable to other known families of efficiently obtainable guarantees.

We then turn to uncertainty quantification in machine learning, and consider the problem of producing *prediction sets* for online adversarial multiclass and multilabel classification. We show how to produce class scores that have *transparent coverage guarantees*: they can be used to produce prediction sets covering the true labels at the same rate as they would *had our scores been the true conditional class probabilities*. We then show that these transparent coverage guarantees imply strong online adversarial *conditional validity* guarantees (including *set-size conditional* coverage and *multigroup-fair* coverage) for (potentially *multiple*) downstream prediction set algorithms relying on our class scores. Moreover, we show how to guarantee that our class scores have improved L_2 loss (or cross-entropy loss, or more generally any separable Bregman loss) compared to any collection of benchmark models. This can be viewed as a high-dimensional, real-valued version of *omniprediction*. Compared to conformal prediction techniques, our uncertainty quantification framework gives increased flexibility and eliminates the need to choose a non-conformity score.

. An up-to-date version of this manuscript is available at https://arxiv.org/abs/2310.17651.

1. An Overview of Our Results

When is it a good idea for a decision maker to react to a predicted outcome as if the prediction is correct? Understanding the answer to this question has at least two important kinds of applications:

1. It gives us a natural *algorithm design* principle for sequential decision making under uncertainty, which we call *predict-then-act*. We first *predict* the payoffs for each of our available actions, taking care to produce predictions that are a "good idea" for us to follow, and then choose our action to optimize our payoff as if the prediction was correct.

2. It gives us a means to coordinate action amongst a wide variety of potentially unsophisticated agents, each with different objective functions: we produce predictions that are a "good idea" for each of them to follow, and then allow them to best-respond to our predictions as if they were correct. In this scenario we don't need the agents to have the sophistication to run a complicated decision making algorithm themselves — but we might hope that they obtain the same kinds of utility guarantees that they would if they were instead running a more sophisticated algorithm.

Calibration is a natural candidate answer to our question. Informally speaking, predictions are calibrated when they are unbiased even conditional on the value of the prediction: In other words, our predictions p for an unknown outcome y should satisfy (for all possible values v our prediction might take): $\mathbb{E}[y - v | p = v] = 0$. Amongst all policies mapping predictions to actions, the "best response" policy that acts as if the predictions are correct is payoff optimal if the predictions are calibrated. Moreover, it has been known since [33] that one can make sequential predictions that are guaranteed to be calibrated in hindsight, even if outcomes are chosen by an adaptive adversary.

But calibration has at least two serious shortcomings:

1. The statistical and computational complexity of producing forecasts for d-dimensional outcomes that are guaranteed to be calibrated grows exponentially with d. This is because the number of (discretized) values v that we might predict in d dimensions grows exponentially with d. For calibration, not only is our prediction space exponentially large, but we ask that our predictions be unbiased subject to exponentially many conditioning events. In fact, even in 1 dimension, it is known that achieving adversarial calibration at a rate of $O(\sqrt{T})$ is impossible [78] — even though it is possible to obtain swap regret at this rate [11].

2. Calibrated forecasts also need not be informative: even a constant predictor can be calibrated (so long as it predicts the mean outcome), and, in general, since calibration is a *marginal* guarantee, it can fail to hold after conditioning on various decision-relevant features x that are available at decision time.

When can we overcome these shortcomings?

Our Algorithmic Framework for Unbiased Prediction We study the problem of sequentially making predictions $p_t \in \mathbb{R}^d$ of d-dimensional adversarially selected vectors $y_t \in \mathbb{R}^d$, potentially as a function of available contexts x_t. We ask for predictions that are *unbiased* subject to some set of conditioning events $\mathcal{E}$, where each event $E(x_t, p_t) \in \mathcal{E}$ can be a function of both the context x_t and our own prediction p_t. In other words, we want that after T rounds of interaction:

$$\left\| \mathop{\mathbb{E}}_{t \in [T]} [(p_t - y_t) | E(x_t, p_t)] \right\|_\infty \leq \alpha(E) \quad \text{for all } E \in \mathcal{E}.$$

Here we say that such predictions have $\mathcal{E}$-bias bounded by α. The standard notion of calibration corresponds to the set of events defined as $E_v(p_t) = \mathbb{1}[p_t = v]$ for each v in some ϵ-net of the prediction space, when predictions are limited to being made from the same net. The size of an ϵ-net in d dimensions is exponential in d, which is a major difficulty with high-dimensional calibration.

Our first result is an algorithm for making $\mathcal{E}$-unbiased predictions for any collection of events $\mathcal{E}$, with bias $\alpha(E) = O\left(\frac{\ln(d|\mathcal{E}|T)}{\sqrt{T_E}}\right)$ and with per-round running time scaling polynomially with d, t, and $|\mathcal{E}|$ for each round $t \in [T]$. Here $T_E = \sum_{t=1}^{T} E(x_t, p_t)$ is the number of rounds for which event E was active. Thus the algorithm is computationally efficient whenever the set of events $\mathcal{E}$ is polynomially sized, and up to low order terms, achieves in an adversarial setting a bias rate that would be statistically optimal even in a stochastic setting. When the events are "disjoint" (i.e. at most one is active for any given prediction), we show how to improve the per-round running time to depend only polylogarithmically on t at each round $t \in [T]$.

We establish several basic connections between our ability to make unbiased predictions subject to events defined by the best-response correspondence of a downstream decision maker, and the regret of that downstream decision maker, whenever the utility function of the decision maker is linear in the state vector y_t that we are predicting. This captures decision makers who have *arbitrary* valuations over d discrete states when we predict a probability distribution over those states, as well as a number of other important cases. We then apply this result to obtain new results in online combinatorial optimization, learning in extensive form games, and uncertainty quantification using prediction sets.

Warm-up Application: Groupwise Swap Regret for Many Decision Makers To build intuition, we start by deriving algorithms for obtaining no swap regret in the experts problem with d actions. Informally, an agent has no swap regret if they have obtained utility that is as large as they would have had they played the best action in hindsight—not just overall, but also on every subsequence on which they selected any particular one of their actions. For a decision maker with utility function u, to guarantee that they will have no swap regret when they best respond to our predictions, it suffices that our predictions are unbiased with respect to the d disjoint events defined by their best response correspondence — i.e. the events in which they choose to play each of their d actions. Similar observations have been made before [52, 77, 91]. But now, paired with our prediction algorithm, we can efficiently make predictions that guarantee diminishing swap regret to *every* downstream decision maker with utility function in a polynomially sized set U, with bounds that are optimal up to a term growing logarithmically with $|U|$. We can also enlarge the set of events as a function of external context x_t to simultaneously give diminishing swap regret to each agent not just overall, but for arbitrary subsequences of actions that might e.g. correspond to demographic or other decision relevant groupings of the prediction space. This improves upon [10], who gave algorithms for obtaining diminishing *external* groupwise regret for a single decision maker.

Application to Uncertainty Quantification: Online Prediction Sets with Anytime Transparent Coverage A popular way of quantifying the uncertainty of machine learning predictions in multiclass classification problems is to produce prediction sets rather than point predictions. A prediction set is a set of labels that is intended to contain the true label with some target probability, say 95%. Given an example x, if we knew the true conditional probability $p(y|x)$ of each label y, we could produce the smallest possible prediction set subject to the coverage guarantee by sorting the labels in decreasing order of their likelihood, and including them in the prediction set until their cumulative probability exceeded 95%. More generally, we could optimize any other objective function to produce a prediction set, and read off the coverage rate of our set by summing the probabilities of the labels included within our prediction set. Unfortunately, the scores produced by machine learning models are *not* true conditional probabilities, and so this approach generally does not work. The approach taken by the conformal prediction literature (see e.g. [4, 82]) is to use a "non-conformity score" to reduce the high dimensional space of prediction sets to a 1-dimensional nested set of prediction sets, and solve a 1-D quantile estimation problem. Much of the art in making conformal prediction work well is choosing the "right" 1-dimensional scoring function.

We give a score-free method of producing prediction sets in arbitrary sequential prediction problems. Given a method for mapping individual label probabilities to prediction sets (such as e.g. finding the smallest set subject to a coverage constraint), we produce predicted probabilities for each label that are unbiased conditional on the event that the method includes the label within the prediction set. For any polynomial collection of such methods, our predicted probabilities can be used by each of these methods, and they will be guaranteed to satisfy the same coverage guarantees they would have had the probabilities been correct. In other words, the prediction sets have "transparent" coverage guarantees, in that we can estimate the coverage of each method by simply summing up the predicted scores for each label that is contained within the method's prediction sets. For example, this lets us give class probabilities that can be simultaneously used to produce prediction sets for many different coverage probabilities, and optimizing many different objectives (e.g. weighted prediction set sizes with different weights). By defining our events in terms of relevant context, we can also extend all of our guarantees to offer *groupwise* or *multivalid* coverage as in [7, 60]. Through another appropriate instantiation of our event collection, we also, for the first time, obtain online adversarial *set-size-conditional* ([3]) coverage guarantees.

Moreover, we can produce class scores with these transparent coverage guarantees that are simultaneously as accurate as any other prediction method at our disposal: we show how to produce scores satisfying a high-dimensional notion of *calibeating* [30]. What this means is given any polynomial collection of predictors $f(x)$ which map features to class probabilities, we produce predicted class probabilities that have smaller Brier score — or cross entropy — or more generally lower loss according to any *Bregman score* than any predictor in the class, while simultaneously being useful for producing prediction sets. This can be viewed as an online, high-dimensional, and real-valued extension of *omniprediction* [36, 41]. In addition to giving a more flexible collection of guarantees than conformal prediction methods, this eliminates the need to choose a non-conformity score.

Application: Subsequence Regret in Online Combinatorial Optimization and Extensive Form Games We next consider the online combinatorial optimization problem, in which a decision maker has a combinatorially large action space corresponding to subsets of d base actions, and has a utility function that is linear in the payoff of each of the base actions. A canonical special case of this setting is the online shortest paths problem, in which the base actions correspond to edges in a network, and an agent's action set corresponds to the set of $s \to t$ paths in the network (different agents might have different source and destination pairs). The action space of each player can be as large as 2^d. Given any polynomial collection of subsequence indicator functions (which can depend both on external context as well as the decisions made by the decision makers — e.g. the subsequence of days on which the chosen path includes toll roads and it is raining), we show how to efficiently make predictions such that the downstream decision maker has no regret to any of their $O(2^d)$ actions, not just overall, but also as restricted to any of the subsequences. Subsequence regret guarantees like this were previously known for settings with small action spaces [11] (i.e., via algorithms with running time that is polynomial in the number of actions) — we give the first such result for a large action space setting. Our result naturally extends to making predictions that give this guarantee not just to a single decision maker, but to every decision maker with utility function in any polynomially sized set U.

We then observe that this result applies to extensive form games as a special case. Informally speaking, an extensive form game is a sequential interaction played on a *game tree*. A player controls a subset of the nodes in the tree, and must decide on an action to take at each node; opponents or chance players decide on actions at other nodes, and play proceeds down the tree until it reaches a leaf node which corresponds to a payoff to each agent. Internal nodes of the game tree can be grouped together into "information sets" that are indistinguishable to the agent (which constrains the agent to choose the same action at every node within an information set). Because an agent must decide on an action to take at each node of the tree, the strategy space of the game is exponentially large in the size of the game tree. Nevertheless, the expected payoff that a strategy yields for a player can be expressed as the inner product of a vector representing the set of leaves that the agent's strategy makes reachable, and a vector corresponding to the payoff-weighted vector of probabilities that the agent's opponents' make each leaf reachable. This means it can be expressed as an online combinatorial optimization problem of dimension equal to the number of leaves in the game tree (exponentially smaller than the number of actions). Our methods therefore give algorithms for obtaining subsequence regret in extensive form games for arbitrary polynomial collections of subsequences.

Our method does *not* require that the information sets in the extensive-form game satisfy the "perfect recall" assumption as many prior methods do, and is an efficient reduction to the "best response" problem in extensive form games. That is, whenever it is possible to efficiently compute an extensive form strategy that best responds to fixed opponent strategies, our algorithms are efficient. In some cases, the running time of our algorithms can be improved to depend on the number of information sets rather than the number of game tree leaves. We show that subsequence regret generalizes the existing notion of regret to informed causal deviations [23, 45] — which means it can be used to achieve convergence to notions of extensive form correlated equilibrium; subsequence regret is incomparable to other known families of regret guarantees in extensive form games [26, 27, 74].

1.1. Roadmap

Due to space constraints, we relegate the rest of our discussion to the appendix. Here is a summary of where our various results can be found:

- In Section A, we discuss further related work not covered in this introduction, as well as exciting future directions that we hope this work could be useful for.
- In Section B, we discuss the fundamentals of our transparent decision-making pipeline; in particular, we formally define the online learning setting, our model of a downstream decision maker, and several notions of regret for downstream decisions.
- In Section C, we introduce our generic algorithm for making high-dimensional unbiased predictions, analyze it, and give its performance guarantees.
- In Section D, we prove a basic guarantee for downstream decision makers: that best-responding to our (appropriately unbiased) predictions leads to no swap regret and no type regret.

- In Section E, we rigorously develop our uncertainty quantification framework for prediction sets with transparent coverage (Section E.2) and prove our best-in-class result with respect to all sufficiently smooth separable Bregman losses (Section E.3).
- In Section F, we develop further applications: No group swap regret (Section F.1), as well as no subsequence regret guarantees in online combinatorial optimization (where there are exponentially many possible decisions for the decision maker to choose from) and in extensive-form games (Section F.2).
- In the remaining Sections, all omitted proofs are provided, along with further discussion where appropriate.

References

[1] Jacob D. Abernethy and Rafael M. Frongillo. A characterization of scoring rules for linear properties. In Shie Mannor, Nathan Srebro, and Robert C. Williamson, editors, *Proceedings of the 25th Annual Conference on Learning Theory*, volume 23 of *Proceedings of Machine Learning Research*, pages 27.1–27.13, Edinburgh, Scotland, 25–27 Jun 2012. PMLR. URL `https://proceedings.mlr.press/v23/abernethy12.html`.

[2] Krishna Acharya, Eshwar Ram Arunachaleswaran, Sampath Kannan, Aaron Roth, and Juba Ziani. Oracle efficient algorithms for groupwise regret. *arXiv preprint arXiv:2310.04652*, 2023.

[3] Anastasios Angelopoulos, Stephen Bates, Jitendra Malik, and Michael I Jordan. Uncertainty sets for image classifiers using conformal prediction. *arXiv preprint arXiv:2009.14193*, 2020.

[4] Anastasios N Angelopoulos and Stephen Bates. A gentle introduction to conformal prediction and distribution-free uncertainty quantification. *arXiv preprint arXiv:2107.07511*, 2021.

[5] Arindam Banerjee, Xin Guo, and Hui Wang. On the optimality of conditional expectation as a bregman predictor. *IEEE Transactions on Information Theory*, 51(7):2664–2669, 2005.

[6] Arindam Banerjee, Srujana Merugu, Inderjit S Dhillon, Joydeep Ghosh, and John Lafferty. Clustering with bregman divergences. *Journal of machine learning research*, 6(10), 2005.

[7] Osbert Bastani, Varun Gupta, Christopher Jung, Georgy Noarov, Ramya Ramalingam, and Aaron Roth. Practical adversarial multivalid conformal prediction. In *Advances in Neural Information Processing Systems*, 2022.

[8] Michael Bian and Rina Foygel Barber. Training-conditional coverage for distribution-free predictive inference. *Electronic Journal of Statistics*, 17(2):2044–2066, 2023.

[9] David Blackwell. An analog of the minimax theorem for vector payoffs. *Pacific Journal of Mathematics*, 6(1):1 – 8, 1956.

[10] Avrim Blum and Thodoris Lykouris. Advancing subgroup fairness via sleeping experts. In *11th Innovations in Theoretical Computer Science Conference (ITCS 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

[11] Avrim Blum and Yishay Mansour. From external to internal regret. *Journal of Machine Learning Research*, 8(6), 2007.

[12] Michael Bowling, Neil Burch, Michael Johanson, and Oskari Tammelin. Heads-up limit hold'em poker is solved. *Science*, 347(6218):145–149, 2015.

[13] Noam Brown and Tuomas Sandholm. Superhuman ai for multiplayer poker. *Science*, 365 (6456):885–890, 2019.

[14] Modibo K Camara, Jason D Hartline, and Aleck Johnsen. Mechanisms for a no-regret agent: Beyond the common prior. In *2020 ieee 61st annual symposium on foundations of computer science (focs)*, pages 259–270. IEEE, 2020.

[15] Maxime Cauchois, Suyash Gupta, and John C Duchi. Knowing what you know: valid and validated confidence sets in multiclass and multilabel prediction. *The Journal of Machine Learning Research*, 22(1):3681–3722, 2021.

[16] Liyu Chen, Haipeng Luo, and Chen-Yu Wei. Impossible tuning made possible: A new expert algorithm and its applications. In *Conference on Learning Theory*, pages 1216–1259. PMLR, 2021.

[17] Natalie Collina, Aaron Roth, and Han Shao. Efficient prior-free mechanisms for no-regret agents. *Manuscript*, 2023.

[18] Laurent Condat. Fast projection onto the simplex and the l 1 ball. *Mathematical Programming*, 158(1-2):575–585, 2016.

[19] A Philip Dawid. Calibration-based empirical probability. *The Annals of Statistics*, 13(4): 1251–1274, 1985.

[20] Emir Demirović, Peter J Stuckey, James Bailey, Jeffrey Chan, Chris Leckie, Kotagiri Ramamohanarao, and Tias Guns. An investigation into prediction+ optimisation for the knapsack problem. In *Integration of Constraint Programming, Artificial Intelligence, and Operations Research: 16th International Conference, CPAIOR 2019, Thessaloniki, Greece, June 4–7, 2019, Proceedings 16*, pages 241–257. Springer, 2019.

[21] Zhun Deng, Cynthia Dwork, and Linjun Zhang. Happymap: A generalized multi-calibration method. *arXiv preprint arXiv:2303.04379*, 2023.

[22] Priya Donti, Brandon Amos, and J Zico Kolter. Task-based end-to-end model learning in stochastic optimization. *Advances in neural information processing systems*, 30, 2017.

[23] Miroslav Dudík and Geoffrey Gordon. A sampling-based approach to computing equilibria in succinct extensive-form games. *arXiv preprint arXiv:1205.2649*, 2012.

[24] Othman El Balghiti, Adam N Elmachtoub, Paul Grigas, and Ambuj Tewari. Generalization bounds in the predict-then-optimize framework. *Advances in neural information processing systems*, 32, 2019.

[25] Adam N Elmachtoub and Paul Grigas. Smart "predict, then optimize". *Management Science*, 68(1):9–26, 2022.

[26] Gabriele Farina and Charilaos Pipis. Polynomial-time linear-swap regret minimization in imperfect-information sequential games. *arXiv preprint arXiv:2307.05448*, 2023.

[27] Gabriele Farina, Andrea Celli, Alberto Marchesi, and Nicola Gatti. Simple uncoupled no-regret learning dynamics for extensive-form correlated equilibrium. *Journal of the ACM*, 69 (6):1–41, 2022.

[28] Shai Feldman, Stephen Bates, and Yaniv Romano. Improving conditional coverage via orthogonal quantile regression. *Advances in neural information processing systems*, 34:2060–2071, 2021.

[29] Dean P Foster and Sergiu Hart. Smooth calibration, leaky forecasts, finite recall, and nash dynamics. *Games and Economic Behavior*, 109:271–293, 2018.

[30] Dean P Foster and Sergiu Hart. " calibeating": beating forecasters at their own game. *arXiv preprint arXiv:2209.04892*, 2022.

[31] Dean P Foster and Sham M Kakade. Calibration via regression. In *2006 IEEE Information Theory Workshop-ITW'06 Punta del Este*, pages 82–86. IEEE, 2006.

[32] Dean P Foster and Rakesh Vohra. Regret in the on-line decision problem. *Games and Economic Behavior*, 29(1-2):7–35, 1999.

[33] Dean P Foster and Rakesh V Vohra. Asymptotic calibration. *Biometrika*, 85(2):379–390, 1998.

[34] Dean P Foster, Alexander Rakhlin, Karthik Sridharan, and Ambuj Tewari. Complexity-based approach to calibration with checking rules. In *Proceedings of the 24th Annual Conference on Learning Theory*, pages 293–314. JMLR Workshop and Conference Proceedings, 2011.

[35] Rina Foygel Barber, Emmanuel J Candes, Aaditya Ramdas, and Ryan J Tibshirani. The limits of distribution-free conditional predictive inference. *Information and Inference: A Journal of the IMA*, 10(2):455–482, 2021.

[36] Sumegha Garg, Christopher Jung, Omer Reingold, and Aaron Roth. Oracle efficient online multicalibration and omniprediction. In *ACM-SIAM Symposium on Discrete Algorithms*, 2024.

[37] Isaac Gibbs and Emmanuel Candes. Adaptive conformal inference under distribution shift. *Advances in Neural Information Processing Systems*, 34:1660–1672, 2021.

[38] Isaac Gibbs and Emmanuel Candès. Conformal inference for online prediction with arbitrary distribution shifts. *arXiv preprint arXiv:2208.08401*, 2022.

[39] Isaac Gibbs, John J Cherian, and Emmanuel J Candès. Conformal prediction with conditional guarantees. *arXiv preprint arXiv:2305.12616*, 2023.

[40] Ira Globus-Harris, Declan Harrison, Michael Kearns, Aaron Roth, and Jessica Sorrell. Multicalibration as boosting for regression. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume

202 of *Proceedings of Machine Learning Research*, pages 11459–11492. PMLR, 2023. URL https://proceedings.mlr.press/v202/globus-harris23a.html.

[41] Parikshit Gopalan, Adam Tauman Kalai, Omer Reingold, Vatsal Sharan, and Udi Wieder. Omnipredictors. In Mark Braverman, editor, *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 - February 3, 2022, Berkeley, CA, USA*, volume 215 of *LIPIcs*, pages 79:1–79:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi: 10.4230/LIPIcs.ITCS.2022.79. URL https://doi.org/10.4230/LIPIcs.ITCS.2022.79.

[42] Parikshit Gopalan, Michael P Kim, Mihir A Singhal, and Shengjia Zhao. Low-degree multicalibration. In *Conference on Learning Theory*, pages 3193–3234. PMLR, 2022.

[43] Parikshit Gopalan, Lunjia Hu, Michael P Kim, Omer Reingold, and Udi Wieder. Loss minimization through the lens of outcome indistinguishability. In *14th Innovations in Theoretical Computer Science Conference (ITCS 2023)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2023.

[44] Parikshit Gopalan, Michael P Kim, and Omer Reingold. Characterizing notions of omniprediction via multicalibration. *arXiv preprint arXiv:2302.06726*, 2023.

[45] Geoffrey J Gordon, Amy Greenwald, and Casey Marks. No-regret learning in convex games. In *Proceedings of the 25th international conference on Machine learning*, pages 360–367, 2008.

[46] Amy Greenwald and Amir Jafari. A general class of no-regret learning algorithms and game-theoretic equilibria. In *Learning theory and kernel machines*, pages 2–12. Springer, 2003.

[47] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer, 1988. ISBN 978-3-642-97883-8. doi: 10.1007/978-3-642-97881-4.

[48] Peter D. Grünwald and A. Philip Dawid. Game theory, maximum entropy, minimum discrepancy and robust Bayesian decision theory. *The Annals of Statistics*, 32(4):1367 – 1433, 2004. doi: 10.1214/009053604000000553. URL https://doi.org/10.1214/009053604000000553.

[49] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR, 2017.

[50] Varun Gupta, Christopher Jung, Georgy Noarov, Mallesh M Pai, and Aaron Roth. Online multivalid learning: Means, moments, and prediction intervals. In *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2022.

[51] Nika Haghtalab, Michael I Jordan, and Eric Zhao. A unifying perspective on multi-calibration: Unleashing game dynamics for multi-objective learning. *arXiv preprint arXiv:2302.10863*, 2023.

[52] Nika Haghtalab, Chara Podimata, and Kunhe Yang. Calibrated stackelberg games: Learning optimal commitments against calibrated agents. *arXiv preprint arXiv:2306.02704*, 2023.

[53] James Hannan. Approximation to bayes risk in repeated play. *Contributions to the Theory of Games*, 3:97–139, 1957.

[54] Elad Hazan. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016.

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

[56] Ursula Hébert-Johnson, Michael Kim, Omer Reingold, and Guy Rothblum. Multicalibration: Calibration for the (computationally-identifiable) masses. In *International Conference on Machine Learning*, pages 1939–1948. PMLR, 2018.

[57] Shinji Ito. A tight lower bound and efficient reduction for swap regret. *Advances in Neural Information Processing Systems*, 33:18550–18559, 2020.

[58] Rafael Izbicki, Gilson T Shimizu, and Rafael B Stern. Flexible distribution-free conditional predictive bands using density estimators. *arXiv preprint arXiv:1910.05575*, 2019.

[59] Christopher Jung, Changhwa Lee, Mallesh Pai, Aaron Roth, and Rakesh Vohra. Moment multicalibration for uncertainty estimation. In *Conference on Learning Theory*, pages 2634–2678. PMLR, 2021.

[60] Christopher Jung, Georgy Noarov, Ramya Ramalingam, and Aaron Roth. Batch multivalid conformal prediction. In *International Conference on Learning Representations (ICLR)*, 2023.

[61] Sham M Kakade and Dean P Foster. Deterministic calibration and nash equilibrium. *Journal of Computer and System Sciences*, 74(1):115–130, 2008.

[62] Adam Kalai and Santosh Vempala. Efficient algorithms for online decision problems. *Journal of Computer and System Sciences*, 71(3):291–307, 2005.

[63] Michael Kearns, Seth Neel, Aaron Roth, and Zhiwei Steven Wu. Preventing fairness gerrymandering: Auditing and learning for subgroup fairness. In *International conference on machine learning*, pages 2564–2572. PMLR, 2018.

[64] Michael Kearns, Seth Neel, Aaron Roth, and Zhiwei Steven Wu. An empirical study of rich subgroup fairness for machine learning. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 100–109, 2019.

[65] Elias Khalil, Hanjun Dai, Yuyu Zhang, Bistra Dilkina, and Le Song. Learning combinatorial optimization algorithms over graphs. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/d9896106ca98d3d05b8cbdf4fd8b13a1-Paper.pdf.

[66] Michael P Kim, Amirata Ghorbani, and James Zou. Multiaccuracy: Black-box post-processing for fairness in classification. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pages 247–254, 2019.

[67] Robert Kleinberg, Renato Paes Leme, Jon Schneider, and Yifeng Teng. U-calibration: Forecasting for an unknown agent. *arXiv preprint arXiv:2307.00168*, 2023.

[68] Meelis Kull and Peter Flach. Novel decompositions of proper scoring rules for classification: Score adjustment as precursor to calibration. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2015, Porto, Portugal, September 7-11, 2015, Proceedings, Part I 15*, pages 68–85. Springer, 2015.

[69] Daniel Lee, Georgy Noarov, Mallesh Pai, and Aaron Roth. Online minimax multiobjective optimization: Multicalibeating and other applications. *Advances in Neural Information Processing Systems*, 35:29051–29063, 2022.

[70] Ehud Lehrer. A wide range no-regret theorem. *Games and Economic Behavior*, 42(1):101–115, 2003.

[71] Heyuan Liu and Paul Grigas. Risk bounds and calibration for a smart predict-then-optimize method. *Advances in Neural Information Processing Systems*, 34:22083–22094, 2021.

[72] Jayanta Mandi, Peter J Stuckey, Tias Guns, et al. Smart predict-and-optimize for hard combinatorial optimization problems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34 (02), pages 1603–1610, 2020.

[73] Mehryar Mohri and Scott Yang. Online learning with transductive regret. *Advances in Neural Information Processing Systems*, 30, 2017.

[74] Dustin Morrill, Ryan D'Orazio, Marc Lanctot, James R Wright, Michael Bowling, and Amy R Greenwald. Efficient deviation types and learning for hindsight rationality in extensive-form games. In *International Conference on Machine Learning*, pages 7818–7828. PMLR, 2021.

[75] Dustin Morrill, Ryan D'Orazio, Reca Sarfati, Marc Lanctot, James R Wright, Amy R Greenwald, and Michael Bowling. Hindsight and sequential rationality of correlated play. In *AAAI*, volume 35, pages 5584–5594, May 2021. doi: 10.1609/aaai.v35i6.16702. URL https://ojs.aaai.org/index.php/AAAI/article/view/16702.

[76] Georgy Noarov and Aaron Roth. The statistical scope of multicalibration. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 26283–26310. PMLR, 2023. URL https://proceedings.mlr.press/v202/noarov23a.html.

[77] Vianney Perchet. Internal regret with partial monitoring: Calibration-based optimal algorithms. *Journal of Machine Learning Research*, 12(6), 2011.

[78] Mingda Qiao and Gregory Valiant. Stronger calibration lower bounds via sidestepping. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 456–466, 2021.

[79] Yaniv Romano, Rina Foygel Barber, Chiara Sabatti, and Emmanuel Candès. With malice toward none: Assessing uncertainty via equalized coverage. *Harvard Data Science Review*, 2 (2):4, 2020.

[80] Aaron Roth. Uncertain: Modern topics in uncertainty estimation. https://www.cis.upenn.edu/ aaroth/uncertainty-notes.pdf, 2022.

[81] Mauricio Sadinle, Jing Lei, and Larry Wasserman. Least ambiguous set-valued classifiers with bounded error levels. *Journal of the American Statistical Association*, 114(525):223–234, 2019.

[82] Glenn Shafer and Vladimir Vovk. A tutorial on conformal prediction. *Journal of Machine Learning Research*, 9(3), 2008.

[83] Maurice Sion. On general minimax theorems. *Pacific Journal of Mathematics*, 8(1):171 – 176, 1958.

[84] Eiji Takimoto and Manfred K Warmuth. Path kernels and multiplicative updates. *The Journal of Machine Learning Research*, 4:773–818, 2003.

[85] Toon Vanderschueren, Tim Verdonck, Bart Baesens, and Wouter Verbeke. Predict-then-optimize or predict-and-optimize? an empirical evaluation of cost-sensitive learning strategies. *Information Sciences*, 594:400–415, 2022.

[86] Bernhard Von Stengel and Françoise Forges. Extensive-form correlated equilibrium: Definition and computational complexity. *Mathematics of Operations Research*, 33(4):1002–1022, 2008.

[87] Vladimir Vovk. Conditional validity of inductive conformal predictors. In *Asian conference on machine learning*, pages 475–490. PMLR, 2012.

[88] Vladimir Vovk, Valentina Fedorova, Ilia Nouretdinov, and Alexander Gammerman. Criteria of efficiency for conformal prediction. In *Conformal and Probabilistic Prediction with Applications: 5th International Symposium, COPA 2016, Madrid, Spain, April 20-22, 2016, Proceedings 5*, pages 23–39. Springer, 2016.

[89] Bryan Wilder, Bistra Dilkina, and Milind Tambe. Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33 (01), pages 1658–1665, 2019.

[90] Margaux Zaffran, Olivier Féron, Yannig Goude, Julie Josse, and Aymeric Dieuleveut. Adaptive conformal predictions for time series. In *International Conference on Machine Learning*, pages 25834–25866. PMLR, 2022.

[91] Shengjia Zhao, Michael Kim, Roshni Sahoo, Tengyu Ma, and Stefano Ermon. Calibrating predictions to decisions: A novel approach to multi-class calibration. *Advances in Neural Information Processing Systems*, 34:22313–22324, 2021.

[92] Martin Zinkevich, Michael Johanson, Michael Bowling, and Carmelo Piccione. Regret minimization in games with incomplete information. *Advances in neural information processing systems*, 20, 2007.

Appendix A. Further Future Applications and Related Work

More Applications We expect that our framework will find many other applications. We here mention a few. First, there are some basic applications that take advantage of the ability of our framework to make predictions within an arbitrary convex feasible region. For example, we can make probability forecasts in d outcome settings that satisfy marginal (or top label) calibration *subject to the constraint that the predicted probabilities sum to 1*. Similarly we can produce predicted CDFs for a real valued outcome satisfying marginal quantile calibration [50] at each quantile level, subject to the constraint that our predicted quantile values are monotone. Although these applications are extremely simple, we are not aware of ways to easily obtain these guarantees with prior work.

We also briefly mention an application of our techniques that is explored in concurrent work of [17], who apply our algorithms in a repeated principle agent setting defined by [14]. Briefly, [14] gave a mechanism that replaced the standard "common prior" assumptions that underlie principal agent models with calibrated forecasts of an underlying state, and is applicable in adversarial settings. [14] use the traditional notion of calibration, and as a result inherit exponential computational and statistical dependencies on the cardinality of the state space. [17] show how to apply our techniques to recover the same results (under weaker assumptions) with an exponentially improved dependence on the cardinality of the state space.

A.1. Related Work

Calibration, multicalibration and downstream optimization The study of sequential calibration goes back to [19] who viewed it as a way to define the foundations of probability, and algorithms for producing calibrated forecasts in an adversarial setting were first given by [33]. [32] were the first to connect sequential calibration to sequential decision making, showing that a decision maker who best responds to (fully) calibrated forecasts obtains diminishing internal regret (and that when all agents in a game do so, empirical play converges to correlated equilibrium). [61] and [29] make a similar connection between "smooth calibration" (which in contrast to classical calibration can be obtained with deterministic algorithms) and Nash equilibrium.

In the recent computer science literature, there has been interest in constructive calibration guarantees (obtained by efficient algorithms and obtaining good rates) that hold conditional on context in various ways, called *multi-calibration* [56]. Multicalibration has been studied both in the batch setting [40, 51, 56, 66] and in the online sequential setting [31, 34, 36, 50]. For the most part (with a few notable exceptions [42, 91]) multicalibration has been studied in the 1-dimensional setting in which the outcome being predicted is boolean. This has been extended to predicting real valued outcomes, with notions of calibration tailored to variances [59], quantiles [7, 60], and other distributional properties [76]. See [80] for an introductory exposition of this literature.

There is a line of work that aims to use (multi)calibration as a tool for a one-dimensional form of downstream decision making, called omniprediction. The goal of omniprediction, introduced in [41], is to make probabilistic predictions of a binary outcome as a function of contextual information that are useful for optimizing a variety of downstream loss functions. [41] show that a predictor that is multicalibrated with respect to a benchmark class of functions $\mathcal{H}$ and a binary label can be used to optimize any convex, Lipschitz loss function of an action and a binary label (see [43] for a related set of results). These results are in the batch setting. In the online setting, [67] defined "U-calibration", which can be viewed as a non-contextual version of omniprediction, in which the goal is to make predictions that guarantee an arbitrary downstream decision maker no external-regret. Since there is no context in this setting, the benchmark class to which regret is measured is the set of constant functions. In comparison to [67], our goal is to give both stronger guarantees than external regret, and to be able to do so even when the state space is very large. What we pay for these stronger guarantees is a logarithmic dependence on the number of downstream utility functions our guarantees hold for ([67] give algorithms that guarantee no external regret for *any* downstream utility function). Using a connection between multicalibration and swap-regret established by [40] and [44], [36] give *oracle efficient* algorithms for online multicalibration, with applications to online omniprediction — i.e. algorithms that are efficient reductions to the problem of online learning over the benchmark class of functions $\mathcal{H}$. Like other work in omniprediction, this result is limited to the 1-dimensional binary setting.

The most closely related work is [91], who define and study "decision calibration" in the batch setting in the context of predicting a probability distribution over k discrete outcomes. Decision calibration is a slightly weaker requirement than what we study, also defined in terms of the best-response correspondence of a decision maker's utility function. Decision calibration asks, informally, that a decision maker be able to correctly estimate the expected reward of their best response policy; we ask for a slightly stronger condition that requires them to also be able to estimate the utility of deviations as a function of their play. This kind of unbiased estimation (based on the best response correspondence of a decision maker) has also been previously observed to be related to swap regret in [77] and [52]. The algorithmic portion of our work can be viewed as extending [91] from the batch to the online adversarial setting; Our applications hinge crucially on both the online aspect of our algorithm and on the more general setting we consider, beyond predicting distributions on k outcomes.

An expansive recent literature has focused on the similarly named *predict-then-optimize* problem [24, 25, 71]. This line of work investigates a setup in which predictions made from data are to be used in a linear optimization problem downstream in the pipeline. This is similar in motivation to our 'predict-then-act' framework, but with two important differences: (1) the predict-then-optimize framework aims to optimize for a single downstream problem, whereas our framework aims to simultaneously provide guarantees to an arbitrary finite collection of downstream decision makers; and (2) the surrogate loss approach studied in this literature is naturally embedded in a batch/distributional setting, where the goal is to exactly optimize for the Bayes optimal downstream decision policy, up to generalization/risk bounds; meanwhile, our framework naturally lives in the online adversarial setting, and aims for different notions of optimality defined in terms of regret bounds, as well as omniprediction-type 'best-in-class' optimality. Both frameworks can be used to solve downstream combinatorial optimization problems [20, 72]; but our framework appears to have a broader set of applications — as a consequence of its strong calibration properties, we are able to apply our framework to derive strong uncertainty quantification guarantees, which do not appear to naturally fit within the predict-then-optimize framework. There also exist other approaches for learning in batch decision making pipelines, that are different from the predict-then-optimize method; see e.g. [22, 65, 85, 89].

No-regret guarantees in online learning There are also long lines of work in online learning related to our applications; here we survey the most relevant. No-regret learning, which requires that a decision maker obtain cumulative loss at most that of their best single action in hindsight against an adversarial sequence of losses, has been studied at least since [53] — see [54] for a modern treatment of this literature. We highlight [62] (which we will make use of) who give efficient no regret algorithms in online linear and combinatorial optimization problems, which are large-action-space settings in which the cost of each action has linear structure. Internal regret, which corresponds to regret on the subsequences defined by the play of each action, was first defined by [32], who also showed it could be obtained by best responding to calibrated forecasts. [70] defined a notion of "wide-range regret" which is equivalent to subsequence regret: that a player should have no regret not just overall on the whole sequence, but also on subsequences that can be defined both as a function of time ("time selection functions") and as a function of the actions of the learner. [11] gave algorithms for obtaining this kind of subsequence regret (including, notably internal (or "swap") regret as a special case). The algorithm of [11] is efficient when the action space is polynomially sized: it requires computing eigenvectors of a square matrix of dimension equal to the number of actions in the game. Motivated by fairness concerns, [10] give an algorithm for obtaining diminishing "groupwise" regret, which is equivalent to regret with respect to a collection of time selection functions; very recently (and concurrently with this paper), [2] show how to modify the algorithm of [10] to make it "oracle efficient" — to reduce it to the problem of obtaining external regret with overhead polynomial in the number of time selection functions. These results do not accommodate subsequences that can depend on the actions of the learner, which are crucial for our applications. We give the first efficient algorithms for getting subsequence regret, for an arbitrary polynomial number of subsequences, in online combinatorial optimization settings, by operating over the (polynomially sized) linear representation space for the costs rather than the (exponentially sized) action space.

The problem of efficiently obtaining no-regret in extensive form games was first studied by [92], who gave algorithms for obtaining no (external) regret, which is sufficient for convergence to Nash equilibrium in zero-sum games in self-play. [23, 45] give algorithms for obtaining no-regret to *causal deviations*, which is sufficient for convergence to a notion of correlated equilibrium in extensive form games [86]. More recently, there has been renewed theoretical interest in regret-minimization in extensive form games: both [74] and [26] have defined incomparable classes of no-regret guarantees. Our notion of subsequence regret in extensive form games generalizes regret to causal deviations—which can be obtained by asking for no-regret on those subsequences in which the player makes each particular internal node in the game tree reachable—and is incomparable to the classes defined in both [74] and [26].

Uncertainty quantification The goal of conformal prediction is to endow black box predictors with the ability to produce "prediction sets" — predictions corresponding to sets of labels — that have the property that they contain the true label with a desired coverage probability. Conformal prediction uses a one-dimensional "non-conformity score" to reduce the problem to a 1-dimensional quantile estimation problem—see [82] and [4] for approachable introductions to the topic. A primary point of departure for our work is that we dispense with the non-conformity score, and deal directly with the high-dimensional prediction set problem.

Our method, while different from conformal prediction, provides online adversarial coverage guarantees, and thus joins a very recent collection of works on adversarial conformal inference methods [7, 37, 38, 50, 90].

While vanilla conformal prediction approaches give marginal coverage guarantees (i.e., those that hold on average over the entire data set), significant amounts of recent work in conformal prediction has focused on obtaining *conditional* guarantees of various sorts, which hold even over smaller, relevant portions of the data. [87] was perhaps the first to study conditional coverage problems. While it was shown by [35] that obtaining full conditional coverage in a distribution free regression setting is impossible, various special types of conditional coverage have been studied in the batch conformal setting, e.g. [81], [79], [58], [28], [15], [8]. Recently, multicalibration techniques (aimed at calibrating to either the variance or the quantiles of the non-conformity score) have been used to give group-conditional and threshold conditional guarantees in both the batch and adversarial settings [7, 21, 39, 50, 59, 60]—all of this still applied in the presence of a chosen non-conformity score, which makes the problem 1-dimensional.

An appealing feature of our method is that it predicts probability scores that "look like the true conditional probabilities $p(y|x)$ for the purposes of producing prediction sets". The fact that deep neural networks often produce miscalibrated class scores was observed by [49] and generated a large literature that is too expansive to fully survey here. [81] showed via a reduction to the Neyman-Pearson lemma that if scores exactly coincided with the true conditional probabilities, they could be used to produce *set-size optimal* prediction sets. Other relations between true conditional probabilities and optimality of various prediction set efficiency criteria were studied by [88]. Rather than aiming for optimality in an absolute sense, we demonstrate that our multiclass probability predictions satisfy a strong best-in-class property, which is new to the uncertainty quantification literature, but has been studied in several recent works on calibration. [30] introduces the problem of "calibeating": making (one-dimensional) calibrated forecasts in an adversarial setting that have lower Brier score than any fixed and given benchmark model. [69] give improved bounds for simultaneously calibeating many models. We show how to produce multiclass models that give transparent coverage guarantees while simultaneously satisfying a high-dimensional version of calibeating; obtaining lower Brier score (or indeed any Bregman score) than any one of a collection of given benchmark models. This can also be viewed as a high-dimensional and real-valued generalization of omniprediction [41].

Appendix B. Preliminaries

This section introduces the central object of our study, the sequential (online adversarial) pipeline:

$$\text{Data} \to \text{Predictions} \to \text{Decisions}.$$

In Section B.1, we define the prediction task as well as the desired event-conditional unbiasedness guarantee for the predictions. In Section B.2, we formally introduce the utility-based model for downstream decision makers, as well as several types of *regret*, i.e., metrics with which to measure the decision maker's success.

B.1. Predictions

We are faced with a sequential high dimensional prediction setting, defined by an arbitrary *context space* $\mathcal{X}$ and a convex compact prediction space $\mathcal{C} \subseteq \mathbb{R}^d$ for some finite dimension d. Without loss of generality (up to scaling our bounds by a multiplicative constant), we assume that $2\max_{y\in\mathcal{C}}||y||_\infty \leq 1$.

In rounds $t \in \{1, 2, \ldots\}$, a *learner*, or *predictor*, interacts with an *adversary* as follows:

1. The learner (may) observe a context $x_t \in \mathcal{X}$;
2. The learner produces a distribution over predictions $\psi_t \in \Delta\mathcal{C}$, from which a *prediction* $p_t \in \mathcal{C}$ is sampled;
3. The adversary produces an outcome $y_t \in \mathcal{C}$.

Let $\Pi = \{(x, p, y) \in \mathcal{X} \times \mathcal{C} \times \mathcal{C}\}$ denote the set of possible realized triples at each round. An interaction over T rounds produces a transcript $\pi_T \in \Pi^T$. We write $\pi_T^{<t}$ as the prefix of the first $t-1$ triples in π_T, for any $t \leq T$. We write $\Pi^* = \bigcup_{T=1}^{\infty} \Pi^T$ for the space of all transcripts. A learner is a collection of randomized algorithms (one for each t) mapping a transcript of length $t-1$ and a context to a distribution over predictions at round t: $\mathrm{Learner}_t : \Pi^{t-1} \times \mathcal{X} \to \Delta\mathcal{C}$. An adversary is a collection of randomized algorithms, each mapping a transcript of length $t-1$ to a context and a distribution over realizations: $\mathrm{Adv}_t : \Pi^{t-1} \to \mathcal{X} \times \Delta\mathcal{C}$. This models an adaptive adversary who can make decisions as an arbitrary function of the past history, but must be independent of the learner's randomness at round t. A learner paired with an adversary implicitly define a distribution over transcripts.

An intermediate goal (generally in service of downstream decision making, as defined in Section B.2) will be that in hindsight (i.e., in expectation over the empirical distribution of the transcript π^T that ends up being realized after T rounds) our predictions are *unbiased*, not just overall, but also conditional on various *events*. We now formally define the notion of events and our unbiasedness objective.

Definition 1 (Events) *An* event *E is an arbitrary mapping from transcripts, contexts and predictions to $[0,1]$: $E : \Pi^* \times \mathcal{X} \times \mathcal{C} \to [0,1]$.*

If the range of E is $\{0,1\}$, then we say that E is a binary *event. We say that a collection of binary events $\mathcal{E}$ is* disjoint *if for every $(\pi, x, p) \in \Pi^* \times \mathcal{X} \times \mathcal{C}$: $\sum_{E\in\mathcal{E}} E(\pi, x, p) \leq 1$.*

We will elide arguments to E that are not used; for example, if an event is independent of the transcript, we will write $E(x_t, p_t)$ rather than $E(\pi_{t-1}, x_t, p_t)$, and if the event is also independent of context, then we will simply write $E(p_t)$.

Definition 2 (Event Frequency) *Fixing a transcript $\pi_T = \{(x_t, p_t, y_t)\}_{t=1}^T$, the* frequency *of event E with respect to π_T at time t is given by:*

$$n_t(E, \pi_T) = \sum_{\tau=1}^{t} E(\pi_T^{<\tau}, x_\tau, p_\tau)$$

Observe that for any collection of disjoint events $\mathcal{E}$ and any round t, one has $\sum_{E\in\mathcal{E}} n_t(\pi_T^{<\tau}, E, \pi_T) \leq t$.

Definition 3 (Event-conditional unbiasedness) *Fix a collection $\mathcal{E}$ of events and a function $\alpha : \mathbb{R} \to \mathbb{R}$. A transcript $\pi_T = \{(x_t, p_t, y_t)\}_{t=1}^T$ is α-unbiased with respect to $\mathcal{E}$ if for every $E \in \mathcal{E}$ and every coordinate $i \in [d]$:*

$$\left| \sum_{t=1}^{T} (p_{t,i} - y_{t,i}) \cdot E(x_t, p_t) \right| \leq \alpha(n_t(E, \pi_T)).$$

Remark 4 *Two remarks about Definition 3 are in order. First, it asks for our prediction vectors to be unbiased (on average over time) separately across their coordinates. However, we allow the conditioning events E to depend on* the entire vector p_t. *This is where our approach derives its power, compared to asking for multi-calibration guarantees separately in every coordinate (such guarantees can be obtained by running e.g. the algorithms of [50] independently for each of the d coordinates).*

Second, we let α be a function *of $n_t(E, \pi_T)$. This allows us to give finer-grained guarantees that will scale with $\alpha(n_t(E, \pi_T)) \approx \sqrt{n_t(E, \pi_T)}$, as opposed to the coarser style bounds of $\alpha \approx \sqrt{T}$ used in most prior work[1] [50, 69]. One notable exception is [51], who give similarly refined bounds in the batch setting by reducing to no-regret learning algorithms with 2nd order regret bounds — we get our improvement in the same way (by reducing to a different no regret learning algorithm, but one that also has 2nd order regret bounds).*

B.2. Decisions and Regret

We are interested in making predictions that are useful for downstream decision makers. We study decision makers who can choose amongst a set of actions $\mathcal{A} = \{1, \dots, K\}$[2] Agents will obtain utility that is a function of both the action they take, and the outcome $y \in \mathcal{C} \subseteq \mathbb{R}^d$ (which, in a game theoretic setting, may itself depend on the actions taken by the agents). We will assume that the utility functions are linear and Lipschitz-continuous in y.

Definition 5 (Decision maker's utility) *A utility function $u : \mathcal{A} \times \mathcal{C} \to [0, 1]$ maps an action $a \in \mathcal{A}$ and an outcome $y \in \mathcal{C}$ to a real number $u(a, y)$. We assume that for every action $a \in \mathcal{A}$:*

1. *$u(a, \cdot)$ is* linear *in its second argument: for all $\alpha_1, \alpha_2 \in \mathbb{R}$, $y_1, y_2 \in \mathcal{C}$,*

$$u(a, \alpha_1 y_1 + \alpha_2 y_2) = \alpha_1 u(a, y_1) + \alpha_2 u(a, y_2)$$

2. *$u(a, \cdot)$ is L-Lipschitz in its second argument, in the L_∞ norm: for all $y_1, y_2 \in \mathcal{C}$,*

$$|u(a, y_1) - u(a, y_2)| \leq L||y_1 - y_2||_\infty$$

We write $\mathcal{U}_L$ to denote a collection of L-Lipschitz utility functions of this form.

Remark 6 *For simplicity we assume that the utility function is* linear *in y, but we can equally well handle the case in which the utility function is* affine *in y: we simply augment the prediction space $\mathcal{C}$ with an extra $(d + 1)$-st coordinate that takes constant value 1. This preserves the convexity of $\mathcal{C}$, and now allows for arbitrary constant offsets in the utility of each action a. This allows us to capture many settings of interest. As an example, if there are d discrete outcomes that are payoff relevant to the decision maker in arbitrary ways, then we can define $\mathcal{C}$ as the simplex of probability distributions over outcomes, and let $u(a, p)$ be the expected utility for the agent who plays action a when the outcome is sampled from p; this is linear in p, and captures the setting studied in prior work [67, 91]; but our results extend to the more general case of arbitrary convex compact $\mathcal{C}$, and so we are not limited to talking about distributions over discrete outcomes. Most of our applications will take advantage of this generality.*

1. For the problem of online "quantile multicalibration", [7] gave bounds scaling with a quantity analogous to $\sqrt{n_t(E, \pi_T)}$, but at a cost of an exponentially sub-optimal dependence on other problem parameters.
2. Without loss of generality, we assume all downstream have the same action set. If not, let K be the cardinality of the largest action set, and introduce dummy actions for any agent with fewer actions.

A utility function naturally induces a best response function, mapping outcomes $y \in \mathcal{C}$ to actions $a \in \mathcal{A}$ that are utility maximizing given the outcomes:

Definition 7 (Best response) *Fix a utility function $u : \mathcal{A} \times \mathcal{C} \to [0, 1]$. The corresponding* Best Response *function $\delta_u : \mathcal{C} \to \mathcal{A}$ is defined as:*

$$\delta_u(y) = \operatorname*{argmax}_{a \in \mathcal{A}} u(a, y)$$

We assume that all ties are broken lexicographically. If $\delta_u(y) = a$, we say that a is a best response for utility function u given y. We write $E_{u,a}(y) = \mathbb{1}[\delta_u(y) = a]$ to denote the binary event that a is a best response to y for utility function u. Observe that for any utility function u, the set of events $\{E_{u,a}\}_{a \in \mathcal{A}}$ is disjoint.

Lemma 8 *For any utility function u that is linear in its second argument, the corresponding best response function δ_u has convex levelsets: for any $\alpha_1, \alpha_2 \in \mathbb{R}^{\geq 0}$ with $\alpha_1 + \alpha_2 \leq 1$, and any $y_1, y_2 \in \mathcal{C}$, if $\delta_u(y_1) = \delta_u(y_2) = a$, then $\delta_u(\alpha_1 y_1 + \alpha_2 y_2) = a$.*

Proof This follows from the linearity of u in its second argument and the definition of δ_u. For any alternative action $a' \in \mathcal{A}$, We can compute:

$$\begin{aligned} u(a, (\alpha_1 y_1 + \alpha_2 y_2)) &= \alpha_1 u(a, y_1) + \alpha_2 u(a, y_2) \\ &\geq \alpha_1 u(a', y_1) + \alpha_2 u(a', y_2) \\ &= u(a', (\alpha_1 y_1 + \alpha_2 y_2)) \end{aligned}$$

Here the first and last equalities follow from linearity of u in its second argument, and the inequality follows from the definition of the best response function δ_u. ■

As we make predictions $p_1, \ldots, p_t$ a decision maker with utility function u may use these predictions to take a sequence of actions $a_1, \ldots, a_t$. We call a decision maker *straightforward* if they take the predictions at face value, assuming that $y_t = p_t$:

Definition 9 (Straightforward Decision Maker) *A straightforward decision maker with utility function u treats predictions as correct and at each day t chooses $a_t = \delta_u(p_t)$.*

Because the predictions need not be correct, in hindsight (i.e. with knowledge of the realizations $y_1, \ldots, y_t$), a straightforward decision maker may regret not having taken some other sequence of actions. We study several kinds of regret in this paper. Φ-regret, as defined by [46], is defined with respect to a collection of mappings $\phi : \mathcal{A} \to \mathcal{A}$ from played actions to alternative actions called *strategy modification rules*.

Definition 10 (Φ-regret) *Fix a transcript π_T. The* regret *that a straightforward decision maker with utility function u has with respect to a strategy modification rule $\phi : \mathcal{A} \to \mathcal{A}$ is:*

$$r(\pi_T, u, \phi) = \frac{1}{T} \sum_{t=1}^{T} u(\phi(a_t), y_t) - u(a_t, y_t),$$

where $a_t = \delta_u(p_t)$ for each t.

Let Φ be a collection of strategy modification rules ϕ. We say that the decision maker has Φ-regret α if $r(\pi_T, u, \phi) \leq \alpha$ for all $\phi \in \Phi$.

Definition 11 (External regret and Swap regret [11, 32]) *For each action $a' \in \mathcal{A}$, let $\phi_{a'}$ be the constant function defined as $\phi(a) = a'$ for all $a \in \mathcal{A}$.* External *regret corresponds to Φ-regret for $\Phi_{\text{Ext}} = \{\phi_{a'} : a' \in \mathcal{A}\}$, the set of all constant strategy modification rules.*

Swap *regret corresponds to Φ-regret for Φ_{Swap} equal to the set of* all *strategy modification rules.*

Subsequence regret is defined by a collection of events, and requires that the decision maker have no *external* regret on any of the subsequences defined by the events.

Definition 12 (Subsequence regret [11, 69, 70]) *Fix a transcript π_T. The* regret *that a straightforward decision maker with utility function u has with respect to an event E and strategy modification rule $\phi : \mathcal{A} \to \mathcal{A}$ is:*

$$r(\pi_T, u, E, \phi) = \sum_{t=1}^{T} E(\pi_{t-1}, x_t, p_t)\left(u(\phi(a_t), y_t) - u(a_t, y_t)\right)$$

where $a_t = \delta_u(p_t)$ for each t.

Fix a collection of events $\mathcal{E}$. We say that the decision maker has $(\mathcal{E}, \alpha)$-regret if for every $E \in \mathcal{E}$ and for every $\phi \in \Phi_{\text{Ext}}$, $r(\pi_T, u, E, \phi) \leq \alpha(n_T(E, \pi_T))$.

Note that although that events are defined as a function of the *predictions* p_t, since a straightforward decision maker takes actions as a function of the prediction, we can equally well define events as a function of the actions a_t taken by the decision maker. Observe that for a decision maker with utility function u, swap regret corresponds to subsequence regret for the special case of $\mathcal{E} = \{E_{u,a} : a \in \mathcal{A}\}$, the set of subsequences on which the straightforward decision maker would take each action.

Finally, we consider *type* regret, which was studied (without a name) by [91]. Type regret, informally, is the regret that a straightforward decision maker has to straightforwardly optimizing some *other* utility function u' (or *type*) instead of her own utility function u.

Definition 13 (Type regret [91]) *Fix a transcript π_T. The* regret *that a straightforward decision maker with utility function u has with respect to an an alternative utility function u' is:*

$$r(\pi_T, u, u') = \frac{1}{T}\sum_{t=1}^{T} u(\delta_{u'}(p_t), y_t) - u(\delta_u(p_t), y_t)$$

Fix a collection of utility functions $\mathcal{U}$. We say that the decision maker has $\mathcal{U}$-regret α if for every $u' \in \mathcal{U}$, $r(\pi_T, u, u') \leq \alpha$.

Type regret is not a special case of Φ regret, since the comparator class cannot be described by any strategy modification rule.

Appendix C. Making Unbiased Predictions

In this section we give a general algorithm for obtaining α-unbiased predictions with respect to a collection of events $\mathcal{E}$, with running time scaling polynomially with d and $|\mathcal{E}|$, and error α scaling only logarithmically with d and $|\mathcal{E}|$. The algorithm and its analysis consist of two parts.

The first part is a standard reduction from the problem of online multiobjective optimization to minimax optimization, using the framework of [69]. Similar reductions appear in [50, 52]. Rather than giving an analysis from first principles as [69] do, we follow [52] in directly reducing to an experts problem (although rather than reducing to a sleeping experts problem as [52] do, we are able to get improved bounds by using the `MsMwC` algorithm of [16] for the standard experts problem). As [51, 52] do, we reduce to a no-regret algorithm that has "2nd order" regret bounds, which allow us to get more refined guarantees. This reduces the problem of obtaining online unbiased predictions to the problem of solving for the minimax equilibrium strategy of a particular zero sum game that has exponentially many (in d) strategies for each player, which is an obstacle to solving the problem straightforwardly. This step can be viewed as applying a contextual/time-varying variant of Blackwell's approachability theorem [9], in which the set to be "approached" may change at each round.

The second part of our algorithm solves this minimax problem — with a polylogarithmic running time dependence in the approximation parameter (via reduction to the Ellipsoid algorithm) in the case of $\mathcal{E}$ consisting of binary-valued and disjoint events, or with polynomial running time dependence in the approximation parameter using Follow-the-Perturbed-Leader in the more general case.

To preface the detailed derivation of the algorithm below, we give its pseudocode (presented as a single-round call, such that the learner needs to call `UnbiasedPrediction`$(\mathcal{E}, t, \pi_{t-1}, x_t)$ at every round t in the protocol of Section B), as well as its performance guarantees.

A few comments are in order. First, here and in the future we sometimes denote by $\mathcal{E}_t$ (or $\mathcal{E}^t$) the projection of the event collection $\mathcal{E}$ onto round t, i.e., the collection of mappings $\mathcal{E}_t := \{E(\pi_t, \cdot, \cdot) : \mathcal{X} \times \mathcal{C} \to [0, 1]\}$. This is both for notational convenience as well as to emphasize that our framework allows for the collection of events $\mathcal{E}$ to be revealed gradually over time, so long as the events $E(\pi_t, \cdot, \cdot)$ are available to the learner by the start of each round t.

Second, in the pseudocode below $g_{1:t-1}$ denotes the collection of *event gains* $\{g_1, \ldots, g_{t-1}\}$, which we construct to be used by the experts algorithm (`MsMwC`) in order to come up with *event weights* q_t in round t.[3]

Third, in the future we may leave out the π_t and x_t inputs to `UnbiasedPrediction` for brevity (and also if the setting is non-contextual and there is no x_t to pass), and may simply write calls to it as `UnbiasedPrediction`$(\mathcal{E}, t)$.

3. We do not explicitly pass $g_{1:t-1}$, nor the internal state of `MsMwC` at the end of the previous round, to `UnbiasedPrediction`, but it is understood that when `UnbiasedPrediction` is called at round t, this information is readily available for the next call to `MsMwC` (i.e., `MsMwC` does not need to recompute its previous trajectory from scratch in each round).

```
Calculate *event gains* from the preceding round $t-1$ (see Section C.1):

$g_{t-1} \leftarrow (g^{t-1}_{i,\sigma,E})_{i,\sigma,E},\quad$ where $g^{t-1}_{(i,\sigma,E)} = \sigma \cdot E(x_{t-1}, p_{t-1}) \cdot (p_{t-1,i} - y_{t-1,i})$ for $i \in [d], \sigma = \pm 1, E \in \mathcal{E}_{t-1}$

Compute *event weights* using Multiscale Multiplicative Weights with Correction (see Section C.1):
    $q_t = (q_{t,(i,\sigma,E)})_{i,\sigma,E} \leftarrow \texttt{MsMwC}(g_{1:t-1})$

Solve the following minimax problem *up to $\epsilon = 1/t$ error in minimax value*, using either the Ellipsoid method (if the events in $\mathcal{E}_t$ are binary and disjoint) or FTPL (otherwise) (see Section C.2):
    $\psi_t \leftarrow \operatorname{argmin}_{\psi'_t \in \Delta\mathcal{C}} \max_{y \in \mathcal{C}} \mathbb{E}_{p_t \sim \psi'_t} \left[ \sum_{i=1}^{d} \sum_{\sigma \in \{-1,1\}} \sum_{E \in \mathcal{E}_t} q_{t,(i,\sigma,E)} \cdot \sigma \cdot E(x_t, p_t) \cdot (p_{t,i} - y_i) \right]$

Return distribution over predictions $\psi_t$
```

Algorithm 1: `UnbiasedPrediction`$(\mathcal{E}, t, \pi_{t-1}, x_t)$

Theorem 14 (Guarantees for `UnbiasedPrediction`) *Given a convex compact prediction space $\mathcal{C} \subseteq \mathbb{R}^d$ and a collection $\mathcal{E}$ of events of size $|\mathcal{E}|$, our algorithm `UnbiasedPrediction` outputs, on any T-round transcript π_T, a sequence of distributions over predictions $\psi_1, \psi_2, \ldots, \psi_T \in \Delta\mathcal{C}$ satisfying:*

$$\mathbb{E}_{p_t \sim \psi_t\, \forall t} \left[\left| \sum_{t=1}^{T} E(x_t, p_t) \cdot (p_{t,i} - y_{t,i}) \right| \right] \le O\left(\ln(2d|\mathcal{E}|T) + \sqrt{\ln(2d|\mathcal{E}|T) \sum_{t=1}^{T} \mathbb{E}_{p_t \sim \psi_t} [(E(x_t, p_t))^2]} \right).$$

The per-round time complexity is polynomial in d, $|\mathcal{E}|$, and the time it takes to evaluate each $E \in \mathcal{E}$. When the events are binary and disjoint, the running time is polylogarithmic in t at any round $t \in [T]$; in the general case the running time is polynomial in t at any round $t \in [T]$.

Proof The theorem follows by combining the non-constructive minimax bound (i.e., the bound that assumes the minimax problem in each round $t \in [T]$ is solved up to an ϵ_t error in minimax value) of Theorem 20 with the runtime guarantees for solving the minimax game with binary disjoint events (Theorem 25) and with general, not necessarily binary or disjoint, events (Theorem 27). ■

Remark 15 ((Nearly-)Anytime Guarantee) *As formulated, our `UnbiasedPrediction` procedure does not rely on knowing the time horizon T, except through its oracle invocations of `MsMwC`. It can be checked that `MsMwC` only needs an upper bound on T — call it $T_{\max}$ — and thus by setting the internal components of `MsMwC` to depend on $T_{\max}$,* [4] *we automatically ensure that the bias of*

4. Specifically, the internal learning rate η of `MsMwC` should be set proportional to $\sqrt{\log T_{\max}}$, and its internal optimization should be performed over all weight vectors whose individual components are no less than a multiple of $1/T_{\max}$.

`UnbiasedPrediction` *will be:*

$$O\left(\ln(2d|\mathcal{E}|T_{\max}) + \sqrt{\ln(2d|\mathcal{E}|T_{\max})\sum_{t=1}^{T}\mathop{\mathbb{E}}_{p_t\sim\psi_t}[(E(x_t,p_t))^2]}\right) \quad \textit{after every round } T \textit{ such that } T \le T_{\max}.$$

The property of our online bound holding at all intermediate rounds, until the final time horizon, can be referred to as anytime.

It should be noted that algorithms in online learning are often only considered fully anytime if they have no *dependence at all on $T_{\max}$. Although* `UnbiasedPrediction` *does have a dependence on $T_{\max}$, (1) the regret bound depends on $T_{\max}$ only logarithmically and so is insensitive to even large over-estimates; and (2) the per-round runtime of* `UnbiasedPrediction` *(and of its subroutine* `MsMwC`*) does not depend on $T_{\max}$; rather, it scales with t at every round $t \in [T]$.*

C.1. Reduction to a Minimax Problem

We follow [50, 52, 69] in reducing a multicalibration-like problem to a multi-objective learning problem, in which for each event $E \in \mathcal{E}$ and each coordinate of our predictions $i \in [d]$, we have two objective values, one corresponding to the positive bias of our predictions, and one corresponding to the negative bias of our predictions. Our goal is to make predictions so that the maximum value across all of these $2d \cdot |\mathcal{E}|$ objectives is small, leading to small bias with respect to events $\mathcal{E}$.

We encode these $2d|\mathcal{E}|$ objectives as an experts problem with $n = 2d|\mathcal{E}|$ experts. We index the experts by (i, σ, E) for $i \in [d]$ representing a coordinate, $\sigma \in \{-1, 1\}$ representing a sign, and $E \in \mathcal{E}$ representing an event. At round t, if the learner predicts $p_t \in \mathcal{C}$ and the adversary produces outcome $y_t \in \mathcal{C}$, then the gains for each expert are defined to be:

$$g^t_{(i,\sigma,E)} = \sigma \cdot E(x_t, p_t) \cdot (p_{t,i} - y_{t,i})$$

Recalling our w.l.o.g. assumption that $2\max_{y\in\mathcal{C}} ||y||_\infty \le 1$, we observe that $g^t_{(i,\sigma,E)} \in [-1, 1]$, and that the cumulative gain of each expert (i, σ, E) up through round t is: $\sigma \cdot \sum_{\tau=1}^{t} E(x_\tau, p_\tau) \cdot (p_{\tau,i} - y_{\tau,i})$, which is exactly the σ-signed bias of the predictions through round t in coordinate i conditional on event E.

Recall that our goal is to make each of these σ-signed event-conditional bias terms small over time, i.e., each such term should over time diminish at a rate $\alpha(n_T(E))$, where E is the number of rounds on which the corresponding event E was active. To assist us in this, we invoke the MsMwC experts no-regret algorithm of [16], which was several years ago used to resolve the "impossible tuning" problem in online learning with experts.

Tool: MsMwC, A Small Loss Algorithm for the Experts Problem Multi-Scale Multiplicative Weights With Correction (MsMwC) [16] is an algorithm that can be applied to the experts learning problem in a way that simultaneously gets regret bounds to every expert that scale with the cumulative (squared) gain of that expert. Recall that in the experts learning setting, the learning algorithm must sequentially choose amongst n *experts* in rounds $t \in \{1, 2, \ldots\}$, whose *gains* (or losses) are determined by the adversary. In each round t:

1. The algorithm chooses a distribution $q_t \in \Delta[n]$ over the experts, and then:

2. A vector of gains for the experts $g_t \in [-1, 1]^n$ is chosen by an adversary, and the algorithm experiences gain $\hat{g}_t = q_t \cdot g_t$.

The *regret* that the algorithm has to expert i after t rounds is defined as $R_{t,i} = \sum_{\tau=1}^{t} (g_{t,i} - \hat{g}_t)$, and the goal in this setting is to make the average regret to each expert diminish over time. Classic algorithms for this problem, such as the Multiplicative Weights Update, attain $O(\sqrt{T})$ regret bounds relative to each expert; such regret bounds are not *adaptive*, in the sense that they don't depend on the profile of gains of each expert and thus cannot exploit any possible observed trends in each particular expert's gains. In contrast, the algorithm of [16] guarantees that we can simultaneously bound the regret to each expert i, $R_{t,i}$, by a quantity that scales with the cumulative (squared) gain of expert i:

Theorem 16 (Special Case of Theorem 1 from [16]) *There exists an algorithm (MsMwC) with per-round running time polynomial in n, that simultaneously for every expert i, guarantees regret:*

$$R_{T,i} \leq O\left(\ln(nT) + \sqrt{\ln(nT) \sum_{t=1}^{T} g_{t,i}^2}\right)$$

Applying MsMwC to event-conditional gains: We may now apply the regret-bound for MsMwC in Theorem 16) to find that:

Corollary 17 (MsMwC applied to event gains) *When using the gains constructed in this section, MsMwC plays a sequence of distributions $q_t \in \Delta[2d|\mathcal{E}|]$ that satisfies, for every triple $(i^*, \sigma^*, E^*) \in [d] \times \{-1, +1\} \times \mathcal{E}$,*

$$\sum_{t=1}^{T} \sum_{i \in [d], \sigma = \pm 1, E \in \mathcal{E}} q_{t,(i,\sigma,E)} \cdot \sigma \cdot E(x_t, p_t) \cdot (p_{t,i} - y_{t,i})$$
$$\geq \sigma^* \cdot \sum_{t=1}^{T} E^*(x_t, p_t) \cdot (p_{t,i^*} - y_{t,i^*}) - O\left(\ln(2d|\mathcal{E}|T) + \sqrt{\ln(2d|\mathcal{E}|T) \sum_{t \in [T]} (E^*(x_t, p_t))^2}\right).$$

Thus, to upper bound the bias in our predictions across all coordinates and all events, it suffices to make predictions p_t that upper bound the gain of MsMwC. Towards this end, we define a zero sum game between the learner and the adversary with objective function equal to the per-round expected gain of MsMwC:

$$u_t(p, y) = \sum_{i=1}^{d} \sum_{\sigma \in \{-1,1\}} \sum_{E \in \mathcal{E}} q_{t,(i,\sigma,E)} \cdot \sigma \cdot E(x_t, p) \cdot (p_i - y_i) \tag{1}$$

Non-constructively, we can argue that the learner has a prediction strategy which guarantees that the gain of MsMwC (and hence by construction, the bias of the cumulative predictions, conditional on any event $E \in \mathcal{E}$) is small. We will make use of Sion's minimax theorem.

Theorem 18 ([83]) *Let $X \subset \mathbb{R}^n$ and $Y \subset \mathbb{R}^m$ be convex and compact sets, and $u : X \times Y \to \mathbb{R}$ be a objective function, such that $u(x, \cdot)$ is upper semi-continuous and quasi-concave for all $x \in X$ and $u(\cdot, y)$ is lower semi-continuous and quasi-convex for all $y \in Y$. Then:*

$$\min_{x \in X} \max_{y \in Y} u(x, y) = \max_{y \in Y} \min_{x \in X} u(x, y).$$

The learner will play the role of the minimization player in this zero sum game, and the adversary will play the role of the maximization player. Their pure strategy spaces will both be (subsets of) $\mathcal{C}$, the prediction space. Observe that $u_t(p, y)$ is linear in the adversary's action y (and hence concave), and so we may take the adversary's action space to be $A_2 = \mathcal{C}$. On the other hand, $u_t(p, y)$ is *not* necessarily convex in p, because of the functions $E(x_t, p)$ (which may be arbitrary), and so to satisfy the conditions of Sion's minimax theorem, we must discretize the learner's action space and allow them to randomize.

Definition 19 *Let $\mathcal{C}_\epsilon$ be any finite ϵ-net of $\mathcal{C}$ in the ℓ_∞ norm — i.e. any finite set $\mathcal{C}_\epsilon \subset \mathcal{C}$ such that for all $p \in \mathcal{C}$, there is a $\hat{p} \in \mathcal{C}_\epsilon$ such that $||\hat{p} - p||_\infty \leq \epsilon$.*

We take the learner's action space in this game to be $A_1 = \Delta\mathcal{C}_\epsilon$, the set of distributions over an ϵ-net of the prediction space $\mathcal{C}$. This is a convex compact action space, and the utility function is linear over A_1 (by linearity of expectation).

We can compute the minimax value of the game by considering the world in which the adversary moves first, picking a realization $y \in \mathcal{C}$, and then the learner best responds with some $p \in \mathcal{C}_\epsilon$: Since $\mathcal{C}_\epsilon$ is an ϵ-net of $\mathcal{C}$, the learner can in particular choose p satisfying $|p_i - y_i| \leq \epsilon$ in every coordinate i. Together with Corollary 17, this gives us the following theorem:

Theorem 20 *If at every round $t \in [T]$ the learner samples a prediction from ψ_t, a $(1/t)$-approximate minimax strategy for the zero-sum game with utility function u_t defined in Equation 1, then for every coordinate i and every event $E \in \mathcal{E}$, the expected bias in coordinate i conditional on event E is bounded by:*

$$\mathop{\mathbb{E}}_{p_t \sim \psi_t \, \forall t}\left[\left|\sum_{t=1}^{T} E(x_t, p_t) \cdot (p_{t,i} - y_{t,i})\right|\right] \leq O\left(\ln(2d|\mathcal{E}|T) + \sqrt{\ln(2d|\mathcal{E}|T) \cdot \sum_{t=1}^{T} \mathop{\mathbb{E}}_{p_t \sim \psi_t}[(E(x_t, p_t))^2]}\right).$$

Proof Let $\epsilon_t = \frac{1}{t}$. By definition of an ϵ_t-net, for every $y \in \mathcal{C}$ we are guaranteed that there is a $p \in \mathcal{C}_{\epsilon_t}$ with $||p - y||_\infty \leq \epsilon_t$, and thus:

$$\max_{y \in \mathcal{C}} \min_{p \in \mathcal{C}_{\epsilon_t}} u_t(p, y) \leq \max_{E \in \mathcal{E}, i \in [d]} \epsilon_t \cdot \left|q_{t,(i,\sigma,E)} \cdot E(x_t, p)\right| \leq \epsilon_t.$$

Thus, Sion's minimax theorem implies that:

$$\min_{\psi \in \Delta\mathcal{C}_{\epsilon_t}} \max_{y \in \mathcal{C}} \mathop{\mathbb{E}}_{p \sim \psi}[u_t(p, y)] \leq \epsilon_t.$$

Let ψ_t be any ϵ_t-approximate minimax distribution of this game, such that: $\max_{y \in \mathcal{C}} \mathbb{E}_{p_t \sim \psi_t}[u_t(p_t, y)] \leq \epsilon_t$. If the learner samples a prediction from ψ_t at each round t, then by definition of the utility function u_t, the expected loss of MsMwC will be bounded by ϵ_t at each round t. By Corollary 17, after taking expectations with respect to the randomness in the learner's play according to the distributions ψ_t across rounds $t \in [T]$, we have that the expected bias in each coordinate i conditional on each event $E \in \mathcal{E}$ is bounded as:

$$\mathop{\mathbb{E}}_{p_t \sim \psi_t \, \forall t}\left[\left|\sum_{t=1}^{T} E(x_t, p_t) \cdot (p_{t,i} - y_{t,i})\right|\right] \leq \sum_{t=1}^{T} \epsilon_t + O\left(\ln(2d|\mathcal{E}|T) + \mathop{\mathbb{E}}_{p_t \sim \psi_t \, \forall t} \sqrt{\ln(2d|\mathcal{E}|T) \cdot \sum_{t \in [T]} (E(x_t, p_t))^2}\right)$$

$$\leq O(\ln T) + O\left(\ln(2d|\mathcal{E}|T) + \mathop{\mathbb{E}}_{p_t\sim\psi_t\,\forall t}\sqrt{\ln(2d|\mathcal{E}|T)\cdot\sum_{t\in[T]}(E(x_t,p_t))^2}\right)$$
$$\leq O\left(\ln(2d|\mathcal{E}|T) + \sqrt{\ln(2d|\mathcal{E}|T)\cdot\sum_{t\in[T]}\mathop{\mathbb{E}}_{p_t\sim\psi_t}[(E(x_t,p_t))^2]}\right),$$

where the second line follows by bounding the harmonic series, and the final line is by Jensen's inequality with respect to the (concave) square root function. ■

Theorem 20 gives a (non-constructive) algorithm for producing predictions that have bounded bias conditional on all events in $\mathcal{E}$. The difficulty is that the algorithm requires playing an approximate minimax strategy for a game in which the learner has an exponentially large strategy space, and the adversary has a continuously large strategy space. In Section C.2, we show how to make this algorithm constructive by giving polynomial time algorithms for solving this minimax problem.

C.2. Solving the Minimax Problem

C.2.1. For Disjoint Binary Convex Events

In this section we show how to find an ϵ-approximate solution to the minimax problem defined in Section C.1, with running time that is polynomial in d, $|\mathcal{E}|$, and $\log(1/\epsilon)$ in the case in which the events $E \in \mathcal{E}$ are binary valued and disjoint: for all x, p: $\sum_{E\in\mathcal{E}} E(x,p) \leq 1$. We will also assume that for every history π and context x, the predictions p that satisfy $E(\pi, x, p) = 1$ form a convex set for which we have a polynomial time separation oracle. In Section C.2.2 we will show how to solve the same minimax problem in the general case, without any of these assumptions (assuming only that we can *evaluate* the quantity $E(\pi, x, p)$ in polynomial time) — with the caveat that our general solution will have a running time dependence that is polynomial in $1/\epsilon$ rather than $\log 1/\epsilon$. Proofs from this Section can be found in Appendices G and H.

Following the reduction to a zero-sum game in Section C.1, our goal is to solve for the learner's equilibrium strategy $\psi_t \in \Delta(\mathcal{C})$ in the game with utility function

$$u_t(p, y) = \sum_{i=1}^{d}\sum_{\sigma\in\{-1,1\}}\sum_{E\in\mathcal{E}} q_{t,(i,\sigma,E)}\cdot\sigma\cdot E(\pi_{t-1}, x_t, p)\cdot(p_i - y_i)$$

corresponding to the per-round gain of MsMwC. In other words, we need to approximately solve the minimax problem defined as:

$$\psi_t^* = \mathop{\mathrm{argmin}}_{\psi\in\Delta(\mathcal{C})}\max_{y\in\mathcal{C}}\mathop{\mathbb{E}}_{p\sim\psi}[u_t(p, y)] \tag{2}$$

By relaxing the minimization player's domain from $\mathcal{C}$ to $\Delta(\mathcal{C})$, the set of distributions over predictions, we have made the objective linear (and hence convex/concave), but we have continuously many optimization variables — both primal variables (for the minimization player) and dual variables (for the maximization player). Our strategy for solving this problem in polynomial time will be to argue that it has a solution in which only $|\mathcal{E}|$ many primal variables take non-zero values, that we can efficiently identify those variables, and that we can implement a separation oracle for the dual "constraints" in polynomial time. This will allow us to construct a reduced but equivalent linear program that we can efficiently solve with the Ellipsoid algorithm.

We first observe that in the utility function $u_t(p, y)$, the learner's predictions p "interact" with the outcomes y only through the activation of the events $E(\pi_{t-1}, x_t, p)$. This implies that *conditional* on the values of the events $E(\pi_{t-1}, x_t, p)$, there is a unique p_t that minimizes $u_t(p, y)$ *simultaneously* for all y. In general the collection of events $E(\pi_{t-1}, x_t, p)$ could take on many different combinations of values — but our assumption in this section that the events are disjoint and binary means that there are in fact only $|\mathcal{E}|$ different values for us to consider. The predictions we need to consider are defined by the following efficiently solvable convex programs:

Definition 21 *For $E \in \mathcal{E}$, let $p_t^{*,E}$ be a solution to the following convex program (selecting arbitrarily if there are multiple optimal solutions):*

$$\begin{aligned} &\text{minimize}_{p\in\mathcal{C}} && \sum_{i=1}^{d} \sum_{\sigma\in\{-1,1\}} q_{t,(i,\sigma,E)} \cdot \sigma \cdot p_i \\ &\text{subject to} && \\ & && E(\pi_{t-1}, x_t, p) = 1 \end{aligned}$$

Let $\mathcal{P}_t = \{p_t^{*,E}\}_{E\in\mathcal{E}}$ be a collection of $|\mathcal{E}|$ vectors in $\mathcal{C}$ constituting solutions to the above programs.

Remark 22 *As we have assumed in this section, the set of p such that $E(\pi_{t-1}, x_t, p) = 1$ is a convex region endowed with a separation oracle, and so these are indeed convex programs that we can efficiently solve with the Ellipsoid algorithm. This is often the case: for example, if we have a decision maker with a utility function u over K actions, the disjoint binary events $E_{u,a}$ (for each action $a \in [K]$) are defined by K linear inequalities, and so form a convex polytope with a small number of explicitly defined constraints. This will turn out to be the collection of events relevant for obtaining diminishing swap regret for downstream decision makers.*

We next verify that the prediction values defined in Definition 21 are best responses for the minimization player against all possible realizations y_t that the maximization player might choose, *conditional* on a positive value of a particular event:

Lemma 23 *Simultaneously for all $y \in \mathcal{C}$, we have:*

$$p_t^{*,E} \in \underset{p:E(\pi_{t-1},x_t,p)=1}{\operatorname{argmin}} u_t(p, y).$$

A consequence of this is that solutions to the following reduced minimax problem (which now has only $|\mathcal{E}|$ variables for the minimization player — the weights defining a distribution over the $|\mathcal{E}|$ points $p_t^{*,E}$) are also solutions to our original minimax problem 2:

$$\psi_t^* = \underset{\psi\in\Delta(\mathcal{P}_t)}{\operatorname{argmin}} \max_{y\in\mathcal{C}} \mathop{\mathbb{E}}_{p\sim\psi} [u_t(p, y)] \tag{3}$$

Lemma 24 *Fix any optimal solution ψ_t^* to minimax problem 3. Then ψ_t^* is also an optimal solution to minimax problem 2.*

Thus, to find a solution to minimax problem 2, it suffices to find a solution to minimax problem 3. Minimax problem 3 can be expressed as a linear program with $|\mathcal{E}| + 1$ variables but with continuously many constraints, one for each $y \in \mathcal{C}$:

$$\begin{aligned} &\text{minimize}_{\psi \in \Delta(\mathcal{P}_t)} \quad \gamma \\ &\text{subject to} \\ &\qquad\qquad \mathbb{E}_{p\sim\psi}[u_t(p,y)] \le \gamma \quad \forall y \in \mathcal{C} \end{aligned} \tag{4}$$

We can find an ϵ-approximate solution to a polynomial-variable linear program using the Ellipsoid algorithm in time polynomial in the number of variables and $\log(1/\epsilon)$ so long as we have an efficient *separation oracle* — i.e., an algorithm to find an ϵ-violated constraint whenever one exists, given a candidate solution. In this case, implementing a separation oracle corresponds to computing a *best response* for the adversary (the maximization player) in our game—and since the utility function in our game given in Equation 1 is *linear* in the Adversary's chosen action y, implementing a separation oracle corresponds to solving a linear maximization problem over the convex feasible region $\mathcal{C}$—a problem that we can solve efficiently assuming we have a separation oracle for $\mathcal{C}$. There are a number of technical details involved in making this rigorous, which can be found in Appendix G. Here we state the final algorithm and guarantee.

```
for $E \in \mathcal{E}$ do
    Solve the convex program from Definition 21 to obtain $p^{*,E}$.
Let $\mathcal{P}_t = \{p^{*,E}\}_{E\in\mathcal{E}}$.
Solve linear program 4 over $\mathcal{P}_t$ using the weak Ellipsoid algorithm to obtain solution $\psi'_t$.
if $\psi'_t \notin \Delta(\mathcal{P})$ then
    Let $\psi^*_t$ be the Euclidean projection of $\psi_t$ onto $\Delta(\mathcal{P})$ returned by the simplex projection algorithm.
else
    Let $\psi^*_t = \psi'_t$.
return $\psi^*_t$
```
Algorithm 2: `Get-Approx-Equilibrium-LP` $(t, \epsilon, \mathcal{E})$

Theorem 25 *Given a polynomial-time separation oracle for $\mathcal{C}$, for any $\epsilon > 0$, there exists an algorithm (Algorithm 2) that returns an ϵ-approximately optimal solution ψ^*_t to minimax problem 2 and runs in time polynomial in d, $|\mathcal{E}|, \log(\frac{1}{\epsilon})$.*

C.2.2. THE GENERAL CASE

For a general collection of events — which are not necessarily disjoint, may be real valued, and may not correspond to convex subsets of the prediction space — the technique we gave in Section C.2.1 for approximately solving the game with a polylogarithmic dependence on the approximation parameter no longer applies. Nevertheless, by simulating play of the zero-sum game that we wish to solve using appropriately chosen learning dynamics, we can still compute an ϵ-approximate minimax strategy in time polynomial in $1/\epsilon$.

At a high level, our strategy will be to simulate play of the zero-sum game with objective function defined in Equation 1. The maximization player (the adversary) will play according to the Follow-the-Perturbed-Leader algorithm introduced below. This allows us to efficiently minimize regret over the adversary's high dimensional action space because we can express the adversary's utility function as a linear function of their own action, which casts their learning problem as an instance of online linear optimization. Most straightforwardly, we would like the minimization player (the Learner) to *best respond* to the Adversary's actions at each round: but the Learner's utility function is complex, because of interactions with the events E, which can be arbitrarily defined. Thus it is not clear how to efficiently compute a best response for the learner. Nevertheless, the structure of the utility function makes it such that they can guarantee themselves the value of the game at each round simply by copying the Adversary's strategy, which is computationally easy. Although this is not necessarily a best response, it suffices for our purposes: standard arguments will imply that the time-averaged play for the Learner converges to an approximate minimax strategy of the game.

Tool: FTPL, An Oracle Efficient Algorithm for Online Linear Optimization In the present setting of general (not necessarily disjoint) events, we will make use of *Follow the Perturbed Leader (FTPL)*, an oracle efficient algorithm for online linear optimization [53, 62]. In the online linear optimization problem, the learner has a decision space $\mathcal{D} \subseteq \mathbb{R}^n$ and the adversary has a state space $\mathcal{S} \subseteq \mathbb{R}^n$. In rounds $t \in \{1, \ldots, \}$:

1. The algorithm chooses a distribution over decisions $D_t \in \Delta\mathcal{D}$;
2. The adversary chooses a state $s_t \in \mathcal{S}$;
3. The algorithm experiences gain $\hat{g}_t = \mathbb{E}_{d_t \sim D_t}[\langle d_t, g_t \rangle] = \langle \mathbb{E}_{d_t \sim D_t}[d_t], g_t \rangle$.

After t rounds, the regret of the algorithm to decision $d \in \mathcal{D}$ is: $R_{t,d} = \sum_{\tau=1}^{t} \left(\langle d, s_\tau \rangle - \hat{g}_\tau \right)$.

In contrast to the experts problem defined in Section C.1, in which the number of choices for the algorithm at each round is n, here the algorithm can choose any element of $\mathcal{D}$ at each round, which may be exponentially large in n (or continuously large). An algorithm ("Follow the Perturbed Leader") of [62] gives an *oracle efficient* algorithm for obtaining diminishing regret guarantees for this problem, which means that it runs in time polynomial in n and the time needed to solve linear optimization problems over $\mathcal{D}$ — i.e. to solve problems of the form $d_t = \text{argmax}_{d \in \mathcal{D}} \langle d, s \rangle$ for any vector $s \in \mathbb{R}^n$. The algorithm is simple and maintains its distribution D_t at each round implicitly by giving an efficient sampling algorithm: At round t, D_t is the distribution corresponding to the sampling algorithm:

1. Let $s = \sum_{\tau=1}^{t-1} s_\tau + z$, where $z \in \mathbb{R}^n$ is such that each coordinate z_i is sampled $z_i \sim \text{Unif}[0, 1/\delta]$ from the uniform distribution over the interval $[0, 1/\delta]$ for some parameter δ.
2. Let $d_t = \text{argmax}_{d \in \mathcal{D}} \langle d, s \rangle$.

Theorem 26 ([62]) *Let $\Delta = \sup_{d_1, d_2 \in \mathcal{D}} ||d_1 - d_2||_1$ be the diameter of the decision space, let $K = \sup_{d \in \mathcal{D}, s \in \mathcal{S}} |\langle d, s \rangle|$ be the maximum per-round gain of the algorithm, and let $A = \sup_{s \in \mathcal{S}} ||s||_1$ be the maximum norm of any state. Then for any $\delta \leq 1$, Follow the Perturbed Leader has regret at round T to each decision d that is at most:*

$$R_{T,d} \leq \delta KAT + \frac{\Delta}{\delta}.$$

Choosing $\delta = \sqrt{\frac{\Delta}{KAT}}$ gives:

$$\max_{d \in \mathcal{D}} R_{T,d} \leq 2\sqrt{\Delta KAT}.$$

Simplifying the Adversary's objective: When thinking of the Adversary's optimization problem, it is helpful to elide the parts of the objective function (from Equation 1) that are independent of the adversary's actions. Towards this end we define the following objective:

$$u'_t(p, y) = \sum_{i=1}^{d} -y_i \cdot s_{t,i}(p) = \langle -y, s_t(p) \rangle,$$

where for any $p \in \mathcal{C}$, we define:

$$s_{t,i}(p) = \sum_{\sigma \in \{-1,1\}} \sum_{E \in \mathcal{E}} q_{t,(i,\sigma,E)} \cdot \sigma \cdot E(x_t, p) \tag{5}$$

for each $i \in [d]$. Note the relationship between the two objective functions:

$$u_t(p, y) = \langle (p - y), s_t(p) \rangle = \langle p, s_t(p) \rangle + u'_t(p, y). \tag{6}$$

Since the difference between $u_t(p, y)$ and $u'_t(p, y)$ (the term $\langle p, s_t(p) \rangle$) is independent of the Adversary's chosen action y, the Adversary's regret as measured with respect to u'_t is identical to regret as measured with respect to u_t; thus from this point onwards, we imagine the Adversary to be optimizing for u'_t.

Observe that fixing p (and hence $s_t(p)$), $u'_t(p, \cdot)$ is a linear gain function of exactly the form that Follow-the-Perturbed-Leader is designed to optimize. To implement Follow-the-Perturbed-Leader, we will need to assume the existence of an online linear optimization oracle over $\mathcal{C}$ — i.e. an optimization algorithm $M : \mathbb{R}^d \to \mathcal{C}$ which solves:

$$M(s) \in \underset{y \in \mathcal{C}}{\text{argmax}} \ \langle -y, s \rangle.$$

The complete algorithm is stated in Algorithm 3.

```
Initialize $s_t(p_0)$ to 0.
Set $T' = \frac{2dC^2}{\epsilon'^2}$, $\delta = \sqrt{\frac{2d}{T'}}$
Fix distribution $\mathcal{Z} = \text{Unif}[0, \frac{1}{\delta}]^d$.
**for** $\tau = 1, \dots, T'$ **do**
    Compute $y_\tau = \mathbb{E}_{z \in \mathcal{Z}} \left[ M\left( \sum_{k=0}^{\tau-1} s_t(p_k) + z \right) \right]$ // Adversary chooses the FTPL distribution
    Set $p_\tau = y_\tau$ // Learner copies Adversary's expected action
    Compute $s_t(p_\tau)$ as defined in Equation 5.
Define $\overline{p}$ as the uniform distribution over the sequence $(p_1, p_2, \cdots, p_{T'})$.
**return** $\overline{p}$
```

Algorithm 3: Get-Approx-Equilibrium(t, ϵ')

Theorem 27 *For any $t \in [T]$ and $\epsilon' > 0$, Algorithm 3 returns a distribution over actions $\overline{p}$ which is an ϵ'-approximate minimax equilibrium strategy for the zero-sum game with objective u_t.*

To prove Theorem 27, we first establish an intermediate fact — that within the learning dynamics simulated in Algorithm 3, the Adversary has low regret. The proof can be found in Appendix I, and just instantiates the guarantees of Follow the Perturbed Leader in our setting.

Lemma 28 *Playing FTPL for T' rounds with perturbation parameter $\delta = \sqrt{\frac{2d}{T'}}$, for a learner with decision space $\mathcal{C}$ and adversary with state space $\mathcal{S} = \{s_t(p) \mid p \in \mathcal{C}\}$ yields a regret bound:*

$$R_{T',y} \leq \sqrt{2dC^2T'}$$

for all $y \in \mathcal{C}$, where d is the dimension of all vectors in $\mathcal{C}$ and $C = 2\max_{y\in\mathcal{C}} \|y\|_\infty$.

Proof [Proof of Theorem 27] Define y^* as the adversary's best response in hindsight to the realized sequence of actions taken by the learner during the run of Algorithm 3:

$$y^* = \operatorname*{argmax}_{y\in\mathcal{C}} \sum_{\tau=1}^{T'} u'_t(p_\tau, y^*) = \operatorname*{argmax}_{y\in\mathcal{C}} \sum_{\tau=1}^{T'} u_t(p_\tau, y),$$

where in the second equality we use the fact that the Adversary's best-response function is identical under u and u'. Let D_τ denote the adversary's FTPL distribution over actions at round τ — i.e. the distribution defined by $M\left(\sum_{k=0}^{\tau-1} s_t(p_k) + Z\right)$, where $Z \sim \mathcal{Z}$. Looking first at the expected gain of the adversary over the realized transcript, we see that:

$$\sum_{\tau=1}^{T'} \mathop{\mathbb{E}}_{y\in D_\tau} [u'_t(p_\tau, y)] \geq \sum_{\tau=1}^{T'} u'_t(p_\tau, y^*) - R_{T',y^*},$$

where R_{T',y^*} is the adversary's regret from playing FTPL. The cumulative utility that the adversary would have achieved with their best fixed action in hindsight (as measured with respect to the original utility function u_t) can now be bounded as:

$$\begin{aligned}\max_{y\in\mathcal{C}} \sum_{\tau=1}^{T'} u_t(p_\tau, y) = \sum_{\tau=1}^{T'} u_t(p_\tau, y^*) &= \sum_{\tau=1}^{T'} u'_t(p_\tau, y^*) + \sum_{\tau=1}^{T'} \langle p_\tau, s_t(p_\tau)\rangle \\ &\leq \left(R_{T',y^*} + \sum_{\tau=1}^{T'} \mathop{\mathbb{E}}_{y\sim D_\tau} \left[u'_t(p_\tau, y)\right]\right) + \sum_{\tau=1}^{T'} \langle p_\tau, s_t(p_\tau)\rangle.\end{aligned}$$

The first line follows from Equation 6. To simplify the expected sum of utilities $\sum_{\tau=1}^{T'} \mathbb{E}_{y\sim D_\tau}\left[u'_t(p_\tau, y)\right]$, we use the fact that at each round τ in Algorithm 3, the learner's response p_τ is defined to be exactly the expectation of the adversary's distribution D_τ. So, the expected utility under the *original* objective function u_τ for the realized sequence of plays can be computed for each τ:

$$\mathop{\mathbb{E}}_{y\sim D_\tau} [u_t(p_\tau, y)] = \mathop{\mathbb{E}}_{y\sim D_\tau} [\langle (p_\tau - y), s_t(p_\tau)\rangle] = \left\langle \left(p_\tau - \mathop{\mathbb{E}}_{y\sim D_\tau}[y]\right), s_t(p_\tau)\right\rangle = 0$$

Using Equation 6 once more,

$$\begin{aligned}0 = \sum_{\tau=1}^{T'} \mathop{\mathbb{E}}_{y\sim D_\tau} [u_t(p_\tau, y)] &= \sum_{\tau=1}^{T'} \mathop{\mathbb{E}}_{y\sim D_\tau} \left[u'_t(p_\tau, y)\right] + \sum_{\tau=1}^{T'} \langle p_\tau, s_t(p_\tau)\rangle \\ \implies \sum_{\tau=1}^{T'} \mathop{\mathbb{E}}_{y\sim D_\tau} \left[u'_t(p_\tau, y)\right] &= -\sum_{\tau=1}^{T'} \langle p_\tau, s_t(p_\tau)\rangle\end{aligned}$$

Substituting this into the bound on the utility of the adversary's best response in hindsight:

$$\begin{aligned}\max_{y\in\mathcal{C}}\sum_{\tau=1}^{T'} u_t(p_\tau, y) &\le \left(R_{T',y^*} - \sum_{\tau=1}^{T'}\langle p_\tau, s_t(p_\tau)\rangle\right) + \sum_{\tau=1}^{T'}\langle p_\tau, s_t(p_\tau)\rangle \\ &= R_{T',y^*} \\ &\le \sqrt{2dC^2T'},\end{aligned}$$

with the last inequality following directly from Lemma 28. Notice that for any fixed strategy $y \in \mathcal{C}$:

$$\frac{1}{T'}\sum_{i=1}^{T'} u(p_\tau, y) = \mathop{\mathbb{E}}_{p\sim\overline{p}}[u_t(p, y)],$$

where $\overline{p}$ is the learner's mixed strategy returned from Algorithm 3, the uniform distribution over all realized vectors p_τ in the transcript. Thus,

$$\max_{y\in\mathcal{C}}\mathop{\mathbb{E}}_{p\sim\overline{p}}[u_t(p, y)] \le \frac{\sqrt{2dC^2T'}}{T'},$$

and so $\overline{p}$ is a $\sqrt{\frac{2dC^2}{T'}}$-approximate equilibrium strategy for the learner. By our choice of $T' = \frac{2dC^2}{\epsilon'^2}$, this simplifies to ϵ', as desired. ■

Assuming we can compute the term $\mathbb{E}_{D_\tau}[y_\tau]$ exactly for each round τ, Algorithm 3 offers a computationally efficient method for finding an ϵ'-approximate minimax strategy for the learner. However, the ability to do so depends on the geometric structure of $\mathcal{C}$, and though there are natural sets $\mathcal{C}$ (such as the unit hypercube and the simplex) for which obtaining a closed-form expression for the expectation is straightforward, in general this may not be the case. In these cases, we may sample from the distribution D_τ to obtain an approximate value for the expectation at round τ. We give the sampling-based Algorithm 6, along with Theorem 87 that establishes its guarantees, in Appendix I. The proof of Theorem 87 will be similar to the proof of Theorem 27 but with additional use of concentration inequalities in order to bound the sampling errors.

Appendix D. Connecting Predictions and Decision Making

We next make connections between our ability to make unbiased predictions and the quality of decisions that are made downstream as a function of our predictions in a general setting. The form of the argument will proceed in the same way that it will in our main applications, and so is instructive.

Specifically, we will show that a straightforward decision maker who simply best responds to our predictions can be guaranteed both *no swap regret* and *no type regret* (see Section B.2 for definitions) — if we just make our predictions *unbiased* conditional on the events defined by the decision maker's best-response correspondence.

The Predict-then-Act Paradigm These results, whose formal statements are given below, suggest a natural design paradigm for sequential decision algorithms, which we call *predict-then-act*. The idea is simple: first we make a prediction p_t for an unknown payoff-relevant parameter y_t, and then we choose an action as if our prediction were correct — i.e. we best respond to p_t. This is nothing more than implementing a straightforward decision maker for our predictions. We can parameterize the predict-then-act algorithm with various events $\mathcal{E}$, such that our predictions will be unbiased with respect to events in $\mathcal{E}$. Whenever $\mathcal{E}$ is a collection of polynomially many events that can each be evaluated in polynomial time, the predict-then-act algorithm can be implemented in polynomial time per step. While Predict-Then-Act is quite simple, its flexibility in a variety of settings lies in the design of the event set $\mathcal{E}$ and prediction space $\mathcal{C}$. By choosing the events $\mathcal{E}$ to be appropriately tailored to the task at hand, we can arrange that Predict-Then-Act has guarantees of various sorts.

```
for $t$ in $1 \dots T$ do
    Compute $\psi_t \leftarrow$ UnbiasedPrediction($\mathcal{E}, t$)
    Predict $p_t \sim \psi_t$
    for $u_i \in \mathcal{U}$ do
        Decision maker $i$ selects action $a_t = \delta_{u_i}(p_t) = \mathrm{argmax}_{a \in \mathcal{A}}\, u_i(a, p_t)$
    Observe outcome $y_t \in \mathcal{C}$
Algorithm 4: Predict-Then-Act($T, \mathcal{U}, \mathcal{E}, \mathcal{C}, \mathcal{A}$)
```

D.1. No Swap Regret

We start by showing that predictions that are unbiased conditional on the outcome of the best response functions of downstream decision makers cause the actions of those downstream decision makers to have low swap regret. Similar observations have been previously made by [77] and [52] in the context of a single decision maker in the experts learning problem. In contrast, [32] showed a similar connection (also for a single decision maker in the experts learning problem) for *full distributional calibration*, which conditions on *every* event in a (discretization) of the prediction space. This is an exponentially large number of conditioning events in the dimension of the prediction space. The advantage of the theorem below is that it requires a number of conditioning events that is only *linear* in the number of utility functions u of interest and downstream actions a, and — in view of our unbiasedness guarantees obtained in Section C gives a regret bound that scales only logarithmically in the dimension d of the decision space using an algorithm with running time that is only polynomial in d.

Theorem 29 *Fix a collection of L-Lipschitz utility functions $\mathcal{U}_L$, an action set $\mathcal{A} = \{1, \dots, K\}$, and any transcript π_T. Let $\mathcal{E} = \{E_{u,a} : u \in \mathcal{U}_L, a \in \mathcal{A}\}$ be the set of binary events corresponding to straightforward decision makers with utility functions $u \in \mathcal{U}_L$ taking each action $a \in \mathcal{A}$. Then if π_T is α-unbiased with respect to $\mathcal{E}$, for every $u \in \mathcal{U}_L$, the straightforward decision maker with utility function u has swap regret at most:*

$$\max_{\phi:\mathcal{A}\to\mathcal{A}} r(\pi_T, u, \phi) \le \frac{2L \sum_{a\in\mathcal{A}} \alpha(n_T(E_{u,a}, \pi_T))}{T}$$

If α is concave, then this bound is at most:

$$\max_{\phi:\mathcal{A}\to\mathcal{A}} r(\pi_T, u, \phi) \le \frac{2LK\alpha(T/K)}{T}$$

The logic of the proof is simple. For every action a and alternative action b, we consider the subsequence of rounds on which a straightforward decision maker would have chosen to play a in response to our predictions. Our predictions are unbiased conditional on the event that the decision maker chooses to play a — i.e. this subsequence — and so (on average over the subsequence), the decision maker's estimates for the payoff they received for playing a, as well as the payoff that *they would have received* for playing b are correct. Moreover, the reason they chose to play a on each round in this subsequence (as a straightforward decision maker) was because pointwise, on each round, we estimated that a would obtain higher payoff than b. Thus, it must be that on average over the subsequence, a really did obtain higher payoff than b. Since this is true for every pair of actions, the decision maker must have had no swap regret. The proof formalizes this logic:

Proof [Proof of Theorem 29] Fix any $\phi : \mathcal{A} \to \mathcal{A}$ and any $u \in \mathcal{U}_L$. We need to upper bound $r(\pi_T, u, \phi)$. Using the linearity of $u(a, \cdot)$ in its second argument for all $a \in \mathcal{A}$, we can write:

$$\begin{aligned}
r(\pi_T, u, \phi) &= \frac{1}{T}\sum_{t=1}^{T} u(\phi(\delta_u(p_t)), y_t) - u(\delta_u(p_t), y_t) \\
&= \frac{1}{T}\sum_{a\in\mathcal{A}}\sum_{t:\delta_u(p_t)=a} u(\phi(a), y_t) - u(a, y_t) \\
&= \sum_{a\in\mathcal{A}}\frac{1}{T}\sum_{t=1}^{T} E_{u,a}(p_t)\left(u(\phi(a), y_t) - u(a, y_t)\right) \\
&= \sum_{a\in\mathcal{A}}\left(u\left(\phi(a), \frac{1}{T}\sum_{t=1}^{T} E_{u,a}(p_t)y_t\right) - u\left(a, \frac{1}{T}\sum_{t=1}^{T} E_{u,a}(p_t)y_t\right)\right) \\
&\leq \sum_{a\in\mathcal{A}}\left(u\left(\phi(a), \frac{1}{T}\sum_{t=1}^{T} E_{u,a}(p_t)p_t\right) - u\left(a, \frac{1}{T}\sum_{t=1}^{T} E_{u,a}(p_t)p_t\right) + \frac{2L\alpha(n_T(E_{u,a}, \pi_T))}{T}\right) \\
&\leq \sum_{a\in\mathcal{A}}\left(\frac{2L\alpha(n_T(E_{u,a}, \pi_T))}{T}\right) \\
&= \frac{2L\sum_{a\in\mathcal{A}}\alpha(n_T(E_{u,a}, \pi_T))}{T}
\end{aligned}$$

Here the first inequality follows from the α-unbiasedness condition and the L-Lipschitzness of u: indeed, for every a' (and in particular for $a' \in \{a, \phi(a)\}$) we have

$$\left|u\left(a', \frac{1}{T}\sum_{t=1}^{T} E_{u,a}(p_t)p_t\right) - u\left(a', \frac{1}{T}\sum_{t=1}^{T} E_{u,a}(p_t)y_t\right)\right| \leq L||\frac{1}{T}\sum_{t}(p_t - y_t)E_{u,a}(p_t)||_\infty$$

$$= \frac{L}{T}\max_{i\in[d]}|\sum_{t}(p_{t,i} - y_{t,i})E_{u,a}(p_t)| \leq \frac{\alpha(n_T(E_{u,a}, \pi_T))L}{T}.$$

The 2nd inequality follows from the fact that by definition, whenever $\delta_u(p_t) = a$ (and hence whenever $E_{u,a}(p_t) = 1$), $u(a, p_t) \geq u(a', p_t)$ for all $a' \in \mathcal{A}$, and the fact that by Lemma 8, the levelsets of δ_u are convex, and hence $u(a, \frac{1}{T}\sum_{t=1}^{T} E_{u,a}(p_t)p_t) \geq u(a', \frac{1}{T}\sum_{t=1}^{T} E_{u,a}(p_t)p_t)$ for all a'.

Recall that for any utility function u, the events $\{E_{u,a}\}_{a\in\mathcal{A}}$ are disjoint, and so for any u and any π_T, $\sum_{a\in\mathcal{A}} n_T(E_{u,a}, \pi_T) \leq T$. Therefore, whenever α is a concave function (as it is for the algorithm we give in this paper, and as it is for essentially any reasonable bound), the term $\sum_{a\in\mathcal{A}} \alpha(n_T(E_{u,a}, \pi_T))$ evaluates to at most $K\alpha(T/K)$. ■

D.2. No Type Regret

Predictions that are unbiased conditional on the outcome of the best response functions of downstream decision makers with utility functions in $\mathcal{U}$ also suffice to guarantee that downstream decision makers have no type regret to any utility function in $\mathcal{U}$. Guarantees of this sort were first given by [91], in a batch setting.

Theorem 30 *Fix a collection of L-Lipschitz utility functions $\mathcal{U}_L$, an action set $\mathcal{A} = \{1, \ldots, K\}$, and any transcript π_T. Let $\mathcal{E} = \{E_{u,a} : u \in \mathcal{U}, a \in \mathcal{A}\}$ be the set of binary events corresponding to straightforward decision makers with utility functions $u \in \mathcal{U}_L$ taking each action $a \in \mathcal{A}$. Then if π_T is α-unbiased with respect to $\mathcal{E}$, for every $u \in \mathcal{U}_L$, the straightforward decision maker with utility function u has type regret with respect to $\mathcal{U}_L$ at most:*

$$\max_{u'\in\mathcal{U}_L} r(\pi_T, u, u') \le \max_{u'\in\mathcal{U}_L} \frac{L\sum_{a\in\mathcal{A}}\left(\alpha(n_T(E_{u',a}, \pi_T)) + \alpha(n_T(E_{u,a}, \pi_T))\right)}{T}$$

For any concave α, this bound implies:

$$\max_{u'\in\mathcal{U}_L} r(\pi_T, u, u') \le \frac{2LK\alpha(T/K)}{T}$$

The proof of Theorem 30 can be found in Appendix J.

Appendix E. Application: Score-Free Prediction Sets with Anytime Transparent Coverage

Set-valued multiclass prediction Consider a multi-class prediction problem, in which examples are of the form $(x, y) \in \mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ is some feature space and $\mathcal{Y}$ is some finite (but possibly large) label space with $|\mathcal{Y}| = k$. A standard task in the area of distribution-free uncertainty quantification is to train a model $S : \mathcal{X} \to 2^{\mathcal{Y}}$ that, given features $x \in \mathcal{X}$, outputs a *prediction set* $S(x) \in 2^{\mathcal{Y}}$. The idea is that, in contrast to regular classification, here the trained model S need not map every feature vector $x \in \mathcal{X}$ to a single label in $\mathcal{Y}$: instead, it is at liberty to map any $x \in \mathcal{X}$ to a set consisting of multiple labels at once, thus indicating uncertainty about the prediction.

A central objective in this setting is to ensure that the prediction sets produced by the model *cover* (i.e. include) the true label y of a feature vector x with probability at least $1 - \alpha \in [0, 1]$ over the data. More formally, the *coverage probability* $\Pr_{(x,y)}[y \in S(x)]$ must be *no less than* (or, sometimes, approximately *equal to*) a certain prespecified *coverage level* $1 - \alpha \in [0, 1]$. We refer to prediction sets satisfying this coverage guarantee as $(1 - \alpha)$-*prediction sets*. Typical choices of α include 0.1 and 0.05, but other values may be appropriate depending on the domain (e.g., safety-critical applications such as autonomous driving may require a very small α).

A naive approach to building prediction sets For many multiclass prediction tasks, there exist readily available high-performance pretrained classifiers such as deep neural networks which, as an intermediate layer of their prediction mechanism, estimate a probability vector $\tilde{p}(x) \in \Delta\mathcal{Y}$ on any input $x \in \mathcal{X}$. Suppose we could take these probabilities at face value — i.e., trust that they closely approximate the true conditional label distribution $p(y|x)$. Then, one compelling way to build $(1-\alpha)$-prediction sets would be to:

1. Sort the probabilities $\tilde{p}(x)$ in nonincreasing order, such that each label $i \in [k]$ gets assigned a position $\sigma_{\text{sorted}}(i) \in [k]$ in this sorted order, and
2. Output the prediction set $S(x)$ consisting of the top k_x labels in this sorted order, where k_x is the smallest index such that the sum of predicted probabilities of all chosen labels is at least $1-\alpha$. That is,

$$S_{\text{sorted}}(x) := \left\{ i \in [k] \,\middle|\, \sigma_{\text{sorted}}(i) \in [1, k_x], \text{ where } k_x = \min_{k_{\text{thr}} \in \mathbb{N}} \sum_{i \in [k]: 1 \le \sigma_{\text{sorted}}(i) \le k_{\text{thr}}} (\tilde{p}(x))_i \ge 1-\alpha \right\}.$$

The sorting step in this procedure guarantees that if $\tilde{p}(x)$ in fact truly represented conditional label probabilities, this procedure would not only give the desired $1-\alpha$ coverage, but would in fact, on average, yield the *smallest* prediction sets with $1-\alpha$ coverage, as measured by the number of labels included in $S(x)$.

Unfortunately, the caveat here is that such probability scores $\tilde{p}(x)$, even those output by pretrained models with very low cross-entropy (or other) loss, are not guaranteed to even be close to true conditional probabilities. For this reason, the above procedure will generically fail to obtain its target coverage level.

Prediction sets with provable coverage guarantees The field of *conformal prediction*, which has seen substantial development in recent years (see [4] for a gentle introduction), provides one principled way to address the shortcomings of using predicted probabilities to form prediction sets. Namely, instead of using raw predicted probabilities $\tilde{p}(x)$, conformal prediction sets are formed with the help of a so-called *non-conformity score* function $s : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, which projects the otherwise high-dimensional problem of coming up with a prediction set onto a one-dimensional quantile estimation problem: For a new example x, a prediction set is chosen by selecting all labels y such that $s(x, y) \le \tau$, for some threshold $\tau \in \mathbb{R}$ which should ideally be a $1-\alpha$ quantile of the distribution on non-conformity scores over examples (x, y) in the calibration dataset. Most conformal prediction methods in both the batch and sequential setting boil down to estimating quantiles of this distribution in various ways.

Conformal prediction, in contrast to naive set prediction methods, offers *guaranteed* valid coverage: in other words, it provably forms $(1-\alpha)$-prediction sets, and is oblivious to the choice of score s or the underlying data distribution. A disadvantage of this approach is that the (average) prediction set size, and hence the informativeness and usefulness of the produced prediction sets in practice, *is* very sensitive to the choice of non-conformity score. For this reason, significant effort goes into designing custom non-conformity scores $s(x, y)$ for various applications, and such scores are usually not guaranteed to have any form of tight relationship with the true conditional probabilities $p(y|x)$.

Our contribution: Prediction sets with *anytime transparent* coverage In contrast to conformal prediction, in what follows we present an online algorithm for obtaining prediction sets with valid coverage that directly calibrates (high dimensional) raw predicted probability vectors, rather than working in a one-dimensional projection of the coverage problem. The probability forecasts we produce have a number of desirable properties:

1. *Transparent coverage:* Our predicted probability vectors can immediately be used by any downstream prediction set algorithm as if they were the true conditional probabilities — in particular, the naive sorting-based prediction set algorithm discussed above, when fed our probability vectors as input, will in hindsight, on actual data, obtain exactly (up to vanishing terms) the same (marginal or conditional) coverage that it would obtain if our probabilities were the true conditional class probabilities.

   This transparency property can be understood as a novel *interpretability*, or *trustworthiness*, guarantee for prediction-set-based uncertainty quantification: While existing conformal prediction methods rely on calibrating thresholds for nonconformity scores — which are not necessarily interpretable as, or guaranteed to look like, true conditional class probabilities — our method of producing raw predicted class probabilities bestows transparency upon any number of downstream prediction set algorithms, allowing them to produce valid-coverage sets by simply taking our probabilities at face value.

2. *Anytime coverage:* It is typical for online adversarial learning approaches to specify a fixed time horizon T upfront, such that both the learning rates and the final guarantees are a function of T and may only hold *upon reaching* the time horizon. In contrast, on any prespecified subsequence of data points of length t, we are able to attain statistically optimal $O\left(\frac{1}{\sqrt{t}}\right)$ coverage error rates, and we only need a very coarse upper bound on the maximum time horizon $T_{\max}$ as (1) our method's per-round running time at any round t does not depend on $T_{\max}$ (instead it depends on t itself), and (2) our coverage error bound only depend *logarithmically* on $T_{\max}$, i.e. is very insensitive to $T_{\max}$.[5] Thus, for example, we could pause the algorithm at any point $t \leq T_{\max}$, and then restart it when more data becomes available, and the transparency and coverage guarantees would continue to smoothly evolve from where we left off. A similar property holds for the one-dimensional adaptive conformal inference methods of [38, 90], but these methods do not accommodate conditional coverage constraints (or have the other desirable downstream properties of our method).

3. *Accommodates multiple diverse downstream tasks at once:* Our probability predictions not only help avoid the need to choose or commit to a non-conformity score, but allow us to make predictions that are *simultaneously* useful for a *variety* of downstream tasks. For example, we can predict probabilities that are useful for simultaneously producing prediction sets with different coverage levels (see Corollary 39) and/or aimed at optimizing different objectives.

4. *Achieves best-in-class prediction quality:* In our probability vector predictions, we are able to incorporate side information in the form of logits of any existing multiclass predictor: in fact, we show the stronger property that, given any finite collection $\mathcal{Q}$ of such external

5. See also Remark 15 on anytime coverage in Section C. Also, note that, while all our transparent coverage results are (almost) anytime, our Best-in-Class result, Theorem 54, does require a more time-horizon-sensitive setting of a discretization parameter.

predictors (which could, for instance, be pretrained NNs), we can increase the quality of our probability vector predictions to nearly match the performance of the logits of the best model in $\mathcal{Q}$, simultaneously as measured by Brier score, cross-entropy loss, and other well-behaved proper scoring rules.

5. *Enforces set-size-conditional validity, multigroup fairness, and other conditional coverage guarantees:* By making our predictions unbiased conditional on appropriate (modestly-sized) families of events, our probability predictions can easily guarantee various forms of conditional validity on the outputs of downstream prediction set algorithms. This includes not only strengthened multigroup fairness guarantees as compared to [7] (in terms of the dependence on the number of demographic groups $|\mathcal{G}|$), but also the highly desirable property of valid coverage conditional on the prediction set size (dubbed *size-stratified coverage validity* in [3]), which to our knowledge has not been provably obtained in the adversarial conformal prediction literature until now. More generally, we can efficiently obtain coverage conditional on *any* polynomially sized collection of events that are determined by any combination of external context and our own predictions.

We now provide (the most general form of) our algorithm, displayed below as Algorithm 5: `Class-Probabilities-for-Prediction-Sets`. We also informally summarize the statements of its diverse set of donwstream guarantees, which we just listed above. Before doing that, we give an informal definition that captures the behavior of *straightforward* prediction set algorithms whose aim is to achieve valid $(1-\alpha)$-coverage.

Definition (Informal) *[Prediction set algorithm]A* prediction set algorithm *S is a mapping*

$$S : \Pi^* \times \mathcal{X} \times \Delta\mathcal{Y} \to [0,1]^{|\mathcal{Y}|}.$$

At each round t, it takes in the past history, π_{t-1}, the new context x_t, our predicted class probability vector in this round, p_t, and outputs this round's prediction set $S(\pi_{t-1}, x_t, p_t)$ in the format of a vector in $[0,1]^{\mathcal{Y}}$, where the entry corresponding to each label $y \in \mathcal{Y}$, $(S(\pi_{t-1}, x_t, p_T)) \in [0,1]$, designates the probability with which the algorithm includes label y into its prediction set. (If the algorithm is deterministic, then each entry $(S(\pi_{t-1}, x_t, p_T)) \in \{0,1\}$.

This definition is reiterated, somewhat more formally, in Definition 31 below.

Definition (Informal) *[$(1-\alpha)$-aspiring prediction set algorithm]A downstream prediction set algorithm S is called* $(1-\alpha)$-aspiring *if, when fed any predicted probability vector $\tilde{p}(x) \in \Delta\mathcal{Y}$, it always outputs a prediction set that would have coverage $1-\alpha$ if $\tilde{p}(x)$ was the true conditional probability vector.*

This definition parallels Definition 9 of a straightforward decision maker and essentially adapts it to our uncertainty estimation setting: it captures prediction set algorithms that are naive, or straightforward, in that they are willing to *trust* the predicted class probabilities as if they were correct. For instance, the naive greedy sorting-based algorithm described above is (roughly speaking) $\geq (1-\alpha)$-aspiring. (The precise definition is given in Definition 37.)

Theorem (Informal) *[Guarantees for `Class-Probabilities-for-Prediction-Sets`]* *The algorithm `Class-Probabilities-for-Prediction-Sets` guarantees, With any collection of downstream prediction set algorithms $\mathcal{S}$ as input and on any sequence of potentially adversarially generated data, that for every $S \in \mathcal{S}$:*

- *The realized* per-label coverage *of S for each label $y \in \mathcal{Y}$ (the online analog of the batch probability $\Pr_{X,Y}[Y = y \wedge y \in S(X)]$) will be equal, up to a diminishing error, to the* anticipated *per-label coverage for y (i.e., the coverage level on y that S would expect to obtain, based on treating our predicted probabilities at each round as correct). See Theorem 35;*
- *Similarly, the realized marginal coverage of S (the analog of the batch probability $\Pr_{X,Y}[Y \in S(X)]$) will be equal, up to a diminishing error, to the* anticipated *marginal coverage S would obtain if our class probabilities had been correct. See Corollary 36;*
  - *For any downstream prediction set algorithm that is $(1-\alpha)$-aspiring, this implies valid $(1-\alpha)$ realized* marginal *coverage; see Corollary 38.*
- *(set-size-conditional? $=$ Y) Given as input any* set-size function *defined as*[6] $\text{sz} : [0,1]^{|\mathcal{Y}|} \to \{0\} \cup [N_{\text{maxsz}}]$ *for some maximum set size N_{maxsz}, the anticipated and realized coverage will coincide, up to error terms, even conditionally on the prediction set size. See Section E.2.1;*
  - *For any downstream prediction set algorithm S that is $(1-\alpha)$-aspiring, this implies valid realized* set-size-conditional *$(1-\alpha)$ coverage.*
- *(multigroup-fair? $=$ Y) Given as input any family $\mathcal{G}$ of* demographic groups *$G : \mathcal{X} \to [0,1]$ — which for any individual with features x determine if they belong ($G(x) = 1$), do not belong ($G(x) = 0$), or belong partially ($G(x) \in (0,1)$) to a certain (often normatively defined) population group — the anticipated and realized coverage will coincide, up to error terms, conditionally on every group $G \in \mathcal{G}$. See Section E.2.1;*
  - *For any downstream prediction set algorithm S that is $(1-\alpha)$-aspiring, this implies valid realized $(1-\alpha)$* multigroup *coverage.*
- *(booster? $=$ Y) Generalizing set-size-conditional and the multigroup coverage, it can be guaranteed for every family $\mathcal{B}$ of conditional constraints defined by subsequence indicators $B : \Pi^* \times \mathcal{X} \times \Delta\mathcal{Y} \to [0,1]$ (which we call 'boosters') that decide whether or not to invoke the conditional constraint given any transcript π_{t-1}, context x_t, and prediction p_t. See Section E.2.1;*
  - *For any downstream prediction set algorithm S that is $(1-\alpha)$-aspiring, this implies valid realized $(1-\alpha)$* conditional *coverage with respect to the family of conditioning events given by the booster mappings.*
- *(best-in-class? $=$ Y) Given as input any collection of competing class probability predictors $\mathcal{Q}$ — where each predictor $q \in \mathcal{Q}$ outputs, at each round t, a predicted class probability vector $q_t \in \Delta\mathcal{Y}$ as a function of the context x_t (and possibly also as a function of past history and even of our class probability predictions in the current round) — our predictions will*

6. More generally, sz can be any bounded mapping from $[0,1]^{|\mathcal{Y}|}$ to $[0,\infty)$; in that case, we can simply discretize it.

be guaranteed to beat all *predictors* $q \in \mathcal{Q}$ *simultaneously with respect to* all L*-Lipschitz Bregman losses, up to diminishing gap that scales with* L*. (The result also extends to* locally *Lipschitz Bregman losses.) See Theorem 54 and Corollary 56.*

Moreover, at each round $t = 1, 2, \ldots$*,* `Class-Probabilities-for-Prediction-Sets` *has runtime polynomial in* t*,* $|\mathcal{Y}|$ *and in the size* $|\mathcal{E}^t|$ *of the aggregated event collection in that round.*

```
INPUT: Collection of downstream prediction set algorithms $\mathcal{S}$,
        set-size-conditional? $\in$ {Y, N},   set size function sz : $[0,1]^{|\mathcal{Y}|} \to \{0\} \cup [N_{\text{maxsz}}]$,
        multigroup-fair? $\in$ {Y, N},        collection of protected groups $\mathcal{G}$,
        booster? $\in$ {Y, N},                collection of booster predictors $\mathcal{B}$,
        best-in-class? $\in$ {Y, N},          collection of competing predictors $\mathcal{Q}$
for rounds $t = 1, 2, \ldots$ do
    Receive context $x_t \in \mathcal{X}$
    Assemble event collection $\mathcal{E}^t$:
    For every $S \in \mathcal{S}$, form: $\mathcal{E}^t_S := \{E_{S,y}\}_{y \in \mathcal{Y}}$, with $E_{S,y}(p) := (S(\pi_{t-1}, x_t, p))_y$ for $p \in \Delta\mathcal{Y}$
    $\mathcal{E}^t_{\mathcal{S}} \leftarrow \cup_{S \in \mathcal{S}} \mathcal{E}^t_S$
    $\mathcal{E}^t \leftarrow \mathcal{E}^t_{\mathcal{S}}$ // transparent per-label coverage
    if set-size-conditional? $= Y$ then
        For all $S \in \mathcal{S}$, form: $\mathcal{E}^t_{\text{sz,S}} \leftarrow \{E_{\text{sz},S,n}\}_{0 \le n \le N_{\text{maxsz}}}$, with $E_{\text{sz},S,n}(p) := 1[\text{sz}(S(\pi_{t-1}, x_t, p)) = n]$ for all $p$
        $\mathcal{E}^t \leftarrow \mathcal{E}^t \cup \left(\bigcup_{S \in \mathcal{S}} \mathcal{E}^t_S \times \mathcal{E}^t_{\text{sz,S}}\right)$
    if multigroup-fair? $= Y$ then
        $\mathcal{E}^t_{\mathcal{G}} \leftarrow \{E_G\}_{G \in \mathcal{G}}$, where $E_G := G(x_t)$ ;   // each $G$ stands for a context-defined demographic group
        $\mathcal{E}^t \leftarrow \mathcal{E}^t \cup \mathcal{E}^t_{\mathcal{S}} \times \mathcal{E}^t_{\mathcal{G}}$
    if booster? $= Y$ then
        $\mathcal{E}^t_{\mathcal{B}} \leftarrow \{E_B\}_{B \in \mathcal{B}}$, where $E_B(p) := B(\pi_{t-1}, x_t, p)$ for $p \in \Delta\mathcal{Y}$ ;   // boosters find miscovered regions
        $\mathcal{E}^t \leftarrow \mathcal{E}^t \cup \mathcal{E}^t_{\mathcal{S}} \times \mathcal{E}^t_{\mathcal{B}}$
    if best-in-class? $= Y$ then
        $\mathcal{E}^t_{\text{ls},p} \leftarrow \{E_{p,n,i}\}_{n \in \{0\} \cup [\lceil 1/\delta \rceil], i \in [k]}$, where $E_{p,n,i}(p) := 1[p_{t,i} \approx n\delta]$ for $p \in \Delta\mathcal{Y}$ ; // $\delta$-level sets of our predictor; see Definition 53, and see Theorem 54 for the appropriate setting of $\delta$
        $\mathcal{E}^t_{\text{ls},\mathcal{Q}} \leftarrow \{E_{q,n,i}\}_{q \in \mathcal{Q}, n \in \{0\} \cup [\lceil 1/\delta \rceil], i \in [k]}$, where $E_{q,n,i}(p) := 1[(q(\pi_{t-1}, p))_i \approx n\delta]$ for $p \in \Delta\mathcal{Y}$ ;   // $\delta$-level sets of all competing predictors $q \in \mathcal{Q}$
        $\mathcal{E}^t \leftarrow \mathcal{E}^t \cup \mathcal{E}^t_{\text{ls},p} \times \mathcal{E}^t_{\text{ls},\mathcal{Q}}$
    Predict $P_t \leftarrow$ UnbiasedPrediction($\mathcal{E}^t, t, \pi_{t-1}, x_t$) ;   // $P_t \in \Delta\Delta\mathcal{Y}$ is a mixture over probability vectors
    Sample realized predicted probability vector $p_t \sim P_t$
    Receive correct label $y_t \in \mathcal{Y}$ from adversary
    Update transcript: $\pi_t \leftarrow \pi_{t-1} \cup (x_t, p_t, y_t)$
```

Algorithm 5: `Class-Probabilities-for-Prediction-Sets`

E.1. Online Adversarial Multiclass Set Prediction Preliminaries

Setting We let $\mathcal{X}$ be the feature space and $\mathcal{Y}$ be the label space with $|\mathcal{Y}| = k$. When it is notationally convenient for indexing purposes, we without loss of generality assume $\mathcal{Y} = \{1, \dots, k\}$. Examples arrive—and predictions about them are made—in a sequential fashion, in rounds $t = 1, 2, \dots$. Specifically, in each round t:

1. A new example arrives, and its features $x_t \in \mathcal{X}$ are revealed to the learner.
2. The learner computes a (realized) probability prediction $p_t \in \Delta\mathcal{Y}$ as a function of past rounds' history and the observed features x_t.
3. The (realized) prediction set $S_t \subseteq \mathcal{Y}$ is generated as a function of p_t.
4. The realized label $y_t \in \mathcal{Y}$ is revealed by the adversary.

Prediction set algorithms We now define the notion of a prediction set algorithm, which is responsible for forming prediction sets S_t in all rounds t as a function of past history, the current context, and our (the learner's) predicted multiclass probability vector. Intuitively, this algorithm may be viewed as representing a downstream decision maker who takes in our probability predictions and builds a prediction set so as to optimize whichever utility function (e.g., a notion of average prediction set size) it has.

Definition 31 (Prediction set algorithm) *A* prediction set algorithm S *is a mapping*

$$S : \Pi^* \times \mathcal{X} \times \Delta\mathcal{Y} \to [0, 1]^{|\mathcal{Y}|}.$$

The first input to S is the transcript π_{t-1} up until the current round t, thus allowing the algorithm's decisions to depend on past history. The second input takes in the features of the example at round t. The third input captures the dependence of the algorithm's decisions in round t on our predicted probability vector $p_t \in \Delta\mathcal{Y}$.

For any π_{t-1}, x and p_t, the algorithm's output $S(\pi_{t-1}, x, p_t)$ is a $|\mathcal{Y}|$-dimensional vector; by convention here and below, we will index it by $y \in \mathcal{Y}$. The interpretation of the algorithm's output is that for any $y \in \mathcal{Y}$, the algorithm S will include label y in the realized prediction set S_t with probability $(S(\pi_{t-1}, x, p_t))_y \in [0, 1]$ over its internal randomness. Note that the entries of $S(\pi_{t-1}, x, p_t)$ will generally not sum to one: since S outputs *prediction sets* rather than singular labels, the output S_t is allowed to contain multiple labels at once.

As an important special case of the above definition, a *deterministic* prediction set algorithm is defined as a *binary* vector valued mapping

$$S : \Pi^* \times \mathcal{X} \times \Delta\mathcal{Y} \to \{0, 1\}^{|\mathcal{Y}|}.$$

A prediction set algorithm that is not deterministic is referred to as randomized.

When convenient, for any label y, instead of $(S(t, x, p_t))_y$ we may alternatively write $\Pr[y \in S_t]$ when S is randomized and $\mathbf{1}[y \in S_t]$ when S is deterministic, suppressing the dependence on π_{t-1}, x_t and p_t.

Coverage metrics We now define the notions of (marginal and per-label) *realized* and *anticipated* coverage of a prediction set algorithm on any transcript. As discussed above, a natural desideratum in this setting is to ensure that the realized coverage of the prediction set algorithm meets some prespecified target level.

Definition 32 *The* realized (marginal) coverage *of a prediction set algorithm S on a T-round transcript π_T is denoted:*

$$\overline{\Pr}_T[Y \in S] := \frac{1}{T}\sum_{t=1}^{T} \Pr[y_t \in S_t],$$

where y_t, for $t \in [T]$, are the realized labels.

Note that this is simply the *empirical probability* of the realized labels belonging to the prediction sets produced by the algorithm.

Definition 33 *The* anticipated (marginal) coverage *of a prediction set algorithm S on a T-round transcript π_T is denoted:*

$$\widetilde{\Pr}_T[Y \in S] := \frac{1}{T}\sum_{t=1}^{T}\sum_{y\in\mathcal{Y}} p_{t,y} \cdot \Pr[y \in S_t],$$

where $p_{t,y}$ are the learner's predictions. Note that the anticipated coverage does not depend on the actual realized labels y_t.

The interpretation is simple: If our predicted probability vector $p_{t,y}$ were to be taken at face value, i.e. assuming that the labels y_t are sampled from the distribution p_t, then $\widetilde{\Pr}_T[Y \in S]$ would be exactly the expected coverage that the algorithm would get after round T, where the expectation is taken both over the randomness of S and the randomness of the labels y_t being sampled from the distributions p_t. Note that to evaluate the algorithm's anticipated coverage $\widetilde{\Pr}_T[Y \in S]$, one needs to know our predictions p_t, but *not* the realized labels.

Definition 34 (Per-label realized and anticipated coverage) *For any label $y \in \mathcal{Y}$, we define the* realized *and* anticipated *coverage that a prediction set algorithm S achieves* on label y alone *over a T-round transcript π_T, analogously to the marginal coverage metrics defined above:*

$$\overline{\Pr}_T[y \in S] := \frac{1}{T}\sum_{t=1}^{T} 1[y = y_t] \cdot \Pr[y \in S_t], \quad \textit{and}\ \widetilde{\Pr}_T[y \in S] := \frac{1}{T}\sum_{t=1}^{T} p_{t,y} \cdot \Pr[y \in S_t].$$

The just defined notion of per-label coverage will serve as a key stepping stone towards showing our transparent marginal coverage guarantee for any downstream algorithm S. Namely, we will (in Theorem 35) show the stronger guarantee of *transparent per-label coverage* for any S. For intuition, it is instructive to compare *per-label* coverage, which we just introduced, to *label-conditional* coverage, a common object of study in the area of conditional conformal coverage guarantees ([87]). In the batch setting, the former would correspond to $\Pr_{(X,Y)}[Y = y \wedge y \in S(X)]$, whereas the latter would translate to $\Pr_{(X,Y)}[Y \in S(X) | Y = y]$. Thus, per-label coverage is a joint probability (of two events coinciding: label y being selected as the correct label, and it being included in the prediction set by S), while label-conditional coverage is the corresponding conditional probability.

We observe that for any prediction set algorithm S, its time-averaged (anticipated or realized) marginal coverage is simply the sum over all labels $y \in \mathcal{Y}$ of its label-specific (anticipated or realized) coverage values:

$$\overline{\Pr}_T[Y \in S] = \sum_{y \in \mathcal{Y}} \overline{\Pr}_T[y \in S], \quad \text{and } \widetilde{\Pr}_T[Y \in S] = \sum_{y \in \mathcal{Y}} \widetilde{\Pr}_T[y \in S]. \tag{7}$$

This observation will later be useful in translating our per-label transparent coverage guarantees (Theorem 35) into marginal transparent coverage guarantees (Corollary 36).

E.2. Anytime Transparent Coverage: Marginal, Per-Label, and Conditional

In this section, we show how to apply our `UnbiasedPrediction` algorithm in order to produce predictions of per-label probabilities $p_t \in \Delta\mathcal{Y}$ that provide transparent and flexible coverage guarantees for any downstream algorithm S that produces prediction sets $S_t := S(\pi_{t-1} \cup \{x_t\}, p_t)$ as a function of past history and our current predictions.

Intuitively, since our predictions are to be used by S to compute a prediction set, we would like to make them unbiased conditional on the collection of events, for each label $y \in \mathcal{Y}$, that y is included by S in the prediction set once S sees our predicted probabilities. (This may remind the reader of the "regret" constructions in the prior sections, where we condition on the collection of events defined by the downstream decision maker's best-response correspondence.)

Thus, we instantiate `UnbiasedPrediction` on the region $C_{\text{multiclass}} = \Delta\mathcal{Y} \subset \mathbb{R}^k$ with a natural collection of k events: $\mathcal{E}_S := \{E_{S,y}\}_{y \in \mathcal{Y}}$, where each event $E_{S,y}$ is defined by $E_{S,y}(\pi_{t-1}, x_t, p_t) = (S(\pi_{t-1} \cup \{x_t\}, p_t))_y$, encoding the probabilities of inclusion of each fixed label y into the prediction sets S_t across rounds t.

For instance, when the prediction set algorithm S is deterministic, all events $E_{S,y}$ are binary, and simply represent the indicator of whether or not, given the predictions p_t made on that round, the label y is included in the prediction set S_t.

In fact, as we will show in Theorem 35 below, we can achieve transparent coverage simultaneously for multiple downstream algorithms S belonging to any finite collection $\mathcal{S}$. It will suffice to instantiate `UnbiasedPrediction` with $\mathcal{E}_\mathcal{S} = \cup_{S \in \mathcal{S}} \mathcal{E}_S$ the *union* of event families $\mathcal{E}_S$ over all algorithms $S \in \mathcal{S}$. As a result, the runtime of our method will only scale linearly in $|\mathcal{S}|$, while the coverage error bound will only pick up a $\log |\mathcal{S}|$ dependence.

On the dependence of `Class-Probabilities-for-Prediction-Sets` on input prediction set algorithms S: Our algorithm only makes *oracle queries* to the downstream prediction set algorithm S at each round t. It completely disregards the internals of S, and only needs to know (for a deterministic algorithm) which labels the algorithm includes in its prediction set as a function of the underlying probability forecast p_t, or (for a randomized algorithm) the probability that each label is included in its prediction set.

In this manner, `Class-Probabilities-for-Prediction-Sets` functions as a wrapper around any prediction set algorithm S (or even a collection of algorithms $\mathcal{S}$). Given the computational efficiency guarantees for `UnbiasedPrediction` established in Theorem 14, the "wrapped" prediction algorithm will thus be computationally efficient insofar as the downstream prediction algorithm S is efficient and $\mathcal{Y}$ is enumerable — i.e. the running time will be polynomial in $|\mathcal{Y}|$ and the running time of S.

Transparent coverage guarantees: We now show that Algorithm 5 guarantees, no matter the downstream prediction set algorithm(s), that the realized and anticipated coverage for every label $y \in \mathcal{Y}$ will coincide, up to $O(1/\sqrt{T})$ — or better — error terms.

Theorem 35 (Transparent Per-Label Coverage) *Fix any collection $\mathcal{S}$ of prediction set algorithms, where each algorithm $S \in \mathcal{S}$ uses our method's probability vector predictions $p_t \in \Delta\mathcal{Y}$ in every round t. If we instantiate Algorithm 5 with the event collection*

$$\mathcal{E}_{\mathcal{S}} := \{E_{S,y}\}_{S \in \mathcal{S}, y \in \mathcal{Y}}, \text{ with } E_{S,y}(\pi_{t-1}, x_t, p) := (S(\pi_{t-1} \cup \{x_t\}, p))_y \text{ for } p \in \Delta\mathcal{Y}, \tag{8}$$

then in expectation over $P_1, P_2, \ldots \in \Delta\Delta\mathcal{Y}$, the distributions Algorithm 5 outputs, the anticipated *and* realized *coverage of every label $y \in \mathcal{Y}$ by every prediction set algorithm $S \in \mathcal{S}$ will be approximately equal at every round $T \in [T_{\max}]$ (where $T_{\max}$ is the maximum time horizon):*

$$\mathop{\mathbb{E}}_{p_t \sim P_t, \forall t}\left[\left|\overline{\Pr}_T[y \in S] - \widetilde{\Pr}_T[y \in S]\right|\right] \leq O\left(\frac{\ln(k|\mathcal{S}|T_{\max}) + \sqrt{\ln(k|\mathcal{S}|T_{\max}) \cdot \sum_{t=1}^T \mathbb{E}_{p_t \sim P_t}\left[\Pr[y \in S_t]\right]}}{T}\right).$$

Proof Let $\mathcal{C}_{\text{multiclass}} := \Delta\mathcal{Y} \subset \mathbb{R}^k$. Suppose in each round, we predict and receive realizations from the space $\mathcal{C}_{\text{multiclass}}$, where at each round t, the adversary picks a vector in $\{\text{standard basis vectors in } \mathbb{R}^k\} \subset \mathcal{C}_{\text{multiclass}}$, and is interpreted as $(1[y = y_t])_{y \in \mathcal{Y}}$ — the vector that indicates which label $y \in \mathcal{Y}$ was realized in round t.

Accordingly, we abuse notation and index the coordinates of our *predicted* vectors p_t by $y \in \mathcal{Y}$ as well. In the current multiclass setting, we interpret our chosen predicted vector p_t as a distribution over the labels.

We now instantiate Theorem 14 on the space $\mathcal{C}_{\text{multiclass}}$ with the collection of events $\mathcal{E}_{\mathcal{S}}$ defined as in Equation 8. For each $S \in \mathcal{S}$ and $y \in \mathcal{Y}$, the event $E_{S,y}$ is defined simply as the probability in round t that S would include label y into the prediction set, given features x_t and our prediction p_t. In other words, we have, over the internal randomness of the set selection algorithm S given the current round's inputs, that

$$E_{S,y}(\pi_{t-1}, x_t, p_t) = \Pr[y \in S_t].$$

Abusing notation and indexing the coordinates of the predicted vector p_t and the realized vector y_t by $y \in \mathcal{Y}$ rather than $i \in [k]$, we then have, for all $E_{S,y} \in \mathcal{E}_{\mathcal{S}}$, any T, some constant $c' > 0$, and per-round distributions ψ_t over the predictions output by our unbiased prediction algorithm, that

$$\mathop{\mathbb{E}}_{p_t \sim \psi_t \, \forall t}\left|\sum_{t=1}^T E_{S,y}(x_t, p_t) \cdot (p_{t,y} - 1[y = y_t])\right| \leq c' \ln(2k^2|\mathcal{S}|T_{\max}) + c' \sqrt{\ln(2k^2|\mathcal{S}|T_{\max}) \cdot \sum_{t \in [T]} \mathop{\mathbb{E}}_{p_t \sim \psi_t}[(E_{S,y}(x_t, p_t))^2]}.$$

Here, we used that $d = |\mathcal{E}_{\mathcal{S}}| = k|\mathcal{S}|$. Now, rename the distributions over predictions from ψ_t into P_t. Dropping the squares on the events inside the square root (since our events' values are in $[0,1]$) and observing that

$$\sum_{t=1}^T E_{S,y}(t, x_t, p_t) \cdot (p_{t,y} - 1[y = y_t]) = \sum_{t=1}^T \Pr[y \in S_t] \cdot p_{t,y} - \sum_{t=1}^T \Pr[y \in S_t] \cdot 1[y = y_t] = T(\overline{\Pr}_T[y \in S] - \widetilde{\Pr}_T[y \in S]),$$

we obtain, for every prediction set algorithm $S \in \mathcal{S}$ and label $y \in \mathcal{Y}$, that

$$\mathop{\mathbb{E}}_{p_t \sim P_t \, \forall t}\left[T \cdot \left|\overline{\Pr}_T[y \in S] - \widetilde{\Pr}_T[y \in S]\right|\right] \leq c' \ln(2k^2|\mathcal{S}|T_{\max}) + c' \sqrt{\ln(2k^2|\mathcal{S}|T_{\max}) \cdot \sum_{t \in [T]} \mathop{\mathbb{E}}_{p_t \sim P_t}[\Pr[y \in S_t]]},$$

and dividing through by T yields the desired claim. ■

Recalling that marginal (realized or anticipated) coverage is just the sum of per-label (realized or anticipated) empirical coverage probabilities over all labels, and applying the triangle inequality, we thus obtain, as a direct consequence of Theorem 35, the corresponding *marginal* coverage guarantees for the instantiation of `Class-Probabilities-for-Prediction-Sets` with the same event family $\mathcal{E}_{\mathcal{S}}$:

Corollary 36 (Transparent Marginal Coverage) *Fix any collection $\mathcal{S}$ of prediction set algorithms, where each algorithm $S \in \mathcal{S}$ uses our method's probability vector predictions $p_t \in \Delta\mathcal{Y}$ in every round t. If we instantiate Algorithm 5 with the event collection as in Equation 8, i.e.:*

$$\mathcal{E}_{\mathcal{S}} := \{E_{S,y}\}_{S \in \mathcal{S}, y \in \mathcal{Y}}, \text{ with } E_{S,y}(\pi_{t-1}, x_t, p) := (S(\pi_{t-1} \cup \{x_t\}, p))_y \text{ for } p \in \Delta\mathcal{Y},$$

then in expectation over $P_1, P_2, \ldots \in \Delta\Delta\mathcal{Y}$, the distributions Algorithm 5 outputs, the anticipated *and* realized *marginal coverage of every prediction set algorithm $S \in \mathcal{S}$ will be approximately equal at every round $T \in [T_{\max}]$ (where $T_{\max}$ is the maximum time horizon):*

$$\mathop{\mathbb{E}}_{p_t \sim P_t, \forall t}\left[\left|\overline{\Pr}_T[Y \in S] - \widetilde{\Pr}_T[Y \in S]\right|\right]$$
$$\leq O\left(\frac{|\mathcal{Y}| \ln(|\mathcal{Y}||\mathcal{S}|T_{\max})}{T} + \frac{1}{T}\sum_{y \in \mathcal{Y}} \sqrt{\ln(|\mathcal{Y}||\mathcal{S}|T_{\max}) \cdot \sum_{t \in [T]} \mathop{\mathbb{E}}_{p_t \sim P_t}[\Pr[y \in S_t]]}\right) \leq \tilde{O}\left(\frac{|\mathcal{Y}| \ln(|\mathcal{Y}||\mathcal{S}|T_{\max})}{\sqrt{T}}\right).$$

Application: Interpretable Conformal-Style Valid Coverage without Nonconformity Scores We now spell out a sample application of `Class-Probabilities-for-Prediction-Sets` instantiated with the collection of events $\mathcal{E} = \mathcal{E}_{\mathcal{S}}$ for some family of downstream prediction set algorithms $\mathcal{S}$, to illustrate how our method can easily produce $(1-\alpha)$-prediction sets, thereby achieving the main desideratum of e.g. conformal prediction. We begin with a definition:

Definition 37 ($(1-\alpha)$-Aspiring Prediction Set Algorithm) *Fix $\alpha \in [0, 1]$. A prediction set algorithm $S : \Pi^* \times \mathcal{X} \times \Delta\mathcal{Y} \to [0, 1]^{|\mathcal{Y}|}$ is called* $(1-\alpha)$-aspiring *if for all $\pi \in \Pi^*. x \in \mathcal{X}, p = (p_y)_{y \in \mathcal{Y}} \in \Delta\mathcal{Y}$:*

$$\sum_{y \in \mathcal{Y}} p_y \cdot (S(\pi, x, p))_y = 1 - \alpha.$$

Importantly, the property of an algorithm S being $(1-\alpha)$-aspiring guarantees that, when it uses our predicted class probabilities in each round t, its *anticipated marginal coverage* is equal to $1-\alpha$ — and not just on average over all rounds $t \in [T]$, but also *on any given round* t.

Perhaps the simplest example of an aspiring prediction set algorithm S is a fractional variant of the naive algorithm discussed at the beginning of this section:

1. Given a predicted probability vector p, order labels $y \in \mathcal{Y}$ in decreasing order of p_y (ignoring transcript π and features x).
2. Start with an empty prediction set, and keep adding labels to it in decreasing order of p_y, until the anticipated coverage (i.e., the running sum of predicted probabilities) has become $\geq 1-\alpha$.
3. If the final running sum of p_y has overshoot $1-\alpha$, the last included label y_{last} should be included "fractionally", i.e., $(S(\pi, x, p))_{y_{\text{last}}} \in (0,1)$, to satisfy *exactly* $1-\alpha$ anticipated coverage.

Observe that at every round T, the anticipated coverage of any $(1-\alpha)$-aspiring algorithm S satisfies:

$$\widetilde{\Pr}_T[Y \in S] = \frac{1}{T}\sum_{t=1}^{T}\sum_{y\in\mathcal{Y}} p_{t,y} \cdot \Pr[y \in S_t] = \frac{1}{T}\sum_{t=1}^{T}(1-\alpha) = 1-\alpha.$$

Consequently, Corollary 36 implies the following about $(1-\alpha)$-aspiring algorithms that use our predictions:

Corollary 38 (Valid realized marginal coverage for $(1-\alpha)$-aspiring prediction set algorithms) *For any family $\mathcal{S}$ of $(1-\alpha)$-aspiring prediction set algorithms, instantiating Algorithm 5 with the collection of events $\mathcal{E} = \mathcal{E}_{\mathcal{S}}$ guarantees $1-\alpha$ marginal coverage* to all algorithms $S \in \mathcal{S}$ simultaneously*, at all times $T \leq T_{\max}$, with deviation from exact $1-\alpha$ coverage decaying as (or better than) $\tilde{O}\left(\frac{|\mathcal{Y}|\ln(|\mathcal{Y}||\mathcal{S}|T_{\max})}{\sqrt{T}}\right)$.*

When might it make sense to provide appropriately unbiased class probabilities to ensure valid realized coverage for many downstream prediction set algorithms with the same coverage target $1-\alpha$? Note that there are generally many ways to select a prediction set over $\mathcal{Y}$ while achieving the same $(1-\alpha)$ anticipated coverage — the difference between such different ways to form a $(1-\alpha)$-prediction set may come down, e.g., to the difference in the *objective functions* (downstream from the prediction set algorithm) that these selections optimize. Thus, one way to interpret this result is that it provides valid $(1-\alpha)$-coverage in the face of uncertainty over which one of a finite class of downstream objectives the prediction sets will be used to optimize for.

In fact, our method does not need the algorithms in the collection $\mathcal{S}$ to all aspire to the same coverage level $1-\alpha$; they can each have their individual target coverage levels, and our guarantees will still hold:

Corollary 39 (Valid realized marginal coverage at different target levels) *Consider any collection $\mathcal{S} = \{S_1, S_2, \ldots, S_N\}$ of downstream prediction set algorithms where each S_i is $(1-\alpha_i)$-aspiring — i.e., different algorithms in the collection are seeking different coverage levels. Then, by instantiating our `Class-Probabilities-for-Prediction-Sets` method with the collection of events $\mathcal{E} = \mathcal{E}_{\mathcal{S}}$, we guarantee valid realized target $1-\alpha_i$ coverage* to each $S_i \in \mathcal{S}$ simultaneously*, at all times $T \leq T_{\max}$, with deviation from exact $1-\alpha_i$ coverage decaying as (or better than) $\tilde{O}\left(\frac{|\mathcal{Y}|\ln(|\mathcal{Y}||\mathcal{S}|T_{\max})}{\sqrt{T}}\right)$.*

The probability vector predictions issued by `Class-Probabilities-for-Prediction-Sets` under this instantiation will thus have the interesting property of being *simultaneously valid at different coverage levels*.

E.2.1. TRANSPARENT CONDITIONAL COVERAGE FOR SET-SIZE-CONDITIONAL AND MULTIGROUP-FAIR VALIDITY

Thus far we have shown how to make probabilistic predictions that have transparent coverage guarantees. But this is not the same thing as making high quality predictions. Can we guarantee that our predictions are somehow "good" while still offering the kinds of attractive coverage guarantees we have just given?

In general, in an adversarial setting, there is no way to guarantee that our predictions are "good" in an absolute sense, but we can make sure that they satisfy various consistency conditions. For set-valued predictions, a natural goal for us will be to ask that such consistency conditions translate, for downstream predictors using our predictions, into *conditional coverage guarantees*, a subject of intensive study in distribution-free uncertainty quantification at least since [87]. At a high level, conditionally valid coverage guarantees help make sure that target $(1-\alpha)$-coverage was *not* simply achieved by overcovering on some subsets of the data and undercovering on others.

We will now formalize the observation that our framework allows us to layer in various kinds of probabilistic consistency constraints: we can add events to the collection $\mathcal{E}$ that will ensure that our predictions p_t entail transparent coverage guarantees over any — possibly weighted — subsequences of interest in the data.

Boosting a model's *conditional* coverage transparency and validity We begin by defining *conditional* realized and anticipated coverage for a prediction set algorithm S (and their per-label variants); this will directly parallel the marginal coverage Definitions 33, 32, and 34. We define them to be conditional on any (nonempty) weighted subsequence of rounds: recall that a *weighted subsequence* W is a mapping $W : \mathbb{N} \to [0,1]$, where $W(t)$ is the weight with which round t belongs to the subsequence. For unweighted subsequences, i.e., $W : \mathbb{N} \to \{0,1\}$, $W(t) = 1$ means that round t was included in the subsequence, and vice versa for $W(t) = 0$.

Definition 40 (Conditional Coverage Metrics) *Fix a prediction set algorithm S, a T-round transcript π_T, and a weighted subsequence $W : \mathbb{N} \to [0,1]$. Let $(y_t)_{t\in[T]}$ be the realized labels and $(p_{t,y})_{t\in[T],y\in\mathcal{Y}}$ be our realized predictions over the transcript π_T.*

The realized *and* anticipated coverage *of S* conditional *on W are given by:*

$$\overline{\Pr}_T[Y \in S \,|\, W] := \frac{\sum_{t=1}^T W(t) \cdot \Pr[y_t \in S_t]}{\sum_{t\in[T]} W(t)}, \quad \textit{and } \widetilde{\Pr}_T[Y \in S \,|\, W] := \frac{\sum_{t=1}^T W(t) \sum_{y\in\mathcal{Y}} p_{t,y} \cdot \Pr[y \in S_t]}{\sum_{t\in[T]} W(t)}.$$

The realized and anticipated per-label coverage *of S* conditional *on W for any label $y \in \mathcal{Y}$ are given by:*

$$\overline{\Pr}_T[y \in S \,|\, W] := \frac{\sum_{t=1}^T W(t) \cdot 1[y = y_t] \cdot \Pr[y \in S_t]}{\sum_{t\in[T]} W(t)}, \quad \textit{and } \widetilde{\Pr}_T[y \in S \,|\, W] := \frac{\sum_{t=1}^T W(t) \cdot p_{t,y} \cdot \Pr[y \in S_t]}{\sum_{t\in[T]} W(t)}.$$

Clearly, it would be ideal to have our transparent coverage guarantees hold conditional on *all* subsequences. Since this is impossible, we must restrict attention to offering such guarantees conditional on all *relevant* subsequences in the data. Deciding which subsequences are the relevant ones can be done based either on trying to correct for past failures of the model (i.e., over- or undercoverage), or based on normative considerations (e.g., if some regions in the data must be paid special attention due to them corresponding to protected group of points/individuals).

We will address normative scenarios shortly; for now, we assume that our goal is to form subsequences in a data-driven way in an attempt to find and correct for persistent over- or undercoverage of the model on certain regions of the dataset, thus *boosting* its coverage performance. This is often best relegated to side models which are trained to perform such coverage checks, detecting any over- or underconfidence regions based on past history. We will refer to such models as *booster models*. For our purposes, we will abstract away any internal details of booster models, and instead define a booster through the mapping from transcripts, contexts, and predictions to a dynamically evolving subsequence of rounds.

Definition 41 (Booster) *A* booster *is any mapping of the form $B : \Pi^* \times \mathcal{X} \times \Delta\mathcal{Y} \to [0, 1]$, where for any round t, we interpret $B(\pi_{t-1}, x_t, p_t) \in [0, 1]$ as the booster model's decision to include, not include, or partially include the current round t in its generated subsequence of rounds.*

Note that the booster B naturally corresponds to a weighted subsequence of rounds*, so we will denote anticipated and realized coverage conditional on its generated subsequence by $\overline{\Pr}_T[Y \in S \,|\, B], \widetilde{\Pr}_T[Y \in S \,|\, B], \overline{\Pr}_T[y \in S \,|\, B], \widetilde{\Pr}_T[y \in S \,|\, B]$.*

We now provide our most general transparent conditional coverage guarantee, which generalizes the marginal guarantees of Theorem 35 and Corollary 36. The proof proceeds in much the same way as before, with the main difference that we now use an appropriately expanded family of conditioning events to instantiate `UnbiasedPrediction`, as well as that our guarantees for each booster now depend on its cumulative incidence $\sum_{t\in[T]} B(\pi_{t-1}, x_t, p_t)$, rather than on T. (Note: below, we may simply write $B(x_t, p_t)$, omitting the transcript.)

Theorem 42 (Conditional Transparent Coverage for Arbitrary Booster Collection) *Fix a collection $\mathcal{S}$ of prediction set algorithms and a collection $\mathcal{B}$ of boosters. If Algorithm 5 is instantiated with the event collection $\mathcal{E} := \mathcal{E}_{\mathcal{S}} \times \mathcal{E}_{\mathcal{B}}$, where $\mathcal{E}_{\mathcal{S}}$ was defined in Equation 8 and*

$$\mathcal{E}_{\mathcal{B}} := \{E_B\}_{B\in\mathcal{B}}, \quad \text{where } E_B(p) := B(\pi_{t-1}, x_t, p) \text{ for } p \in \Delta\mathcal{Y},$$

then in expectation over $P_1, P_2, \ldots \in \Delta\Delta\mathcal{Y}$, the distributions Algorithm 5 outputs, the anticipated *and* realized *coverage of every label $y \in \mathcal{Y}$ by every prediction set algorithm $S \in \mathcal{S}$ conditional on every $B \in \mathcal{B}$ will be approximately equal at every round $T \in [T_{\max}]$ (where $T_{\max}$ is the maximum time horizon):*

$$\mathop{\mathbb{E}}_{p_t \sim P_t, \forall t}\left[\left(\sum_{t\in[T]} B(x_t, p_t)\right) \cdot \left|\overline{\Pr}_T[y \in S|B] - \widetilde{\Pr}_T[y \in S|B]\right|\right]$$
$$\leq O\left(\ln(|\mathcal{Y}||\mathcal{S}||\mathcal{B}|T_{\max}) + \sqrt{\ln(|\mathcal{Y}||\mathcal{S}||\mathcal{B}|T_{\max}) \cdot \sum_{t=1}^{T} \mathop{\mathbb{E}}_{p_t\sim P_t}[B(x_t, p_t) \cdot \Pr[y \in S_t]]}\right).$$

Moreover, the anticipated *and* realized *conditional coverage of every prediction set algorithm $S \in \mathcal{S}$ conditional on every $B \in \mathcal{B}$ will be approximately equal for all $T \in [T_{\max}]$ (where $T_{\max}$ is the maximum time horizon):*

$$\mathop{\mathbb{E}}_{p_t \sim P_t, \forall t}\left[\left(\sum_{t\in[T]} B(x_t, p_t)\right) \cdot \left|\overline{\Pr}_T[Y \in S|B] - \widetilde{\Pr}_T[Y \in S|B]\right|\right]$$

$$\leq O\left(|\mathcal{Y}|\ln(|\mathcal{Y}||\mathcal{S}||\mathcal{B}|T_{\max}) + \sum_{y\in\mathcal{Y}}\sqrt{\ln(|\mathcal{Y}||\mathcal{S}||\mathcal{B}|T_{\max})\cdot\sum_{t\in[T]}\mathop{\mathbb{E}}_{p_t\sim P_t}[B(x_t,p_t)\cdot\Pr[y\in S_t]]}\right)$$
$$\leq \tilde{O}\left(|\mathcal{Y}|\ln(|\mathcal{Y}||\mathcal{S}||\mathcal{B}|T_{\max})\sqrt{\sum_{t\in[T]}\mathop{\mathbb{E}}_{p_t\sim P_t}[B(x_t,p_t)]}\right).$$

As a corollary of this result, we obtain the corresponding coverage guarantees for any family of $(1-\alpha)$-aspiring downstream prediction set algorithms. This is a strengthening of Corollary 38, and gives our most general valid $(1-\alpha)$ conditional coverage guarantees.

Corollary 43 (Valid realized $\mathcal{B}$-conditional coverage for $(1-\alpha)$-aspiring prediction set algorithms) *For any family $\mathcal{S}$ of $(1-\alpha)$-aspiring prediction set algorithms and any booster family $\mathcal{B}$, instantiating our method* `Class-Probabilities-for-Prediction-Sets` *with the collection of events $\mathcal{E} = \mathcal{E}_{\mathcal{S}} \times \mathcal{E}_{\mathcal{B}}$ guarantees $1-\alpha$ realized coverage conditional on every $B \in \mathcal{B}$ to all algorithms $S \in \mathcal{S}$ simultaneously, at all times $T \leq T_{\max}$, with deviation from exact $1-\alpha$ coverage satisfying:*

$$\mathop{\mathbb{E}}_{p_t\sim P_t,\forall t}\left[\left(\sum_{t\in[T]}B(x_t,p_t)\right)\cdot\left|\overline{\Pr}_T[Y\in S|B]-(1-\alpha)\right|\right] \leq \tilde{O}\left(|\mathcal{Y}|\ln(|\mathcal{Y}||\mathcal{S}||\mathcal{B}|T_{\max})\sqrt{\sum_{t\in[T]}\mathop{\mathbb{E}}_{p_t\sim P_t}[B(x_t,p_t)]}\right).$$

We now briefly discuss two important applications of our transparent conditional coverage guarantees: set-size-conditional coverage (cf. *size-stratified coverage validity* of [3]), and multigroup coverage (introduced in [7, 59, 60]). We have singled them out in the pseudocode for our method (Algorithm 5) due to their importance, even though they follow from very simple instantiations of our boosters.

Set-size-conditional coverage The idea behind obtaining valid coverage conditional on the set size is quite intuitive [3, 4]. We ideally want to ensure that the prediction set is always appropriately large, reflecting the true amount of uncertainty with respect to any sample — and thus want the target $(1-\alpha)$ coverage to hold conditional on picking any possible set size. In particular, this helps avoid situations where the prediction set is very small *and* undercovers (which clearly means the optimal prediction set should have been made larger), or where the prediction set is very large *and* overcovers (which clearly means it would be better to remove some labels from the set, improving both its coverage and its size).

While set-size-conditional coverage has been explored in the batch conformal setting, to our knowledge such coverage guarantees have not been rigorously studied in the online adversarial case; we fill this gap in Theorem 45 below. Before presenting our result, we first formally introduce and discuss a generalized measure of prediction set size.

Definition 44 (Set size function) *A* set-size function sz *is a mapping*[7] $\mathrm{sz} : [0,1]^{|\mathcal{Y}|} \to \{0,1,\ldots,N_{\mathrm{maxsz}}\}$, *where N_{maxsz} denotes the maximum set size. If the prediction set algorithm is known to be deterministic, the set size function can simply be defined as a mapping $\mathrm{sz} : 2^{\mathcal{Y}} \to \{0,1,\ldots,N_{\mathrm{maxsz}}\}$.*

Given any realized prediction set S_{pred}, its size *is then given by* $\mathrm{sz}(S_{\mathrm{pred}})$.

7. As mentioned earlier, sz can more generally be any bounded mapping from $[0,1]^{|\mathcal{Y}|}$ to $[0,\infty)$; in that case, we can simply discretize it to reduce to the definition given here.

What are some examples of useful set size functions? The simplest is to take, for any prediction set $S_{\text{pred}} \in [0,1]^{|\mathcal{Y}|}$, its size to be $\text{sz}(S_{\text{pred}}) := \sum_{y \in \mathcal{Y}} (S_{\text{pred}})_y$. If the prediction set is deterministic, i.e. $S_{\text{pred}} \in \{0,1\}^{|\mathcal{Y}|}$, this reduces to $\text{sz}(S_{\text{pred}}) := \sum_{y \in \mathcal{Y}} 1[y \in S_{\text{pred}}]$, which is the size of S_{pred} as a set of labels. We can also define a *weighted* variant of this set size mapping: given any fixed nonnegative label weights $(w_y)_{y \in \mathcal{Y}}$, we can let $\text{sz}(S_{\text{pred}}) := \sum_{y \in \mathcal{Y}} w_y \cdot (S_{\text{pred}})_y$, which for deterministic prediction sets reduces to $\text{sz}(S_{\text{pred}}) := \sum_{y \in \mathcal{Y}} w_y \cdot 1[y \in S_{\text{pred}}]$. This can be useful when there is a different "cost to pay" for including each label $y \in \mathcal{Y}$ into the prediction set.

Beyond these examples, we can define $\text{sz}(\cdot)$ in arbitrary ways. For instance, one may intuitively desire that for deterministic sets S_{pred}, the set size $\text{sz}(S_{\text{pred}})$ should grow with $|S_{\text{pred}}|$; however, we place no such requirement on sz. Even more generally, sz can be interpreted decision-theoretically as a (dis)utility function that maps any selected prediction set to a certain (dis)utility value. Our next guarantee covers all these cases:

Theorem 45 (Valid set-size-conditional coverage for $(1-\alpha)$-aspiring prediction set algorithms) *For any family $\mathcal{S}$ of $(1-\alpha)$-aspiring prediction set algorithms and any set size function $\text{sz} : [0,1]^{|\mathcal{Y}|} \to \{0,1,\ldots,N_{\text{maxsz}}\}$, instantiating* `Class-Probabilities-for-Prediction-Sets` *with the collection of events $\mathcal{E} = \bigcup_{S \in \mathcal{S}} \mathcal{E}_S \times \mathcal{E}_{\text{sz,S}}$, where for each $S \in \mathcal{S}$, we define $\mathcal{E}_S := \{E_{S,y}\}_{y \in \mathcal{Y}}$ with $E_{S,y}$ defined as in Equation 8, and*

$$\mathcal{E}_{\text{sz,S}} := \{E_{\text{sz},S,n}\}_{0 \le n \le N_{\text{maxsz}}}, \text{ where } E_{\text{sz},S,n}(p) := 1[\text{sz}(S(\pi_{t-1} \cup \{x_t\}, p)) = n] \text{ for all } p \in \Delta\mathcal{Y},$$

guarantees $1-\alpha$ realized coverage, conditional on every particular set size $0 \le n \le N_{\text{maxsz}}$, to all algorithms $S \in \mathcal{S}$ simultaneously, at all times $T \le T_{\text{max}}$, with deviation from target coverage satisfying:

$$\mathop{\mathbb{E}}_{p_t \sim P_t,\, \forall t}\left[\text{num}_T(\text{sz}=n) \cdot \left|\overline{\Pr}_T[Y \in S \,|\, \text{sz}=n] - (1-\alpha)\right|\right] \le \tilde{O}\left(|\mathcal{Y}| \ln(|\mathcal{Y}||\mathcal{S}|N_{\text{maxsz}}T_{\text{max}}) \sqrt{\mathop{\mathbb{E}}_{p_t \sim P_t \, \forall t}[\text{num}_T(\text{sz}=n)]}\right),$$

where $\text{num}_T(\text{sz}=n) := \sum_{t \in [T]} 1[\text{sz}(S(\pi_{t-1}, x_t, p)) = n]$ denotes the incidence of set size n through round T, and $\overline{\Pr}_T[Y \in S|\text{sz}=n]$ is a shorthand for the realized coverage conditional on the corresponding subsequence.

This result is a direct corollary of Corollary 43, since the events corresponding to each prediction set size are instances of boosters.

Multigroup coverage As originally introduced, the idea behind multigroup coverage is to protect various contextually-defined subsets of the data from over- and undercoverage. This is best illustrated by settings where the feature space $\mathcal{X}$ corresponds to feature vectors of individuals in a population. In that case, many notions of demographic groups considered in the fairness literature — e.g., defined by sex, age, skin color, gender — are captured by defining appropriate context-dependent group mappings $G : \mathcal{X} \to [0,1]$, where $G(x) = 1$ means the individual with features x belongs to the group and vice-versa for $G(x) = 0$, and $G(x) \in (0,1)$ denotes *partial* group membership (e.g., belonging to the group with probability $G(x)$). The goal is then to ensure that prediction sets produced by an algorithm S achieve $\approx 1-\alpha$ coverage not just on average over the entire population, but also conditionally on each group G that we wish to protect. This desideratum is referred to as *multi*group coverage to emphasize that the protected groups do not necessarily partition the feature space $\mathcal{X}$ but can instead overlap in arbitrary ways.

Definition 46 (Group family) *A* group family *is a (finite) collection of groups $G \in \mathcal{G}$, where each G is an arbitrary mapping from $\mathcal{X}$ to $[0, 1]$. We assume that we have oracle access to every $G \in \mathcal{G}$, i.e., that we can query it in $O(1)$ time on every individual $x \in \mathcal{X}$.*

We now state our multigroup coverage result; we note that it improves on the previous multigroup coverage result of [7], as our coverage error bound now has the optimal $O(\ln |\mathcal{G}|)$ dependence on the number of groups in the collection, while retaining the statistically optimal $1/\sqrt{\sum_{t\in[T]} G(x_t)}$ dependence on the frequency of occurrence of every group $G \in \mathcal{G}$.

Theorem 47 (Valid multigroup coverage for $(1-\alpha)$-aspiring prediction set algorithms) *For any family $\mathcal{S}$ of $(1-\alpha)$-aspiring prediction set algorithms and any group collection $\mathcal{G}$, instantiating our method `Class-Probabilities-for-Prediction-Sets` with the collection of events $\mathcal{E} = \mathcal{E}_{\mathcal{S}} \times \mathcal{E}_{\mathcal{G}}$, where:*

$$\mathcal{E}_{\mathcal{G}} := \{E_G\}_{G\in\mathcal{G}}, \text{ where } E_G(p) := G(x_t) \text{ for all } p \in \Delta\mathcal{Y},$$

guarantees $1-\alpha$ realized coverage, conditional on every demographic group $G \in \mathcal{G}$, to all algorithms $S \in \mathcal{S}$ simultaneously, at all times $T \leq T_{\max}$, with deviation from exact $1-\alpha$ coverage satisfying:

$$\mathop{\mathbb{E}}_{p_t\sim P_t,\,\forall t}\left[\left|\overline{\Pr}_T[Y \in S \,|\, G] - (1-\alpha)\right|\right] \leq \tilde{O}\left(\frac{|\mathcal{Y}|\ln(|\mathcal{Y}||\mathcal{S}||\mathcal{G}|T_{\max})}{\sqrt{\mathrm{num}_T(G)}}\right),$$

where $\mathrm{num}_T(G) := \sum_{t\in[T]} G(x_t)$ denotes the incidence of group G through round T, and $\overline{\Pr}_T[Y \in S \,|\, G]$ is a shorthand for the realized coverage conditional on the group's subsequence.

This result is a direct corollary of Corollary 43, since the events corresponding to each demographic group G are instances of boosters.

E.3. Best-in-Class Probabilistic Predictions

We have just established that our probability vector predictions are not only capable of providing marginal coverage guarantees, but can in fact (with the help of an appropriately augmented conditioning event collection $\mathcal{E}$) provide conditional coverage guarantees of any desired granularity. This represents one way of ensuring that our predictions are 'good' in an *absolute* sense — i.e. if we had the computational and statistical power to condition on all possible events, our predictor would approach the Bayes optimal one; however, in practice, conditioning on all events is unachievable.

Instead, we now focus on showing that we can make our predictions "good" in a very useful *relative* sense: given any collection of benchmark prediction models (which can be arbitrary), we will want our predictions to have lower loss than those of the best model in the benchmark class. Thus, whenever we have some other way of making good predictions (e.g. by using a large neural network), we can use that method as part of our own to endow predictions of the same or better quality with transparent coverage guarantees.

Indeed, in various domains, we might have high quality predictors trained on large quantities of data: these predictors might perform very well according to standard performance metrics, and yet not have the kinds of transparent coverage guarantees that we aim to give here. Our goal will be to take any (polynomially sized) collection of such predictors as inputs, and provide probabilistic predictions with anytime transparent coverage guarantees that perform as well as the best predictors in this benchmark class. This result will thus demonstrate that, in some sense, our anytime transparent coverage guarantees discussed above come "for free", rather than at the cost of predictive performance.

To formalize this, we introduce two definitions, of a class of predictors against which we will be able to compete, and of a class of loss functions measuring how well we can compete with these predictors.

Definition 48 (Competing Predictor) *A predictor q is defined as a sequence of prediction mappings $q_1, \ldots, q_t, \ldots$, where each $q_t : \Delta\mathcal{Y} \to \Delta\mathcal{Y}$ is a mapping from the learner's predictions to the predictor's predictions such that if the learner makes prediction p_t in round t then the predictor q will make prediction $q_t(p_t)$ in round t.*

In particular, predictor q is called a competing predictor *if for each t, the prediction mapping q_t is revealed to the learner at the beginning of round t (i.e., before the learner has to make a prediction).*

This definition is strong in that it allows (though does not require) the competing predictor to (1) depend in a functional way upon the predictions that the learner is about to issue, and (2) be updated dynamically over time without revealing any prediction mapping q_t to the learner until round t. An important special case is when the predictor q_t is an arbitrary pre-trained model f of the context, and simply predicts $q_t = f(x_t)$ — but our definition is more general and allows the predictor to change over time and even to depend on our own predictions.

E.3.1. SCORING RULES, LOSSES, AND BREGMAN DIVERGENCES

Following [1, 48, 68], we now define a natural family of loss functions known as *Bregman scores*. All such loss functions assess the quality of a predictor using a chosen Bregman divergence of its predictions from the ground truth.

Definition 49 (Scoring Rule, Expected Score) *A* scoring rule *is a nonnegative extended-valued mapping $\phi : \Delta\mathcal{Y} \times \mathrm{oh}(\Delta\mathcal{Y}) \to [0, \infty]$, where $\mathrm{oh}(\Delta\mathcal{Y})$ denotes the set of the $|\mathcal{Y}|$ standard basis vectors in $\Delta\mathcal{Y}$.*

An expected score *ℓ associated with scoring rule ϕ is a mapping $\ell : \Delta\mathcal{Y} \times \Delta\mathcal{Y} \to [0, \infty)$ such that for every $p_1, p_2 \in \Delta\mathcal{Y}$,*

$$\ell(p_1, p_2) = \sum_{i \in [k]} p_{2,i} \cdot \phi(p_1, e_i),$$

where $e_i \in \Delta\mathcal{Y}$ denotes the ith standard basis vector.

Definition 50 (Bregman Divergence) *Given a strictly convex function $f : [0, 1] \to \mathbb{R}$ that is differentiable on $(0, 1)$, its associated* Bregman divergence over the simplex $\Delta\mathcal{Y} \subset \mathbb{R}^k$ *is the mapping*

$d_f : \Delta\mathcal{Y} \times \mathrm{ri}(\Delta\mathcal{Y}) \to \mathbb{R}$ *defined*[8] *as*

$$d_f(p_1, p_2) = \sum_{i \in [k]} \{f(p_{1,i}) - f(p_{2,i}) - f'(p_{2,i}) \cdot (p_{1,i} - p_{2,i})\}.$$

We simply refer to a function $d : \Delta\mathcal{Y} \times \mathrm{ri}(\Delta\mathcal{Y}) \to \mathbb{R}$ *as a* Bregman divergence *if* d *is identically equal to* d_f *for some strictly convex differentiable* $f : [0,1] \to \mathbb{R}$.

Furthermore, for any $i \in [k]$ *we define the notation* $d^i_f(p_1, p_2) := f(p_{1,i}) - f(p_{2,i}) - f'(p_{2,i}) \cdot (p_{1,i} - p_{2,i})$*, so that* $d_f(p_1, p_2) = \sum_{i \in [k]} d^i_f(p_1, p_2)$ *for any* $p_1 \in \Delta\mathcal{Y}, p_2 \in \mathrm{ri}(\Delta\mathcal{Y})$*. Viewed as a mapping from* $p_{1,i} \in [0,1]$ *and* $p_{2,i} \in (0,1)$ *to* $[0,\infty)$*,* d^i *is a* Bregman divergence (between two scalars) over $[0,1]$.

We note that our definition of a Bregman divergence between two vectors asks that it be *separable* across the coordinates of these vectors (as opposed to the most general definition of Bregman divergence that drops this requirement); this separability property is convenient for us to use given our techniques and is satisfied for major Bregman divergences of interest (e.g., Euclidean distance, KL divergence, Itakura-Saito distance).

Definition 51 (Bregman Score) *An expected score* $\ell : \Delta\mathcal{Y} \times \Delta\mathcal{Y} \to [0,\infty)$ *is called a* Bregman score *if the mapping* $D : \Delta\mathcal{Y} \times \Delta\mathcal{Y} \to \mathbb{R}$ *defined, for every* $p_1, p_2 \in \Delta\mathcal{Y}$*, as*

$$D(p_1, p_2) = \ell(p_1, p_2) - \ell(p_2, p_2)$$

satisfies $D(p_1, p_2) = d(p_2, p_1)$ *for some Bregman divergence* d *and all* $p_1 \in \Delta\mathcal{Y}, p_2 \in \mathrm{ri}(\Delta\mathcal{Y})$*.*

The following two examples showcase two key members of the class of Bregman scores.

Example 1 (Brier score) *The* squared Euclidean distance*, defined as*

$$d_{L^2}(p_1, p_2) = ||p_1 - p_2||^2 = \sum_{i \in [k]} (p_{1,i} - p_{2,i})^2,$$

is a Bregman divergence with respect to the strictly convex differentiable function $f(x) = x^2$*.*

Consider the Brier scoring rule ϕ_{Brier}*, given by* $\phi_{\mathrm{Brier}}(p, e) := \sum_{i \in [k]} (p_i - e_i)^2$ *for any* $p \in \Delta\mathcal{Y}$ *and any standard basis vector* e*. The* expected Brier score ℓ_{Brier} *is then given by*

$$\ell_{\mathrm{Brier}}(p_1, p_2) = \sum_{i \in [k]} p_{2,i} \left(\sum_{j \neq i} p_{1,j}^2 + (1 - p_{1,i})^2 \right) = 1 + ||p_1||^2 - 2\langle p_1, p_2 \rangle.$$

Now, it can be easily verified that the expected Brier score is in fact the Bregman score corresponding to the squared Euclidean distance, since $\ell_{\mathrm{Brier}}(p_1, p_2) - \ell_{\mathrm{Brier}}(p_2, p_2) = 1 + ||p_1||^2 - 2\langle p_1, p_2 \rangle - (1 + ||p_2||^2 - 2\langle p_2, p_2 \rangle) = ||p_1||^2 - 2\langle p_1, p_2 \rangle + ||p_2||^2 = ||p_1 - p_2||^2 = d_{L^2}(p_2, p_1)$*.*

8. Here, ri denotes the relative interior of a set. The definition of Bregman divergence that restricts the second argument to be in the relative interior of the domain is customary in the literature [6], and helps avoid problems at boundary points (such as e.g. for KL divergence). However, we note that for sufficiently well-behaved functions f, their Bregman divergence d_f can easily be well-defined on the boundary — such is the case e.g. for the squared loss. When that is the case, additional restrictions that we have to place on the second argument of the Bregman divergence (i.e., in our context, assuming that all coordinates of our and competing predictions are nonzero, as in Theorem 54) can naturally be ignored.

Example 2 (Logarithmic score) *The* KL divergence*, defined as*

$$d_{\mathrm{KL}}(p_1, p_2) = \sum_{i \in [k]} p_{1,i} \log \frac{p_{1,i}}{p_{2,i}},$$

is a Bregman divergence with respect to the strictly convex function $f(x) = x \log x$ differentiable on $(0, \infty)$.

Consider the logarithmic scoring rule ϕ_{Log}*, given by $\phi_{\mathrm{Log}}(p, e) := -\log p_{i(e)}$ for any $p \in \Delta\mathcal{Y}$ and any standard basis vector e (where by $i(e)$ we denote the index of the nonzero coordinate of e). The* expected logarithmic score ℓ_{Log} *is then given by*

$$\ell_{\mathrm{Log}}(p_1, p_2) = \sum_{i \in [k]} p_{2,i} \left(-\log p_{1,i}\right) = -\sum_{i \in [k]} p_{2,i} \log p_{1,i}.$$

As it turns out, the expected logarithmic score is in fact the Bregman score with respect to the KL divergence, since $\ell_{\mathrm{Log}}(p_1, p_2) - \ell_{\mathrm{Log}}(p_2, p_2) = -\sum_{i \in [k]} p_{2,i} \log p_{1,i} + \sum_{i \in [k]} p_{2,i} \log p_{2,i} = \sum_{i \in [k]} p_{2,i} \log \frac{p_{2,i}}{p_{1,i}} = d_{\mathrm{KL}}(p_2, p_1)$.

In the remainder of this section, we will lightly abuse notation. Namely, we henceforth let y_t, formerly the notation for the realized label in round t, stand for the standard basis vector in $\Delta\mathcal{Y}$ whose entry corresponding to the realized label is 1 and all other entries are 0. This makes quantities like $\ell(p_t, y_t)$ well-defined since now both $p_t, y_t \in \Delta\mathcal{Y}$.

We now define our notion of loss for a sequential predictor — which will be its cumulative Bregman loss across all rounds.

Definition 52 (Bregman Loss of a Predictor) *Given any Bregman score ℓ, we define the Bregman loss of a predictor $q = (q_t)_{t \in [T]}$ over any T-round transcript π_T as:*

$$L_\ell(q, \pi_T) := \sum_{t \in [T]} \ell(q_t, y_t).$$

E.3.2. A BEST-IN-CLASS RESULT FOR BREGMAN LOSSES

Now, we are ready to show the following result: for any family $\mathcal{Q}$ of competing predictors q, if we condition on the product of some δ-level sets of each competing predictor $q \in \mathcal{Q}$ with the δ-level sets of our predictor in every coordinate (altogether, this amounts to $O(k|\mathcal{Q}|\lceil \delta^{-2} \rceil)$ events) with δ chosen appropriately, then for all Lipschitz Bregman losses simultaneously, the loss of our predictor will be no greater than the loss of any competing predictor, up to lower-order terms.

Definition 53 (δ-level sets) *A partition Ω of the interval $[0, 1]$ into subintervals, in order, $\Omega_1, \ldots, \Omega_N$ is called a δ-partition if the length of every Ω_i is no greater than δ, for all $i \in [N]$. For a predictor with values in $[0, 1]$, its δ level sets are any level sets Ω generated by a δ-partition of $[0, 1]$.*

Theorem 54 (Beating Competing Predictors under All L-Lipschitz Bregman Losses) *Fix any $L > 0$ and let $\mathcal{L}_L$ be the family of all Bregman losses L_ℓ whose associated Bregman divergence d_ℓ is L-Lipschitz in its second argument:*

$$|d_\ell(p_1, p_2) - d_\ell(p_1, p_2')| \leq L|p_2 - p_2'| \quad \text{for all } p_1 \in \Delta\mathcal{Y} \text{ and } p_2, p_2' \in \mathrm{ri}(\Delta\mathcal{Y}).$$

Consider any family $\mathcal{Q}$ of competing predictors $q = (q_t)_{t\in\mathbb{N}}$. Suppose we make our predictor $p = (p_t)_{t\in\mathbb{N}}$ unbiased conditional on a family of events $\mathcal{E}_\mathcal{Q}$ that includes, for every $q \in \mathcal{Q}$, the product of some coordinate-wise (for $i \in [k]$) δ-level sets $\Omega^{p,i}$ and $\Omega^{q,i}$ of p and q, respectively:

$$\mathcal{E}_\mathcal{Q} = \{\mathcal{E}_{\text{ls(p,q)}}\}_{q\in\mathcal{Q}} \text{ with } \mathcal{E}_{\text{ls(p,q)}} := \{E^i_{a,b}\}_{(i,a,b)\in[k]\times[N_1]\times[N_2]},$$

where for each $i \in [k]$, $a \in [N_1], b \in [N_2]$, we define the product of the ath level set of p and the bth level set of q with respect to coordinate i as

$$E^i_{a,b}(p_t) := \mathbf{1}[p_{t,i} \in \Omega^{p,i}_a \wedge q_{t,i}(p_t) \in \Omega^{q,i}_b].$$

Further, suppose that $p_{t,i}, q_{t,i} \in (0,1)$ for all $t \in \mathbb{N}$, $q \in \mathcal{Q}$ and $i \in [d]$.

Then, the expected Bregman loss L_ℓ of our predictor p (over its internal randomness) is no greater than every *competing predictor's loss L_ℓ for* all Bregman losses *$L_\ell \in \mathcal{L}_L$, up to $o(T)$ terms. Namely, setting $\delta = \Theta(T^{-1/4})$, we have:*

$$\mathbb{E}_{\pi_T}[L_\ell(p,\pi_T)] \le \min_{q\in\mathcal{Q}} L_\ell(q,\pi_T) + O(L\,k\,\sqrt{\ln(k|\mathcal{Q}|T)}\,T^{3/4}) \text{ for all } L_\ell \in \mathcal{L}_L.$$

Proof Fix any L-Lipschitz Bregman loss $L_\ell \in \mathcal{L}_L$, and any competing predictor $q \in \mathcal{Q}$. First, note that for any transcript π_T of the first T rounds:

$$L_\ell(p,\pi_T)-L_\ell(q,\pi_T) = \sum_{t\in[T]} \ell(p_t,y_t)-\ell(q_t,y_t) = \sum_{t\in[T]} d(y_t,p_t)-d(y_t,q_t) = \sum_{i\in[k]}\left(\sum_{t\in[T]} d^i(y_t,p_t) - d^i(y_t,q_t)\right),$$

where d is the Bregman divergence associated with ℓ, and d^i are the coordinate-wise terms that our Bregman divergence d splits into.

Thus, we now fix a single coordinate i and carry out the proof for that coordinate alone; the final bound will just be multiplied by a factor of $k = |\mathcal{Y}|$ to give the overall Bregman loss bound. Note that:

$$\sum_{t\in[T]} d^i(y_t,p_t) = \sum_{a,b}\sum_{t:E^i_{a,b}(p_t)=1} d^i(y_{t,i},p_{t,i}), \quad \sum_{t\in[T]} d^i(y_t,q_t) = \sum_{a,b}\sum_{t:E^i_{a,b}(p_t)=1} d^i(y_{t,i},q_{t,i}).$$

We now focus on any fixed $a \in [N_1], b \in [N_2]$. For convenience, let $n^i_{a,b} := \sum_t \mathbf{1}[E^i_{a,b}(p_t) = 1]$ denote the total number of rounds on which the event $E^i_{a,b}$ transpired, and let the average values over this event's subsequence of p, q, y be defined as $\mu^i_{a,b}(p) := \frac{1}{n^i_{a,b}}\sum_t E^i_{a,b}(p_t)\cdot p_{t,i}$, as $\mu^i_{a,b}(q) := \frac{1}{n^i_{a,b}}\sum_t E^i_{a,b}(p_t)\cdot q_{t,i}$, and as $\mu^i_{a,b}(y) := \frac{1}{n^i_{a,b}}\sum_t E^i_{a,b}(p_t)\cdot y_{t,i}$, respectively.

Regarding these averages, observe that for any t, letting $\alpha^i_{a,b}$ denote any bound on the absolute bias on this subsequence, we get

$$|p_{t,i} - \mu^i_{a,b}(y)| \le |p_{t,i} - \mu^i_{a,b}(p)| + |\mu^i_{a,b}(p) - \mu^i_{a,b}(y)| \le \delta + \alpha^i_{a,b}.$$

Here, the first term is bounded by δ since the subinterval $\Omega^{p,i}_{a,b}$ is convex (so that both $p_{t,i}$ and the average over time $\mu^i_{a,b}(y)$ belong to it) and has length at most δ. The second term is bounded by $\alpha^i_{a,b}$ by definition of subsequence bias.

Similarly, using that $\Omega^{q,i}$ is a δ-level sets partition for the competing predictor q, we have for any t that

$$|q_{t,i} - \mu^i_{a,b}(q)| \leq \delta.$$

By the L-Lipschitzness of d^i in its second argument, we therefore have for our predictor p that

$$\sum_{t:E^i_{a,b}(p_t)=1} d^i(y_{t,i}, p_{t,i}) \leq \sum_{t:E^i_{a,b}(p_t)=1} d^i\left(y_{t,i}, \mu^i_{a,b}(y)\right) + L \cdot n^i_{a,b} \cdot (\delta + \alpha^i_{a,b}),$$

and for the competing predictor q that

$$\sum_{t:E^i_{a,b}(p_t)=1} d^i(y_{t,i}, q_{t,i}) \geq \sum_{t:E^i_{a,b}(p_t)=1} d^i\left(y_{t,i}, \mu^i_{a,b}(q)\right) - L \cdot n^i_{a,b} \cdot \delta.$$

Therefore, we have

$$\begin{aligned}
\sum_{t\in[T]} d^i(y_t, p_t) - d^i(y_t, q_t) &= \sum_{a,b}\left(\sum_{t:E^i_{a,b}(p_t)=1} d^i(y_t, p_t) - \sum_{t:E^i_{a,b}(p_t)=1} d^i(y_t, q_t)\right) \\
&\leq \sum_{a,b}\left(\sum_{t:E^i_{a,b}(p_t)=1} d^i\left(y_{t,i}, \mu^i_{a,b}(y)\right) - \sum_{t:E^i_{a,b}(p_t)=1} d^i\left(y_{t,i}, \mu^i_{a,b}(q)\right)\right) \\
&+ \sum_{a,b} L \cdot n^i_{a,b} \cdot (2\delta + \alpha^i_{a,b}) \\
&\leq \sum_{a,b} L \cdot n^i_{a,b} \cdot (2\delta + \alpha^i_{a,b}) \\
&= 2L\delta T + L \sum_{a,b} n^i_{a,b}\alpha^i_{a,b}.
\end{aligned}$$

Here, the last equality uses that $\sum_i n^i_{a,b} = T$ for every i. Meanwhile, the second inequality is by the fundamental fact that *Bregman divergences elicit means*, first observed by [6]:

Theorem 55 (Adapted from Proposition 1 of [6]) *For any Bregman divergence* $d : [0,1]\times(0,1) \to [0,\infty)$*, and any random variable* Y *with finite support over* $[0,1]$ *such that* $\mathbb{E}[Y] \in (0,1)$*, we have:*

$$\mathbb{E}[Y] = \operatorname*{argmin}_{y\in(0,1)} \mathbb{E}[d(Y,y)].$$

Applying this fact for d^i, which is a Bregman divergence over $[0,1]$, to the random variable Y representing the empirical distribution of the realized labels $y_{t,i}$ over the rounds t on which $E^i_{a,b}(p_t) = 1$, we get:

$$\mu^i_{a,b}(y) = \frac{1}{n^i_{a,b}} \sum_{t:E^i_{a,b}} y_{t,i} = \mathbb{E}[Y] = \operatorname*{argmin}_{y\in(0,1)} \mathbb{E}[d^i(Y,y)] = \operatorname*{argmin}_{y\in(0,1)} \frac{1}{n^i_{a,b}} \sum_{t:E^i_{a,b}} d^i(y_{t,i}, y).$$

This then implies the second inequality in the derivation above, since we now have:

$$\sum_{t:E^i_{a,b}(p_t)=1} d^i\left(y_{t,i}, \mu^i_{a,b}(y)\right) \leq \sum_{t:E^i_{a,b}(p_t)=1} d^i\left(y_{t,i}, \mu^i_{a,b}(q)\right).$$

Summing the inequality $\sum_{t\in[T]} d^i(y_t,p_t) - d^i(y_t,q_t) \leq 2L\delta T + L\sum_{a,b} n^i_{a,b}\alpha^i_{a,b}$ over $i \in [k]$, we obtain:

$$L_\ell(p,\pi_T) - L_\ell(q,\pi_T) \leq \sum_i \left(2L\delta T + L\sum_{a,b} n^i_{a,b}\alpha^i_{a,b}\right) = 2kL\delta T + L\sum_{i,a,b} n^i_{a,b}\alpha^i_{a,b}.$$

By Theorem 14, our prediction algorithm achieves for all i, a, b (recalling that our prediction vector dimension is k and the number of events we condition on is of order $|\mathcal{Q}|k/\delta^2$):

$$\mathop{\mathbb{E}}_{\pi_T}[n^i_{a,b}\alpha^i_{a,b}] \leq c'\ln(2k(|\mathcal{Q}|k/\delta^2)T) + c'\sqrt{\ln(2k(|\mathcal{Q}|k/\delta^2)T)} \cdot \sqrt{\mathop{\mathbb{E}}_{\pi_T}[n^i_{a,b}]}$$

for some constant $c' > 0$. We thus have:

$$\mathop{\mathbb{E}}_{\pi_T}[L_\ell(p,\pi_T)] - L_\ell(q,\pi_T) \leq 2kL\delta T + 2Lc'\ln(2Tk/\delta)k/\delta^2 + c'L\sqrt{\ln(2k(|\mathcal{Q}|k/\delta^2)T)}\sum_i\left(\sum_{a,b}\sqrt{\mathop{\mathbb{E}}_{\pi_T}[n^i_{a,b}]}\right).$$

Now note that for any fixed i, the events corresponding to each pair (a,b) are disjoint by definition, and thus we have that $\sum_{a,b} n^i_{a,b} = T$ in that case. Since there are at most $1/\delta^2$ terms in the sum $\sum_{a,b}\sqrt{\mathbb{E}_{\pi_T}[n^i_{a,b}]}$, we thus have that its worst-case value over all possible values of $n^i_{a,b}$ is when $\mathbb{E}_{\pi_T}[n^i_{a,b}] = T\delta^2$ for all a, b. Bounding for convenience $\sqrt{\ln(2k(|\mathcal{Q}|k/\delta^2)T)}$ by $2\ln(2k|\mathcal{Q}|T/\delta)$ in the last term and rearranging, this gives us the worst-case bound of

$$\mathop{\mathbb{E}}_{\pi_T}[L_\ell(p,\pi_T)] - L_\ell(q,\pi_T) \leq 2kL\delta T + 5\ln(2k|\mathcal{Q}|T/\delta)\cdot c'\cdot L\cdot k\cdot\frac{1}{\delta^2}\cdot\sqrt{T\delta^2}.$$

Finally, setting $\delta = \frac{\sqrt{\ln(k|\mathcal{Q}|T)}}{T^{1/4}}$ now asymptotically equalizes both terms in the upper bound, giving us:

$$\mathop{\mathbb{E}}_{\pi_T}[L_\ell(p,\pi_T)] - L_\ell(q,\pi_T) \leq c''kL\sqrt{\ln(k|\mathcal{Q}|T)}T^{3/4},$$

for some constant $c'' > 0$, thus implying our desired bound for all $L_\ell \in \mathcal{L}_L, q \in \mathcal{Q}$. ■

Now, observe that our theorem critically uses the Lipschitzness of the coordinate-wise Bregman scores in the second argument throughout its domain $(0,1)$. This property will indeed be satisfied by a great many Bregman divergences, including, for example, all those that, like the Euclidean distance, are based on a *Lipschitz* convex function f. However, there also exist many useful Bregman divergences, including notably KL divergence, that are not Lipschitz but are *locally* Lipschitz on $(0,1)$.

Example 3 (Local Lipschitzness of KL divergence) *Consider the ith component of the KL divergence, for any $i \in [k]$: $d^i_{\mathrm{KL}}(x,y) = x\log\frac{x}{y}$. Observe that $\frac{\partial}{\partial y}\left(x\log\frac{x}{y}\right) = -\frac{x}{y}$. Therefore, since in our setting $x \in [0,1]$, we obtain that the KL divergence is $\frac{1}{\epsilon}$-Lipschitz in its second argument y whenever $y \in (\epsilon, 1)$.*

Based on this, we can easily formulate an updated version of Theorem 54 for KL-divergence:[9]

Corollary 56 (Beating the log loss of competing predictors) *Given any competing predictor* $q = (q_t)_{t\in[T]}$, *our predictor* $p = (p_t)_{t\in[T]}$ *can achieve (expected) log loss no worse than that of* q, *up to a sublinear term:*

$$\mathop{\mathbb{E}}_{\pi_T}\left[L_{\text{Log}}(p, \pi_T)\right] \leq L_{\text{Log}}(q, \pi_T) + \tilde{O}\left(k \cdot \frac{1}{\min_{t\in[T],i\in[k]} \min\{p_{t,i}, q_{t,i}\}} \cdot T^{3/4}\right) \quad \textit{for any } T.$$

Since we have control over our predictor p, we can easily enforce that it doesn't have any very small components $p_{t,i}$. However, it would appear that we do not have control over the competing predictor q; thus, in theory, this bound could quickly deteriorate if the competing predictor continually predicts ever smaller values. However, this should not be a problem in practice: indeed, we can threshold any competing predictor to never predict less than a certain small probability value in each coordinate without significantly affecting downstream prediction set making, given that any "reasonable" prediction set algorithm would be extremely unlikely to include an exceedingly small-probability label in its prediction set.

Remark 57 (Theorem 54 holds over any convex compact region $\mathcal{C} \subset \mathbb{R}^k$, not just the simplex $\Delta\mathcal{Y}$) *We want to highlight that our Best-in-Class Theorem 54 (along with natural locally-Lipschitz extensions in the spirit of 56) holds for* any *convex compact prediction space* $\mathcal{C}$, *not just the simplex* $\Delta\mathcal{Y}$ *— we only presented it as a result over the simplex since this is the context in which we are applying this result in this section.*

Remark 58 (An Omniprediction view of Theorem 54; The necessity of using Bregman losses) *Our theorem provides an omniprediction-type statement, in the sense of [41]. That is, it is a statement of the form: "Given a collection of competing predictors, and a family of well-behaved loss functions, it is possible to train a predictor that does (up to a small error term) at least as well as every competing predictor, simultaneously with respect to all loss functions in the family."*

Our results are stronger than the original omniprediction results in that (1) we generalize from binary valued outcomes to real valued outcomes, and (2) we generalize from one-dimensional outputs to multi-dimensional outputs, as primarily considered in [41] and subsequent work. Our results also hold in the online adversarial setting, as in [36]. Necessarily, the class of loss functions that we produce "omnipredictors" with respect to is more limited than in the one-dimensional binary case. One dimensional binary distributions are fully characterized by their mean, and so sufficiently (multi)-calibrated mean prediction is enough information to optimize all loss functions defined over one-dimensional binary values. Real valued distributions are not characterized by their means — and so using tools based on multicalibration and unbiased prediction, we are necessarily limited to giving guarantees for loss functions such that the mean of the distribution is a sufficient statistic for optimization, which is what we do here. Indeed, note that a direct converse to the result of [6] that we used in our proof above, proved under various technical assumptions in [5] and later in [1], shows that any expected score *which elicits means must be a Bregman score; from this, it follows that our omniprediction result cannot hold more generally than over Bregman scores.*

9. We remark that it is straightforward to define a more general analog of Corollary 56, promising to beat competing predictors according not just to log loss, but to any other locally Lipschitz loss ℓ.

Appendix F. Application: Groupwise Swap Regret, and Subsequence Regret for Online Combinatorial Optimization and Extensive-Form Games

We now show how to use the machinery that we have developed to give algorithms in a variety of sequential settings. In all cases, the scenario we analyze is that in rounds t, a predictor makes predictions p_t, after which one or more decision makers choose actions that are best responses to p_t (i.e. they function as *straightforward* decision makers). When we are designing an algorithm for a single decision maker, we always use the *predict-then-act* paradigm (Algorithm 4). When we are designing a coordination mechanism, we imagine that our predictions are made to a variety of decision makers, who each independently act. We show how guaranteeing that our predictions are unbiased subject to appropriately chosen events gives desirable guarantees for the decision makers of various sorts.

F.1. Warm-up: (Groupwise) Swap Regret for the Experts Problem

As the simplest example of how to use the machinery we have developed, we show how to derive an algorithm that recovers optimal swap regret bounds for the experts problem (defined in Section C.1). Recall that in the experts problem, there are n "experts" $i \in [n]$, and at each round, the algorithm must choose an expert i_t (or distribution over experts q_t). A vector of gains $g_t \in [-1, 1]^n$ is realized, and the algorithm obtains reward $g_{t,i}$ if she chose expert $i_t = i$ (and expected reward $\langle q_t, g_t \rangle$ if she chose a distribution over experts q_t). We can derive an algorithm for this problem using our framework by — at every round t —

1. Making a prediction p_t for the vector of gains g_t that will be realized at the end of the round;
2. Selecting the expert i_t that has the largest predicted gain: $i_t = \text{argmax}_{i \in [n]} p_{t,i}$.

To cast this in our framework, we set the prediction space $\mathcal{C} = [-1, 1]^n$ to be the set of all feasible gain vectors, the action space $\mathcal{A} = \{1, \ldots, n\}$ to be the set of experts, and define a utility function $u : \mathcal{A} \times \mathcal{C} \to \mathbb{R}$ as $u(i, p) = p_i$. This utility function is indeed linear and L-Lipschitz in its second argument for $L = 1$. We instantiate our prediction algorithm to produce unbiased predictions with the set of n disjoint binary events $\mathcal{E} = \{E_{u,a}\}_{a \in \mathcal{A}}$ (in this case, for each $i \in [n]$, the event $E_{u,i}$ is defined such that $E_{u,i}(p_t) = 1$ if and only if $i = \text{argmax}_{i' \in [n]} p_{t,i}$). We can now read off swap regret bounds from the machinery we have built up:

Theorem 59 *The Predict-Then-Act Algorithm parameterized with $\mathcal{C} = [-1, 1]^n$ and $\mathcal{E} = \{E_{u,a}\}_{a \in \mathcal{A}}$ obtains at each round $t \le T$ expected swap regret at most:*

$$\mathop{\mathbb{E}}_{\pi_t} \left[\max_{\phi \in \Phi_{\mathit{Swap}}} r(\pi_t, u, \phi) \right] \le O\left(\sqrt{\frac{n \ln(nT)}{t}} \right)$$

We can compute the approximate minimax equilibrium using Algorithm 2, with per-round running time polynomial in n and $\log t$ for each round $t \in [T]$.

Proof From Theorem 14, the predictions of the canonical algorithm with event set $\mathcal{E}$ have expected bias bounded by:

$$\alpha(T, n_t(E, \pi_t)) = O\left(\ln(nT) + \sqrt{\ln(nT) \cdot n_t(E, \pi_t)} \right)$$

where here we use that $\sum_{t'=1}^{t} E(x_{t'}, p_{t'})^2 \leq n_t(E, \pi_t) \leq t$. Plugging this bound into Theorem 29, we get that the canonical algorithm has swap regret bounded by

$$\mathop{\mathbb{E}}_{\pi_t}\left[\max_{\phi \in \Phi_{\text{Swap}}} r(\pi_t, u, \phi)\right] \leq O\left(\sqrt{\frac{n \ln(nT)}{t}}\right)$$

The running time guarantee follows from Theorem 25, noting that the events $\mathcal{E}$ are disjoint and binary. ■

This recovers (up to low order terms) the optimal swap regret bounds for the experts problem [11, 57], and does so in a nearly "anytime" manner (i.e. with bounds depending on t at every time step $t \leq T$).

Of course, more direct algorithms for obtaining these swap regret bounds are already known—but a strength of our approach is that it can be easily and transparently adapted to various extensions of the setting. Most directly, we can simultaneously ask for unbiasedness with respect to the best response correspondence of multiple utility functions, which allows a single set of predictions to enable multiple decision makers to simultaneously obtain low swap regret. It also allows for many generalizations for a single decision maker. We give an example below in Section F.1.1 — obtaining no *group-wise* swap regret in a setting with contexts, There are a variety of other extensions that are possible as well that we leave as simple exercises (e.g. giving swap regret in the sleeping experts setting, offering group-wise sleeping experts bounds with *adaptive* regret guarantees in the style of [55], or giving the kinds of *transductive* regret bounds of [73] that compete with policies that choose actions as a function of the realized transcript (groupwise and in the sleeping experts setting if desired)).

F.1.1. GROUP-WISE SWAP REGRET

Suppose that in each round we need to choose an expert i_t on behalf of an individual with observable features $x_t \in \mathcal{X}$ that we learn at the beginning of the round (i.e. before we need to make the decision). Individuals may be members of different *groups* $G \subseteq \mathcal{X}$, and we may have a collection of (potentially intersecting) groups $\mathcal{G}$ on which we care about the performance of our algorithm. The groups could represent (e.g.) demographic groups defined by features like race, sex, income, etc, in a context in which one is interested in fairness (as in the literature on subgroup fairness, e.g. [56, 63, 64]), or could represent subsets of the population that are believed to be clinically relevant (e.g. defined by medical history, genotype, etc) in a personalized medicine setting, or anything else.

[10] and [69] gave algorithms for obtaining group-wise *external* regret; here we show how to obtain the stronger guarantee of group-wise swap regret, which we can define as follows:

Definition 60 (Group-wise Φ and Swap-Regret) *Fix a transcript π_T and a group $G \subseteq \mathcal{X}$. Let $T_G(\pi_T) = |\{t \leq T : x_t \in G\}|$ denote the number of rounds in which $x_t \in G$. The regret that a straightforward decision maker with utility function u has with respect to a strategy modification rule $\phi : \mathcal{A} \to \mathcal{A}$ on group G is:*

$$r(\pi_T, u, \phi, G) = \frac{1}{T_G(\pi_T)} \sum_{t: x_t \in G} u(\phi(a_t), y_t) - u(a_t, y_t),$$

where $a_t = \delta_u(p_t)$ for each t. Let Φ be a collection of strategy modification rules and let $\mathcal{G}$ be a collection of groups. We say that a decision maker has $(\Phi, \mathcal{G})$-groupwise regret α for a function

$\alpha : \mathbb{R} \to \mathbb{R}$ if for every $G \in \mathcal{G}$ and for every $\phi \in \Phi$, $r(\pi_T, u, \phi, G) \le \alpha(T_G(\pi_T))$. We say that the decision maker has $\mathcal{G}$-groupwise swap regret α if they have $(\Phi_{Swap}, \mathcal{G})$-groupwise regret α for the set Φ_{Swap} of all strategy modification rules.

To solve this variant of the experts problem, we continue to have prediction space $\mathcal{C} = [-1, 1]^n$, $\mathcal{A} = \{1, \ldots, n\}$, and 1-Lipschitz utility function $u : \mathcal{A} \times \mathcal{C} \to \mathbb{R}$ defined as $u(i, p) = p_i$. The only change is that we now instantiate our prediction algorithm to produce unbiased predictions with respect to the $|\mathcal{G}| \cdot n$ events $\mathcal{E} = \{E_{u,a,G}\}_{a \in \mathcal{A}, G \in \mathcal{G}}$ defined such that for each action $i \in \mathcal{A}$ and $G \in \mathcal{G}$, $E_{u,i,G}(x_t, p_t) = 1$ if and only if $x_t \in G$ and $i = \text{argmax}_{i' \in [n]} p_{t,i}$. We can now apply the following straightforward modification of Theorem 29, adapted to groupwise swap regret:

Theorem 61 *Fix a collection of L-Lipschitz utility functions $\mathcal{U}_L$, an action set $\mathcal{A} = \{1, \ldots, K\}$, a collection of groups $\mathcal{G} \in 2^{\mathcal{X}}$, and any transcript π_T. Let $\mathcal{E} = \{E_{u,a,G} : u \in \mathcal{U}_L, a \in \mathcal{A}, G \in \mathcal{G}\}$ be the set of binary events defined as $E_{u,a,G}(x_t, p_t) = 1$ if and only if $x_t \in G$ and $\delta_u(p_t) = a$. Then if π_T is α-unbiased with respect to $\mathcal{E}$, for every $u \in \mathcal{U}_L$, the straightforward decision maker with utility function u has $\mathcal{G}$-groupwise swap regret at most:*

$$\max_{\phi : \mathcal{A} \to \mathcal{A}} r(\pi_T, u, \phi, G) \le \frac{2L \sum_{a \in \mathcal{A}} \alpha(n_T(E_{u,a,G}, \pi_T))}{T_G(\pi_T)}$$

for every group $G \in \mathcal{G}$. If α is concave, then this bound is at most:

$$\max_{\phi : \mathcal{A} \to \mathcal{A}} r(\pi_T, u, \phi, G) \le \frac{2LK\alpha(T_G(\pi_T)/K)}{T_G(\pi_T)}.$$

The proof requires only additional notation compared to the proof of Theorem 29, and can be found in the Appendix. We can now read off group-wise swap-regret bounds for the machinery we have built up to get the following bound on anytime groupwise swap regret:

Theorem 62 *Fix any collection of groups $\mathcal{G} \subset 2^{\mathcal{X}}$. The canonical Predict-Then-Act Algorithm parameterized with $\mathcal{C} = [-1, 1]^n$ and $\mathcal{E} = \{E_{u,a,G} :, a \in \mathcal{A}, G \in \mathcal{G}\}$ obtains for every $t \le T$ group-wise swap regret at most:*

$$\mathbb{E}_{\pi_t}\left[\max_{\phi \in \Phi_{Swap}} r(\pi_t, u, \phi, G) - \Theta\left(\sqrt{\frac{n \ln(n|\mathcal{G}|T)}{t_G(\pi_t)}}\right)\right] \le 0.$$

We can compute the minimax equilibrium using Algorithm 3, with per-round running time polynomial in n and t on every round $t \in [T]$.

Proof From Theorem 14, the predictions of the canonical algorithm with event set $\mathcal{E}$ have expected bias bounded at round t by:

$$\alpha(T, n_t(E_{u,a,G}, \pi_t)) = O\left(\ln(n|\mathcal{G}|T) + \sqrt{\ln(n|\mathcal{G}|T) \cdot n_t(E_{u,a,G}, \pi_t)}\right).$$

Plugging this bound into Theorem 61 we get that the canonical algorithm has $\mathcal{G}$-groupwise swap regret bounded by

$$\mathbb{E}_{\pi_t}\left[\max_{\phi \in \Phi_{\text{Swap}}} r(\pi_t, u, \phi, G) - \Theta\left(\sqrt{\frac{n \ln(n|\mathcal{G}|T)}{t_G(\pi_t)}}\right)\right] \le 0.$$

The events that we condition on are no longer disjoint, and thus our predictions will take per-round running time that is polynomial, rather than polylogarithmic, in the round's index t. ■

F.2. Subsequence Regret in Online Combinatorial Optimization and Extensive Form Games

In Section D we gave two qualitatively different regret guarantees that follow from making predictions that are unbiased with respect to $\mathcal{E} = \{E_{u,a} : u \in \mathcal{U}, a \in \mathcal{A}\}$, the set of binary events corresponding to straightforward decision makers with utility functions $u \in \mathcal{U}$ taking each action $a \in \mathcal{A}$. This is a natural collection of events to condition on in settings in which the action set $\mathcal{A}$ is modestly sized — but when $\mathcal{A}$ is exponentially large, then in general it is not possible to offer non-trivial guarantees over this collection of events without exponentially large data requirements.

In this section, we show that in the online combinatorial optimization problem, despite the fact that $\mathcal{A}$ is exponentially large (in the number of base actions), it is nevertheless possible to efficiently obtain *subsequence* regret with respect to the subsequences defined by any polynomially large collection of events $\mathcal{E}$. Recall that subsequence regret corresponds to no *external* regret on each subsequence — i.e. no regret to each of the *exponentially* many actions in $\mathcal{A}$, simultaneously on each subsequence $E \in \mathcal{E}$.

The online combinatorial optimization problem is an instance of the online linear optimization problem (defined in Section C.2.2 in the context of deriving our main algorithm via FTPL) defined by a set of base actions $B = \{1, \ldots, n\}$ and a collection of *feasible subsets* of the base actions $\mathcal{D} \subseteq 2^B$. In rounds $t \in \{1, \ldots, T\}$:

1. The algorithm chooses a distribution over feasible action subsets $D_t \in \Delta\mathcal{D}$
2. The adversary chooses a vector of gains $g_t \in [-1, 1]^n$ over the base actions.
3. The algorithm experiences gain $\hat{g}_t = \mathbb{E}_{S_t \sim D_t}[\sum_{i \in S_t} g_{t,i}]$.

For example, if the base actions B correspond to the roads in a road network, the feasible subsets $\mathcal{D}$ correspond to collections of roads that form $s - t$ paths in the underlying network, and the gains correspond to the (negative) road congestions for each edge in the network, then we have the online shortest paths problem [62, 84]. More generally, $\mathcal{D}$ could represent *any* combinatorial structure, such as spanning trees, Hamiltonian paths, or anything else. The Follow-the-Perturbed-Leader algorithm [62] reduces the problem of obtaining efficient *external* regret bounds for combinatorial optimization problems to the offline problem of solving linear optimization problems over $\mathcal{D}$. Here we show how to efficiently get the stronger guarantee of no subsequence regret (defined in Section B.2) for any polynomial collection of events $\mathcal{E}$ by reduction to the offline problem of optimizing over $\mathcal{D}$.

To cast online combinatorial optimization in our framework, we set our prediction space to be $\mathcal{C} = [-1, 1]^n$, our action space to be $\mathcal{A} = \mathcal{D}$, and the decision maker's utility function to be $u(S_t, p_t) = \sum_{i \in S_t} p_{t,i}$, which is linear in p_t as required.

Recall that the events $E \in \mathcal{E}$ can depend on the chosen set S_t that the learner plays at each round. If the payoffs of different *compound actions* S_t were unrelated to one another, then we would have to resort to conditioning on events $E_{u,S}$ as in Section F.1 — i.e. the events that correspond to a downstream decision maker with utility function u playing action S. The difficulty is that in the online combinatorial optimization problem, there are exponentially many actions S, and hence exponentially many such events. Here we take advantage of the linear structure. The idea is that we can condition on events defined by *base actions* $b \in B$. In particular, we condition on the events that the downstream decision maker chooses an action S that *contains* base action b. Although there are as many as 2^n compound actions S, there are only n base actions, and hence n such conditioning events. The linear structure of the payoff makes these events sufficient to guarantee the learner no *external* regret. If we now want the learner to additionally have a no *subsequence* regret guarantee with respect to a collection of arbitrary events $\mathcal{E}$, we instead condition on the intersection of the events $E \in \mathcal{E}$ with the events that the downstream learner plays each of their base actions b, which results in only $n \cdot |\mathcal{E}|$ conditioning events. This is enough to give us efficient algorithms promising no subsequence regret on any polynomially sized collection of subsequences:

Theorem 63 *Fix a base action set $B = \{1, 2, \ldots n\}$, prediction space $\mathcal{C} = [-1, 1]^n$, an action set $\mathcal{D} \subseteq 2^B$, a collection of events $\mathcal{E}$, and utility function $u(S_t, p_t) = \sum_{i \in S_t} p_{t,i}$. Let $\mathcal{I} = \{I_{b,E} : b \in B, E \in \mathcal{E}\}$ be the set of binary events defined as $I_{b,E}(p_t) = 1$ if and only if $b \in \delta_u(p_t)$ and $E(\pi_{t-1}, x_t, p_t) = 1$. Then if π_T is α-unbiased with respect to $\mathcal{I} \cup \mathcal{E}$, the straightforward decision maker with utility function u has $\mathcal{E}$-subsequence regret at most:*

$$\max_{E \in \mathcal{E}, \phi \in \Phi_{Ext}} r(\pi_T, u, \phi, E) \leq \sum_{b \in B} \frac{\alpha(n_T(E, \pi_T)) + \alpha(n_T(I_{b,E}, \pi_T))}{T}$$

The proof of Theorem 63 is in Appendix M. We can read off concrete anytime subsequence regret bounds using the machinery we have built up:

Corollary 64 *Fix any collection of events $\mathcal{E}$. The canonical Predict-Then-Act algorithm parameterized with $\mathcal{C} = [-1, 1]^n, \mathcal{A} = \mathcal{D}, \mathcal{E}' = \mathcal{I} \cup \mathcal{E}$, where $\mathcal{I} = \{I_{b,E} : b \in B, E \in \mathcal{E}\}$, obtains expected subsequence regret at each round $t \leq T$ at most:*

$$\mathop{\mathbb{E}}_{\pi_t}\left[\max_{E \in \mathcal{E}, \phi \in \Phi_{Ext}} r(\pi_t, u, \phi, E)\right] \leq O\left(\frac{n\sqrt{\ln(n|\mathcal{E}|T)}}{\sqrt{t}}\right).$$

We can implement the Predict-Then-Act algorithm using Algorithm 3, with runtime in every round $t \in [T]$ polynomial in n and t.

Remark 65 *We have described the result as if there is a single decision maker. However we can equally well handle the case in which there are many different decision makers j who experience affine gain $g^j_{t,i}(y^t)$ for base action i (as a function of some common state y), and who experience total gain $\hat{g}^j_t(S_t) = \sum_{i \in S} g^j_{t,i}(y^t)$ for compound action S_t. For example, in the context of an online shortest paths problem, y_t could be a vector of congestions on each road segment, but different downstream agents could have different disutilities for delay, could have preferences for or against toll roads, scenic routes, etc, and could have different action sets corresponding to different source-destination pairs in the road network. If we have m such downstream agents, we can produce forecasts that are unbiased according to the events defined in Theorem 63 for* each *of their utility functions, which will have the effect of increasing the running time of the algorithm linearly with m, and increasing the regret guarantee logarithmically in m. The result will be forecasts that yield regret guarantees simultaneously for all of the m downstream agents.*

F.2.1. REGRET IN EXTENSIVE-FORM GAMES

Extensive-form games are a generalization of normal form games in which players may take actions in sequence, have multiple interactions with one another, and reveal some (but possibly not all) information about their actions to one another over the course of their interaction. Many real-world interactions are fruitfully modeled as extensive form games — for example, superhuman poker players were developed by modelling poker as a large extensive form game [12, 13].

An extensive form game can be represented by a game tree, where at each node of the tree, a particular agent plays an action, until a terminal node is reached and utilities for all players are revealed. Players are able to condition on past play in the game, as specified by the *information set*, or a set of nodes for which the revealed past play is the same, capturing the notion of imperfect information.

Recent work in extensive-form game solving has explored the connection between various no-regret dynamics and equilibrium concepts, in attempts to find the largest set of strategy modification functions, or deviations, such that polynomial time convergence to equilibria is still possible. [26, 27] find that minimizing regret defined by a set of trigger deviations and linear swap deviations on sequence-form strategies converge to extensive-form correlated equilibrium (EFCE) and linear correlated equilibrium (LCE) respectively. On the other hand, [74, 75] categorize families of non-linear (with respect to sequence-form strategies) behavioral deviations which converge to equilibria concepts that are subsets or supersets of EFCE.

In this section, we present regret minimization in the extensive-form game setting as a special case of online combinatorial optimization.

Extensive-form Games An extensive-form game of n players $\{1, 2, \dots, n\} \cup \{c\}$ is represented by a tree, which includes a set of nodes $\mathcal{H}$ and a set of directed edges, or actions $\mathcal{A}$. There is a chance player c, representing Nature, who at each of their nodes plays actions with fixed probability determined when the game begins. There are two types of nodes: a *terminal node* $z \in \mathcal{H}$ is a leaf of the tree, while all other non-terminal nodes are referred to as *internal nodes*. Let $\mathcal{Z}$ be the set of all terminal nodes in the game. An *information set* $I \in \mathcal{I}$ is a set of internal nodes such that the information available to the player at these nodes is indistinguishable, i.e. the known history of past play is the same. The *partition function* $H : \mathcal{H} \to \mathcal{I}$ defines this partition of $\mathcal{H}$ into information sets. At each information set $I \in \mathcal{I}$, a single player $i \in [n]$ selects an action from the available actions at that information set, denoted $\mathcal{A}(I)$ (a basic consistency condition requires that the actions available at each node of an information set all be the same, and equal to $\mathcal{A}(I)$). Let $\mathcal{I}_i$ be the set of all information sets at which player i chooses an action. Once a terminal node $z \in \mathcal{Z}$ of the tree is reached, the game ends and the vector of gains $\{g_1(z), g_2(z), \dots g_n(z)\}$ for all players is revealed.

An *assignment function* $\rho : \mathcal{I} \to [n] \cup \{c\}$ takes as input an information set $I \in \mathcal{I}$ and returns the player who takes an action at that information set. We overload ρ to also return the acting player for single nodes in $\mathcal{H}$. Note that for any two nodes h_1 and h_2 in the same information set, $\rho(h_1) = \rho(h_2)$. The *successor function* $c : \mathcal{H} \times \mathcal{A} \to \mathcal{H}$ maps any node-action pair (h, a) to the node resulting from taking action a at h. For any two information sets $I, I' \in \mathcal{I}_i$, we write $I \prec I'$ if there exists a path from any node $h \in I$ to any node $h' \in I'$.

Strategy Representations and Reachability A standard extensive-form strategy representation is the *behavioral strategy*, where at each one of the player's information sets, a probability is assigned to each of the actions that the player may take.

Definition 66 *A* behavioral strategy $\sigma_i : \mathcal{I} \to \Delta(\mathcal{A}(\mathcal{I}))$ *for player i assigns a valid probability distribution over actions at each information set $I \in \mathcal{I}_i$. Let $\sigma_i(I, a)$ be the probability action a is played in the distribution over information set I. A deterministic behavioral strategy, denoted s_i, assigns probability 1 to some action at each information set $I \in \mathcal{I}$. Let S_i be the set of all* *deterministic behavioral strategies of player i, and Σ_i be the set of all (randomized) behavioral strategies of player i.*

One disadvantage of the behavioral strategy representation is that it results in nonconvexity in the computation of important quantities, such as the probability of reaching any terminal node. In our framework, we use a terminal node-based representation of deterministic strategies, representing each strategy by the subset of terminal nodes it makes reachable, which is a sufficient statistic to compute the payoff of that strategy. This representation relies on the concept of *reachability*. Reachability can be reasoned about as a binary-valued property (whether or not there is a chance a node is reached) or as a probability (the exact probability of reaching a node). We discuss the former in the context of a learner's strategy σ_i, and the latter with respect to a collection of opponents' strategies σ_{-i}:

Definition 67 *A node $h \in \mathcal{H}$ (or information set $I \in \mathcal{I}$) is made* reachable by player i under behavioral strategy σ_i *if, for a path $P_h = \{(I_1, a_1), (I_2, a_2), \ldots, (I_K, a_K)\}$ from the root to h (or I), for all $k \in [K]$ such that $\rho(I_k) = i$, $\sigma_i(I_k, a_k) > 0$.*

Definition 68 *The* reach probability *r_h (or r_I) of a node $h \in \mathcal{H}$ (or information set $I \in \mathcal{I}$) under opponents' strategies σ_{-i} is*

$$r_h = \prod_{k \in [K], \rho(I_k) \neq i} \sigma_{\rho(I_k)}(I_k, a_k)$$

given a path $P_h = \{(I_1, a_1), (I_2, a_2), \ldots, (I_K, a_K)\}$ from the root to h (or I).

With these concepts, we can define our notion of a "leaf-form strategy", which represents any behavioral strategy σ_i by an indicator vector specifying which subset of leaf nodes are made reachable by σ_i. For the purposes of this work, it is sufficient to restrict our attention to the collection of pure leaf-node strategies induced by deterministic behavioral strategies.

Definition 69 *A* leaf-form strategy representation *of a deterministic behavioral strategy s_i is a binary vector $\pi \in \{0, 1\}^{|\mathcal{Z}|}$ indexed by all terminal nodes $z \in \mathcal{Z}$, with $\pi_z = 1$ if z is reachable under s_i. Let Π_i be the space of all leaf-form strategies of player i induced by the set of all deterministic behavioral strategies S_i.*

We can similarly define a representation of opponents' strategies in terms of the probabilities of reaching each of the terminal nodes, weighted by the learner's payoff at each of them:

Definition 70 *Fix a collection of opponents' strategies σ_{-i}. The* payoff-weighted reachability vector *v induced by σ_{-i} is a vector $v \in \mathbb{R}^{|\mathcal{Z}|}$ indexed by terminal nodes $z \in \mathcal{Z}$ with $v_z = r_z \cdot g_i(z)$, where $g_i(z)$ is the payoff for player i at z.*

For a strategy profile σ where player i is playing a deterministic strategy, if a node is made reachable under σ_i, then the node's reach probability under σ_{-i} is the exact probability of reaching it. Therefore, for any such strategy profile, we can compute the expected payoff for the learner (playing a deterministic strategy) as the inner product of a corresponding leaf-form strategy and payoff-weighted reachability vector:

Lemma 71 *Fix a learner's deterministic behavioral strategy s_i and opponent's strategies σ_{-i}. Let π be the leaf-form strategy representing s_i. Let v be the* payoff-weighted reachability vector *induced by σ_{-i}. Then, the expected utility for the learner is $\langle \pi, v \rangle$.*

Oracle-Efficient Optimization over Leaf-form Strategies The learning algorithms we present will be efficient reductions to the problem of solving linear optimization problems over the space of leaf-form strategies. In the remaining exposition, we will assume that we have an oracle for solving this best response problem. Whenever an extensive form game satisfies the properties of *perfect recall* and *path recall* for the learner, a best-response oracle can be efficiently implemented using backwards induction — we give details in Appendix L.1.

Definition 72 *A* best response oracle *for a game G is an algorithm that, when given as input game G, player i, and vector of values $v \in \mathbb{R}^{|\mathcal{Z}|}$, returns $\pi^* = \text{argmax}_{\pi \in \Pi_i} \langle \pi, v \rangle$.*

Extensive-form Games as Online Combinatorial Optimization We can now cast learning in extensive-form games as an instance of online combinatorial optimization by viewing each player as a learner operating with the leaf-form representation of their strategies, maximizing their expected payoff against an adversary, who chooses a payoff-weighted reachability vector at each round. In the setup of online combinatorial optimization, for rounds $t \in \{1, 2, \ldots, T\}$:

1. The learner (representing player i) picks a leaf-form strategy $\pi_t \in \Pi_i$.

2. The adversary picks a payoff-weighted reachability vector v_t fixing any $\sigma_{-i} \in \Sigma_{-i}$.

3. The learner experiences a gain of $g_t = \langle \pi_t, v_t \rangle$.

The algorithm for achieving subsequence regret for the learner in this framework proceeds by at each round t first making a prediction $p_t \in \mathbb{R}^{|\mathcal{Z}|}$ of the opponents' payoff-weighted reachability vector for that round. The straightforward decision maker then picks a leaf-form strategy π_t that maximizes their expected payoff with respect to p_t, using the best response oracle, after which the true payoff-weighted reachability vector v_t is revealed. We can obtain $\mathcal{E}$-subsequence regret bounds of the form described in Theorem 63, where the action space is $\mathcal{A} = \Pi_i$, the set of all leaf-form strategies, using base actions B, the set of all one-hot vectors of length $|\mathcal{Z}|$.

As written, the running time of our algorithm is linear in the number of terminal nodes in the game tree. This comes from the fact that we have represented the payoff of strategies in the extensive form game as linear functions over the payoff obtainable at each terminal node. More generally, however, if payoffs can be described as linear functions over a lower dimensional space, then the same algorithm can be run with running time dependence on this lower dimension. In Appendix L.2 we describe conditions under which the running time can be taken to depend only on the number of information sets of a player, rather than the number of terminal nodes in the game tree, which can sometimes be a large improvement.

Informed Causal Deviations Choosing different sets of events $\mathcal{E}$ will define different classes of strategy deviations. Our guarantees will allow us to state that we can obtain no-regret over the set of all deviations that map the periods for which an event E has occurred ($E(x_t, p_t) = 1$) to any another fixed strategy. In the following, we show how to pick a small collection of events $\mathcal{E}$ to recover the well-studied notion of regret to *informed causal deviations*, which lead to convergence to extensive-form correlated equilibrium [86].

We show that regret to informed causal deviations [23, 45] is bounded as a special case of regret to subsequence deviations defined by a polynomial number of events. Informed causal deviations of a strategy allow a player to consider, in hindsight, the best strategy they could have taken given that they reach a particular information set (among their own information sets) and take a particular action at that information set, known as the trigger sequence. A formal definition is given below:

Definition 73 *Fix an information set $I' \in \mathcal{I}_i$, subsequent action $a' \in \mathcal{A}(I')$, and strategy s'_i. An* informed causal deviation *ϕ of strategy $s_i \in S_i$ returns a strategy such that at each information set $I \in \mathcal{I}$,*

$$\phi(s_i) = \begin{cases} s'_i(I), & I \succeq I', s_i(I') = a' \\ s_i(I), & \textit{otherwise} \end{cases}$$

Let $\Phi_{causal} = \{\phi_{I,a,s} : I \in \mathcal{I}_i, a \in \mathcal{A}(I), s' \in S_i\}$ be the set of all informed causal deviations.

We set up our algorithm to make predictions that are unbiased with respect to the $|\mathcal{I}| \cdot | \max_{I \in \mathcal{I}} \mathcal{A}(I)|$ events defined as $\mathcal{E} = \{E_{I,a} : I \in \mathcal{I}_i, a \in \mathcal{A}(I)\}$. The event $E_{I,a}(p_t) = 1$ if and only if the strategy played at time t makes the information set I reachable and plays a at information set I. We now directly apply Theorem 63 to obtain the following theorem:

Theorem 74 *Fix an extensive-form game G. Let the base action set B be the set of all one-hot vectors b_z with dimension $|\mathcal{Z}|$. Fix the action space of the learner as the leaf-form strategy space $\Pi_i \subseteq 2^B$. Fix events defined as $\mathcal{E} = \{E_{I,a} : I \in \mathcal{I}_i, a \in \mathcal{A}(I)\}$, and fix any transcript π_T. Let $Q = \{q_{b_z,E_{I,a}} : b_z \in B, E_{I,a} \in \mathcal{E}\}$ be the set of binary events defined as $q_{b_z,E_{I,a}}(p_t) = 1$ if and only if $b_z \in \delta_u(p_t)$ and $E_{I,a}(p_t) = 1$. Then if π_T is α-unbiased with respect to $Q \cup \mathcal{E}$, the straightforward decision maker has $\mathcal{E}$-subsequence regret at most:*

$$\max_{E \in \mathcal{E}, \phi \in \Phi_{Ext}} r(\pi_T, u, \phi, E) \leq \sum_{b_z \in B} \frac{(\alpha(n_T(E_{I,a}, \pi_T)) + \alpha(n_T(q_{b_z,E_{I,a}}, \pi_T)))}{T}.$$

Corollary 75 *Fix an extensive-form game G and collection of events $\mathcal{E} = \{E_{I,a} : I \in \mathcal{I}_i, a \in \mathcal{A}(I)\}$. The canonical Predict-Then-Act Algorithm parameterized with $\mathcal{C} = \mathbb{R}^{|\mathcal{Z}|}, \mathcal{A} = \Pi_i, \mathcal{E}' = Q \cup \mathcal{E}$, where $Q = \{q_{b_z,E_{I,a}} : b_z \in B, E_{I,a} \in \mathcal{E}\}$, obtains expected subsequence regret at each round $t \leq T$ at most:*

$$\mathop{\mathbb{E}}_{\pi_t}\left[\max_{E \in \mathcal{E}, \phi \in \Phi_{Ext}} r(\pi_t, u, \phi, E)\right] \leq O\left(\frac{|\mathcal{Z}| \ln(|\mathcal{Z}||\mathcal{I}|| \max_{I \in \mathcal{I}} \mathcal{A}(I)|T)}{\sqrt{t}}\right).$$

From this guarantee, we can show that having no subsequence regret implies having no causal regret.

Theorem 76 *Fix an extensive-form game G and transcript π_T. If the algorithm has $\mathcal{E}$-subsequence regret with event set $\mathcal{E} = \{E_{I,a} : I \in \mathcal{I}_i, a \in \mathcal{A}(I)\}$ bounded by:*

$$\max_{E \in \mathcal{E}, \phi \in \Phi_{Ext}} r(\pi_T, u, \phi, E) \leq \alpha,$$

then we have that causal regret, or Φ-regret with respect to the set Φ_{causal}, is bounded by:

$$\max_{\phi \in \Phi_{causal}} r(\pi_T, u, \phi) \leq \alpha.$$

The proof can be found in Appendix L. An immediate consequence of this result is if all players minimize subsequence regret with respect to the events defined in Theorem 74, the empirical frequency of play will converge to a solution subset of the set of extensive-form correlated equilibria (EFCE).

In general, the kinds of subsequence deviations we define are incomparable to the behavioral deviation landscape defined by [74] or linear swap deviations as studied by [26]. It appears that none of these three families of deviations subsumes any of the others. We note that the machinery we have developed for subsequence deviations is easily able to take into account external context x and past history, and so can be used to give regret conditional on features that may be relevant but are not represented within the action space of the game.

Appendix G. Additional Details from Section C.2.1

We solve for an ϵ-approximate solution of linear program 4 using a *weak separation oracle*, using an approximate version of the Ellipsoid algorithm.

Definition 77 *For any $\epsilon > 0$ and any convex set S, let*

$$S^{+\epsilon} = \{p : ||p - q||_2 \leq \epsilon \quad \textit{for some } q \in S\} \quad S^{-\epsilon} = \{x : B_2(x, \epsilon) \subseteq S\}$$

be the positive and negative ϵ-approximate sets of S, where $B_2(x, r)$ is a ball of radius r under the ℓ_2 norm.

Definition 78 *A* weak separation oracle *for a convex set S is an algorithm that, when given input $\psi \in \mathbb{Q}^d$ and positive $\epsilon \in \mathbb{Q}$, confirms that $\psi \in S^{+\epsilon}$ if true, and otherwise returns a hyperplane $a \in \mathbb{Q}^d$ such that $||a||_\infty = 1$ and $\langle a, \psi \rangle \leq \langle a, \psi' \rangle + \epsilon$ for all $\psi' \in S^{-\epsilon}$.*

We express a separation oracle for linear program 4 as the convex program that solves for the most violated constraint given a candidate solution ψ, which is simply the best response problem for the maximization player in minimax problem 3. This is the problem of maximizing a d-variable linear function over the convex set $\mathcal{C}$. To make sure that we can control the bit complexity of the constraint returned by the separation oracle we round the coordinates of the constraint $a \in \mathbb{R}^d$ output by the separation oracle to a rational-valued vector within $\pm\frac{\epsilon}{2}$ of the exact solution by truncating each coordinate of a to $\log(\frac{1}{\epsilon})$ bits.

Definition 79 *A solution $\psi \in S^{+\epsilon}$ is* ϵ-weakly optimal *if, given $\epsilon > 0$, $\mathbb{E}_{p\sim\psi}[u(p, y)] \leq \mathbb{E}_{p\sim\psi'}[u(p, y)] + \epsilon$ for all $\psi' \in S^{-\epsilon}$ and for all y.*

For an ϵ-approximate solution to minimax problem 3, it suffices to find an ϵ-weakly optimal solution to linear program 4, which we can do using the Ellipsoid method. However, the solution to the weak optimization may not even be a valid probability distribution (since it only approximately satisfies the constraints) – in this case, we can project our infeasible solution back to feasibility. We use the simplex Euclidean projection algorithm given by [18] to project the candidate solution back to a feasible region and show that this projected feasible solution is still ϵ-approximately optimal.

Theorem 25 *Given a polynomial-time separation oracle for $\mathcal{C}$, for any $\epsilon > 0$, there exists an algorithm (Algorithm 2) that returns an ϵ-approximately optimal solution ψ_t^* to minimax problem 2 and runs in time polynomial in d, $|\mathcal{E}|, \log(\frac{1}{\epsilon})$.*

Proof Linear program 4 encodes minimax problem 3. To solve LP 4, we use the Ellipsoid algorithm, which gives an approximate solution in polynomial time under the following conditions:

Theorem 80 ([47], Theorem 4.4.7) *Given a weak separation oracle over convex constraint set S and $\epsilon > 0$, the Ellipsoid algorithm finds a ϵ-weakly optimal solution over S in time polynomial in the bit complexity of the constraints returned by the separation oracle, the bit complexity of the objective function, and the bit complexity of ϵ.*

Fix some $\epsilon > 0$. Let S be the constraint set, which are a set of linear constraints over a convex compact set (i.e. $y \in \mathcal{C}$) and constraints enforcing a probability simplex (i.e. $\psi \in \Delta(\mathcal{P})$), implying that S is a convex set. Let $\epsilon' = \frac{\epsilon}{2C\sqrt{|\mathcal{E}|}}$. Given an exact separation oracle over $\mathcal{C}$, preserving $\log(\frac{1}{\epsilon'})$ bits of the most violated constraint given by the separation oracle and rounding to a rational number yields an rational ϵ'-approximate most violated constraint, which satisfies the conditions for a weak separation oracle. Thus, we can find an ϵ'-weakly optimal solution (γ', ψ') to minimax problem 3, where $\psi' \in S^{+\epsilon}$. In the case that ψ' is a valid probability distribution, we have found an ϵ-approximate optimal solution $\psi_t^* = \psi'$.

Otherwise, ψ' may violate conditions for a valid probability distribution if the linear constraints do not constrain the feasible set (i.e. $S = \Delta(P)$). Since $\psi' \in S^{+\epsilon}$, there exists some $\psi^\epsilon \in S$ such that $||\psi^\epsilon - \psi'|| \leq \epsilon'$. We find this point ψ^ϵ via the simplex projection algorithm in [18].

We show that this projection back to a feasible probability distribution still leaves us with an ϵ-approximately optimal solution. Let $u_t(p_t^*, y)$ be the $|\mathcal{E}|$-dimensional vector such that each coordinate E has entry $u_t(p_t^{*,E}, y)$. First, we show that $|u_t(p_t^{*,E}, y)|$ is bounded by $C = 2\max_{y\in\mathcal{C}} ||y||_\infty$ for $E \in \mathcal{E}$:

$$\begin{aligned}
|u(p_t^{*,E}, y)| &\leq \sum_{i=1}^{d} \sum_{\sigma\in\{-1,1\}} \sum_{E\in\mathcal{E}} q_{t,(i,\sigma,E)} \cdot |\sigma| \cdot E(x_t, p_t^{*,E}) \cdot |p_{t,i}^{*,E} - y_i| \\
&\leq \sum_{i=1}^{d} \sum_{\sigma\in\{-1,1\}} \sum_{E\in\mathcal{E}} q_{t,(i,\sigma,E)} \cdot |p_{t,i}^{*,E} - y_i| \\
&\leq \sum_{i=1}^{d} \sum_{\sigma\in\{-1,1\}} \sum_{E\in\mathcal{E}} q_{t,(i,\sigma,E)} \cdot C \\
&= C
\end{aligned}$$

where we used that $E(x_t, p_t^{*,E}) \leq 1$ and $|\sigma| = 1$, and that $q \in \Delta(2d|\mathcal{E}|)$, implying it must sum to 1. From this, we find that $||u_t(p_t^*, y)||_2 \leq \sqrt{C_1^2 + \ldots + C_{|\mathcal{E}|}^2} \leq C\sqrt{|\mathcal{E}|}$.

Next, by continuity of inner product, given $\epsilon > 0, y \in \mathcal{C}$, there exists $\delta > 0$ such that $||\psi^\epsilon - \psi'|| \leq \delta$ implies that $||\mathbb{E}_{p\sim\psi^\epsilon}[u_t(p,y)] - \mathbb{E}_{p\sim\psi'}[u_t(p,y)]|| \leq \epsilon$. By Cauchy-Schwarz, we can bound the difference between the expectations as follows:

$$\begin{aligned}
||\mathop{\mathbb{E}}_{p\sim\psi^\epsilon}[u_t(p,y)] - \mathop{\mathbb{E}}_{p\sim\psi'}[u_t(p,y)]||_2 &= \langle \psi^\epsilon - \psi^*, u_t(p_t^*, y)\rangle \\
&\leq ||\psi_t^\epsilon - \psi'||_2 \cdot ||u_t(p_t^*, y)||_2 \\
&\leq \delta \cdot ||u_t(p_t^*, y)||_2 \\
&\leq \delta \cdot C\sqrt{|\mathcal{E}|}.
\end{aligned}$$

Thus, using $\psi_t^* = \psi^\epsilon$ as the solution and setting $\delta = \epsilon'$ gives us an solution that is $\epsilon' \cdot C\sqrt{|\mathcal{E}|} + \epsilon' = \frac{\epsilon}{2} + \frac{\epsilon}{2C\sqrt{|\mathcal{E}|}} \leq \epsilon$ approximate. By Lemma 24, any optimal solution to minimax problem 3 is an optimal solution to minimax problem 2, so we must have that ψ_t^* is an ϵ-approximate solution to minimax problem 2.

Now we consider the runtime of the algorithm. In order for LP 4 to be well-formulated, we first solve $|\mathcal{E}|$ convex programs (one for each $p^{*,E}$), which takes time polynomial in d. Now, consider the bit complexity of the constraints. For the inequality constraints, the bit complexity of each constraint bounding the objective function is given by the bit complexity of $\mathbb{E}_{p\sim\psi}[u(p,y)]$. Each coefficient of $\psi(p_t^{*,E})$ is $u_t(p_t^{*,E}, y)$, which is bounded by C from above. Since there are $|\mathcal{E}|$ variables in this constraint, the maximum bit complexity of any constraint is bounded by $O(\log(C|\mathcal{E}|))$. Similarly, the objective function has polynomial bit complexity on the scale of $O(\log(C|\mathcal{E}|))$. Finally, ϵ has a bit complexity of $\log(\frac{1}{\epsilon})$. The simplex projection algorithm has quadratic runtime in the dimension of the vector, which takes $O(|\mathcal{E}|^2)$ time.

Thus, the runtime of the algorithm is polynomial in $d, |\mathcal{E}|, \log(C|\mathcal{E}|)$, and $\log(\frac{1}{\epsilon})$. ■

Appendix H. Proofs from Section C.2.1

Lemma 81 *Simultaneously for all $y \in \mathcal{C}$, we have:*

$$p_t^{*,E} \in \operatorname*{argmin}_{p:E(\pi_{t-1},x_t,p)=1} u_t(p,y).$$

Proof The constraint that $E(\pi_{t-1}, x_t, p) = 1$ together with the fact that the set of events $\mathcal{E}$ is disjoint and binary implies that for all other events $E' \in \mathcal{E}$, $E'(\pi_{t-1}, x_t, p) = 0$. For any p such that $E(\pi_{t-1}, x_t, p) = 1$, we therefore have that $u_t(p, y)$ reduces to:

$$u_t(p,y) = \left(\sum_{i=1}^{d}\sum_{\sigma\in\{-1,1\}} q_{t,(i,\sigma,E)}\cdot\sigma\cdot p_i - q_{t,(i,\sigma,E)}\cdot\sigma\cdot y_i\right)$$

But in this expression, the p terms have no interaction with the y terms, and hence we have that for any y:

$$\begin{aligned}\operatorname*{argmin}_{p:E(\pi_{t-1},x_t,p)=1} u_t(p,y) &= \operatorname*{argmin}_{p:E(\pi_{t-1},x_t,p)=1}\left(\sum_{i=1}^{d}\sum_{\sigma\in\{-1,1\}} q_{t,(i,\sigma,E)}\cdot\sigma\cdot p_i - q_{t,(i,\sigma,E)}\cdot\sigma\cdot y_i\right)\\ &= \operatorname*{argmin}_{p:E(\pi_{t-1},x_t,p)=1}\left(\sum_{i=1}^{d}\sum_{\sigma\in\{-1,1\}} q_{t,(i,\sigma,E)}\cdot\sigma\cdot p_i\right)\\ &= p_t^{*,E}\end{aligned}$$

■

Lemma 82 *Fix any optimal solution ψ_t^* to minimax problem 3. Then ψ_t^* is also an optimal solution to minimax problem 2.*

Proof We first observe that minimax problem 3 is only a more constrained problem for the minimization player than minimax problem 2, as $\mathcal{P}_t \subset \mathcal{C}$. Thus it suffices to show that given a solution $\hat{\psi}_t$ for minimax problem 2, we can transform it into a new solution ψ_t such that:

1. ψ_t has support only over points in $\mathcal{P}_t$, and
2. For all $y \in \mathcal{C}$, $\mathbb{E}_{p_t \sim \psi_t}[u(p_t, y)] \geq \mathbb{E}_{p_t \sim \hat{\psi}_t}[u(p_t, y)]$.

Given $\hat{\psi}_t$, we construct ψ_t as follows: for each event E, we take all of the weight that $\hat{\psi}_t$ places on points p such that $E(\pi_{t-1}, x_t, p) = 1$, and place that weight on $p_t^{*,E} \in \mathcal{P}_t$:

$$\psi_t(p_t^{*,E}) = \hat{\psi}_t(\{p : E(\pi_{t-1}, x_t, p) = 1\})$$

By construction ψ_t has support over points in $\mathcal{P}_t$. It remains to show that ψ_t has objective value that is at least as high as $\hat{\psi}_t$ for every $y \in \mathcal{C}$:

$$\begin{aligned}
\mathop{\mathbb{E}}_{p_t \sim \hat{\psi}_t}[u(p_t, y)] &= \sum_{E \in \mathcal{E}} \Pr_{p_t \sim \hat{\psi}_t}[E(\pi_{t-1}, x_t, p_t) = 1] \mathop{\mathbb{E}}_{p_t \sim \hat{\psi}_t}[u(p_t, y) | E(\pi_{t-1}, x_t, p_t) = 1] \\
&\leq \sum_{E \in \mathcal{E}} \Pr_{p_t \sim \hat{\psi}_t}[E(\pi_{t-1}, x_t, p_t) = 1] u(p_t^{*,E}, y) \\
&= \mathop{\mathbb{E}}_{p_t \sim \psi_t}[u(p_t, y)]
\end{aligned}$$

The inequality follows from Lemma 23. ■

Appendix I. Proofs From Section C.2.2

Lemma 83 *Playing FTPL for T' rounds with perturbation parameter $\delta = \sqrt{\frac{2d}{T'}}$, for a learner with decision space $\mathcal{C}$ and adversary with state space $\mathcal{S} = \{s_t(p) \mid p \in \mathcal{C}\}$ yields a regret bound:*

$$R_{T',y} \leq \sqrt{2dC^2T'}$$

for all $y \in \mathcal{C}$, where d is the dimension of all vectors in $\mathcal{C}$ and $C = 2\max_{y \in \mathcal{C}} \|y\|_\infty$.

Proof Recall from Theorem 26 that regret at round T' for any decision $y \in \mathcal{C}$ is bounded by:

$$R_{T',y} \leq 2\sqrt{\Delta K A T'}$$

when we choose $\delta = \sqrt{\frac{\Delta}{KAT'}}$, where $K = \sup_{y,p \in \mathcal{C}} |\langle y, s_t(p) \rangle|$, $A = \sup_{p \in \mathcal{C}} \|s_t(p)\|_1$ and $\Delta = \sup_{y_1, y_2 \in \mathcal{C}} \|y_1 - y_2\|_1$.

From the definition of state vectors in Equation 5, for any $p \in \mathcal{C}$, we have that

$$\begin{aligned}
\|s_t(p)\|_1 = \sum_{i=1}^d |s_{t,i}(p)| = \sum_{i=1}^d \left| \sum_{\sigma \in \{-1,1\}} \sum_{E \in \mathcal{E}} q_{t,(i,\sigma,E)} \cdot \sigma \cdot E(x_t, p) \right| \\
\leq \sum_{i=1}^d \sum_{\sigma \in \{-1,1\}} \sum_{E \in \mathcal{E}} \left| q_{t,(i,\sigma,E)} \cdot \sigma \cdot E(x_t, p) \right| \\
= \sum_{i=1}^d \sum_{\sigma \in \{-1,1\}} \sum_{E \in \mathcal{E}} q_{t,(i,\sigma,E)} \cdot |\sigma \cdot E(x_t, p)|
\end{aligned}$$

$$\leq \sum_{i=1}^{d} \sum_{\sigma \in \{-1,1\}} \sum_{E \in \mathcal{E}} q_{t,(i,\sigma,E)} = 1$$

since $q_t \in \Delta[2d|\mathcal{E}|]$ and by definition, $|\sigma \cdot E(x_t, p)| \leq 1$. Thus, we can set $A = 1$. Since $\|y\|_\infty \leq C/2$, we similarly have:

$$|\langle y, s_t(p)\rangle| \leq \sum_{i=1}^{d} |y_i \cdot s_{t,i}(p)| \leq \frac{C}{2} \cdot \sum_{i=1}^{d} \left| \sum_{\sigma \in \{-1,1\}} \sum_{E \in \mathcal{E}} q_{t,(i,\sigma,E)} \cdot \sigma \cdot E(x_t, p) \right| \leq \frac{C}{2}$$

for all $y, p \in \mathcal{C}$, so $K = C/2$. Finally, since $|y_{i,1} - y_{i,2}| \leq C$ for all $y_1, y_2 \in \mathcal{C}$, it is clear that $\Delta \leq dC$.

Note that for this setting, we have indeed set δ to the desired value, since

$$\sqrt{\frac{\Delta}{KAT'}} = \sqrt{\frac{2dC}{CT'}} = \sqrt{\frac{2d}{T'}}$$

and now substituting the relevant quantities into the regret bound,

$$R_{T',y} \leq 2\sqrt{dC \cdot C/2 \cdot 1 \cdot T'} = \sqrt{2dC^2T'}$$

■

Algorithm 3 for achieving unbiased predictions makes the implicit assumption that the expected value of FTPL's output is efficiently computable at each round. In conditions where this is not true, we can approximate the expectation via a sampling procedure over the distribution with respect to which we are taking the expectation, and use this approximation to achieve similar results. We make use of the following concentration inequalities:

Theorem 84 (Hoeffding's Inequality) *Let $\{X_j\}_{j=1}^n$ be independent random variables such that $X_j \in [a, b]$ for each $j \in [n]$. Let $S_n = \sum_{j=1}^n X_j$ be their sum. Then, for any $\epsilon > 0$,*

$$\Pr\left[|S_n - \mathbb{E}[S_n]| \geq \epsilon\right] \leq 2\exp\left(-\frac{2\epsilon^2}{n(b-a)^2}\right)$$

Corollary 85 *Let $\{X_j\}_{j=1}^n$ be independent random variables such that $X_j \in [a, b]$ for each $j \in [n]$. Let $\overline{X} = \frac{1}{n}\sum_{j=1}^n X_j$ be the average of their sum. Then, for any $\epsilon > 0$,*

$$\Pr\left[|\overline{X} - \mathbb{E}[\overline{X}]| \geq \epsilon\right] \leq 2\exp\left(-\frac{2n\epsilon^2}{(b-a)^2}\right)$$

Lemma 86 *Given a sample $\{y_\tau^j\}_{j=1}^n \sim D_\tau^n$, define $p_\tau = \sum_{j=1}^n y_\tau^j$ as the average of the sampled values. For $n = \frac{C^2}{2\epsilon_0^2}\ln\left(\frac{2d}{\delta_0}\right)$, we have that with probability $1 - \delta_0$*

$$|E_{y \sim D_\tau}[y_i] - p_{\tau,i}| < \epsilon_0$$

simultaneously for all $i \in [d]$.

Proof For each $i \in [d]$, the set of values $\{y^j_{\tau,i}\}_{j=1}^n$ are independent draws from the same distribution such that $y^j_{\tau,i} \in [-C/2, C/2]$ for all $i \in [d], j \in [n]$. By direct application of Corollary 85,

$$Pr\left[|\mathbb{E}[y_i] - p_{\tau,i}| \geq \epsilon_0\right] \leq 2\exp\left(-\frac{2n\epsilon_0^2}{C^2}\right)$$

Taking a union bound,

$$\Pr\left[|\mathbb{E}[y_1] - p_{\tau,1}| \geq \epsilon_0 \bigcup |\mathbb{E}[y_2] - p_{\tau,2}| \geq \epsilon_0 \bigcup \cdots \bigcup |\mathbb{E}[y_d] - p_{\tau,d}| \geq \epsilon_0\right] \leq 2d\exp\left(-\frac{2n\epsilon_0^2}{C^2}\right)$$
$$\implies \Pr\left[|\mathbb{E}[y_1] - p_{\tau,1}| < \epsilon_0 \bigcap |\mathbb{E}[y_2] - p_{\tau,2}| < \epsilon_0 \bigcap \cdots \bigcap |\mathbb{E}[y_d] - p_{\tau,d}| < \epsilon_0\right] \geq 1 - 2d\exp\left(-\frac{2n\epsilon_0^2}{C^2}\right)$$

With our choice of n, $2d\exp\left(-\frac{2n\epsilon_0^2}{C^2}\right)$ simplifies to δ_0, completing the proof. ■

Lemma 86 details a sampling procedure to select p_τ that is close to $\mathbb{E}_{y_\tau \sim D_\tau}[y_\tau]$ with high probability. We can use this to slightly modify Algorithm 3 to return a strategy for the learner with similar guarantees, but which doesn't make any assumptions on our ability to compute $\mathbb{E}_{y \sim D_\tau}[y]$. This is captured by the following Algorithm 6:

```
Initialize $s_t(p_0)$ to 0.
Set $T' = \frac{8dC^2}{\epsilon'^2}$, $\delta = \sqrt{\frac{2d}{T'}}$, $n = \frac{2C^2}{\epsilon'^2}\ln\left(\frac{2dT'}{\delta'}\right)$
Fix distribution $\mathcal{Z} = \text{Unif}[0, \frac{1}{\delta}]^d$.
for $\tau = 1, \ldots, T'$ do
  for $j = 1, \ldots, n$ do
    Sample $z^j_\tau \sim \mathcal{Z}$
    Compute $y^j_\tau = M\left(\sum_{k=0}^{\tau-1} s_t(p_k) + z^j_\tau\right)$
  Set $p_\tau = \frac{1}{n}\sum_{j=1}^n y^j_\tau$
  Compute $s_t(p_\tau)$ as defined in Equation 5.
Define $\overline{p}$ as the uniform distribution over the sequence $(p_1, p_2, \cdots, p_{T'})$.
return $\overline{p}$
```
Algorithm 6: `Get-Approx-Equilibrium-Sampling`(t, ϵ', δ')

Theorem 87 *For any $t \in [T]$ and $\epsilon', \delta' > 0$, with probability $1 - \delta'$, Algorithm 6 returns a distribution over actions $\overline{p}$ which is an ϵ'-approximate minimax equilibrium strategy for the zero-sum game with objective u_t.*

The proof of Theorem 87 is similar to the proof of Theorem 27. The primary difference is that we must now control and keep track of sampling error that comes from estimating $\mathbb{E}_{z \in \mathcal{Z}}\left[M\left(\sum_{k=0}^{\tau-1} s_t(p_k) + z\right)\right]$ from samples, which we do with standard concentration inequalities.

Proof We bound the adversary's maximum sum of scores in hindsight with respect to u_t:

$$\max_{y \in \mathcal{C}} \sum_{\tau=1}^{T'} u_t(p_\tau, y) \leq \left(R_{T',y^*} + \sum_{\tau=1}^{T'} \mathop{\mathbb{E}}_{y \sim D_\tau}\left[u'_t(p_\tau, y)\right]\right) + \sum_{\tau=1}^{T'} \langle p_\tau, s_t(p_\tau)\rangle \quad (9)$$

where

$$y^* = \underset{y \in \mathcal{C}}{\text{argmax}} \sum_{\tau=1}^{T'} u'_t(p_\tau, y^*) = \underset{y \in \mathcal{C}}{\text{argmax}} \sum_{\tau=1}^{T'} u_t(p_\tau, y).$$

Now, we choose p_τ as the average of $n = \frac{2C^2}{\epsilon'^2} \ln\left(\frac{2dT'}{\delta'}\right)$ independently drawn samples from D_τ, which ensures we satisfy the guarantee detailed in Lemma 86. By our choice of n, we have $\epsilon_0 = \epsilon'/2$ and $\delta_0 = \delta'/T'$. Since now p_τ is an estimate of $\mathbb{E}_{y \sim D_\tau}[y]$ rather than the exact expectation, for any specific $\tau \in [T']$ we can only bound the expected utility with high probability:

$$\begin{aligned}
\mathop{\mathbb{E}}_{y \sim D_\tau}[u_t(p_\tau, y)] = \left\langle \left(\mathop{\mathbb{E}}_{y \sim D_\tau}[y] - p_\tau \right), s_t(p_\tau) \right\rangle &\leq \left| \left\langle \left(\mathop{\mathbb{E}}_{y \sim D_\tau}[y] - p_\tau \right), s_t(p_\tau) \right\rangle \right| \\
&\leq \sum_{i=1}^{d} \left| \left(\mathop{\mathbb{E}}_{y \sim D_\tau}[y_i] - p_{\tau,i} \right) \cdot s_{t,i}(p_\tau) \right| \\
&\leq \epsilon_0 \cdot \sum_{i=1}^{d} \left| \sum_{\sigma \in \{-1,1\}} \sum_{E \in \mathcal{E}} q_{t,(i,\sigma,E)} \cdot \sigma \cdot E(x_t, p_\tau) \right| \\
&\leq \epsilon_0
\end{aligned}$$

where the second last inequality is true with probability $1 - \delta_0$, from Lemma 86. We can quantify the probability that this inequality holds for all $\tau \in [T']$ simultaneously using a union bound:

$$\begin{aligned}
&\Pr\left[\bigcup_{\tau=1}^{T'} \mathop{\mathbb{E}}_{y \sim D_\tau}[u_t(p_\tau, y)] > \epsilon_0 \right] \leq T'\delta_0 \\
\implies &\Pr\left[\bigcap_{\tau=1}^{T'} \mathop{\mathbb{E}}_{y \sim D_\tau}[u_t(p_\tau, y)] \leq \epsilon_0 \right] \geq 1 - T'\delta_0 = 1 - \delta'
\end{aligned}$$

So with probability $1 - \delta'$, the sum of expected utilities can be bound, once again following Theorem 27,

$$\begin{aligned}
&\sum_{\tau=1}^{T'} \mathop{\mathbb{E}}_{y \sim D_\tau}\left[u'_t(p_\tau, y)\right] + \sum_{\tau=1}^{T'} \langle p_\tau, s_t(p_\tau) \rangle = \sum_{\tau=1}^{T'} \mathop{\mathbb{E}}_{y \sim D_\tau}[u_t(p_\tau, y)] \leq \epsilon_0 T' = \epsilon' T'/2 \\
\implies &\sum_{\tau=1}^{T'} \mathop{\mathbb{E}}_{y \sim D_\tau}\left[u'_t(p_\tau, y)\right] \leq \epsilon' T'/2 - \sum_{\tau=1}^{T'} \langle p_\tau, s_t(p_\tau) \rangle
\end{aligned}$$

Substituting this into equation 9,

$$\begin{aligned}
\max_{y \in \mathcal{C}} \sum_{\tau=1}^{T'} u_t(p_\tau, y) &\leq \left(R_{T',y^*} + \epsilon' T'/2 - \sum_{\tau=1}^{T'} \langle p_\tau, s_t(p_\tau) \rangle \right) + \sum_{\tau=1}^{T'} \langle p_\tau, s_t(p_\tau) \rangle \\
&= R_{T',y^*} + \epsilon' T'/2 \\
&\leq \sqrt{2dC^2T'} + \epsilon' T'/2
\end{aligned}$$

Since $\frac{1}{T'}\sum_{i=1}^{T'} u(p_\tau, y) = \mathbb{E}_{p\sim\overline{p}}[u_t(p,y)]$, it follows that

$$\max_{y\in\mathcal{C}} \cdot \mathop{\mathbb{E}}_{p\sim\overline{p}}[u_t(p,y)] \le \frac{\sqrt{2dC^2T'} + \epsilon' T'/2}{T'}$$

and so with probability $1-\delta'$, $\overline{p}$ is a $\left(\sqrt{\frac{2dC^2}{T'}} + \frac{\epsilon'}{2}\right)$-approximate equilibrium strategy for the learner. By our choice of $T' = \frac{8dC^2}{\epsilon'^2}$, this simplifies to ϵ', as desired. ■

Appendix J. Proofs from Section D

Theorem 30 *Fix a collection of L-Lipschitz utility functions $\mathcal{U}_L$, an action set $\mathcal{A} = \{1,\ldots,K\}$, and any transcript π_T. Let $\mathcal{E} = \{E_{u,a} : u \in \mathcal{U}, a\in\mathcal{A}\}$ be the set of binary events corresponding to straightforward decision makers with utility functions $u \in \mathcal{U}_L$ taking each action $a \in \mathcal{A}$. Then if π_T is α-unbiased with respect to $\mathcal{E}$, for every $u \in \mathcal{U}_L$, the straightforward decision maker with utility function u has type regret with respect to $\mathcal{U}_L$ at most:*

$$\max_{u'\in\mathcal{U}_L} r(\pi_T, u, u') \le \max_{u'\in\mathcal{U}_L} \frac{L\sum_{a\in\mathcal{A}}\left(\alpha(n_T(E_{u',a},\pi_T)) + \alpha(n_T(E_{u,a},\pi_T))\right)}{T}$$

For any concave α, this bound implies:

$$\max_{u'\in\mathcal{U}_L} r(\pi_T, u, u') \le \frac{2LK\alpha(T/K)}{T}$$

Proof [Proof of Theorem 30] Fix any pair $u, u' \in \mathcal{U}_L$. We need to upper bound $r(\pi_T, u, u')$. Using linearity of u in its second argument, we can derive:

$$\begin{aligned}
\frac{1}{T}\sum_{t=1}^{T} u(\delta_u(p_t), y_t) &= \sum_{a\in\mathcal{A}} \frac{1}{T}\sum_{t:\delta_u(p_t)=a} u(a, y_t) \\
&= \sum_{a\in\mathcal{A}} \frac{1}{T}\sum_{t=1}^{T} E_{u,a}(p_t) u(a, y_t) \\
&= \sum_{a\in\mathcal{A}} u\left(a, \frac{1}{T}\sum_{t=1}^{T} E_{u,a}(p_t) y_t\right) \\
&\ge \sum_{a\in\mathcal{A}} \left(u\left(a, \frac{1}{T}\sum_{t=1}^{T} E_{u,a}(p_t) p_t\right) - \frac{L\alpha(n_T(E_{u,a},\pi_T))}{T}\right) \\
&= \frac{1}{T}\sum_{t=1}^{T} u(\delta_u(p_t), p_t) - \frac{L\sum_{a\in\mathcal{A}}\alpha(n_T(E_{u,a},\pi_T))}{T}
\end{aligned}$$

where the inequality follows from the α-unbiasedness condition with respect to event $E_{u,a}$ and the L-Lipschitzness of u. Similarly, because of the α-unbiasedness condition with respect to event $E_{u',a}$ we have that:

$$\frac{1}{T}\sum_{t=1}^{T} u(\delta_{u'}(p_t), y_t) \le \frac{1}{T}\sum_{t=1}^{T} u(\delta_{u'}(p_t), p_t) + \frac{L\sum_{a\in\mathcal{A}}\alpha(n_T(E_{u',a},\pi_T))}{T}$$

We can therefore bound the regret $r(\pi_T, u, u')$ as:

$$\begin{aligned}
r(\pi_T, u, u') &= \frac{1}{T}\sum_{t=1}^{T} u(\delta_{u'}(p_t), y_t) - \frac{1}{T}\sum_{t=1}^{T} u(\delta_u(p_t), y_t) \\
&\leq \frac{1}{T}\sum_{t=1}^{T} (u(\delta_{u'}(p_t), p_t) - u(\delta_u(p_t), p_t)) + \frac{L\sum_{a\in\mathcal{A}}\left(\alpha(n_T(E_{u',a}, \pi_T)) + \alpha(n_T(E_{u,a}, \pi_T))\right)}{T} \\
&\leq \frac{L\sum_{a\in\mathcal{A}}\left(\alpha(n_T(E_{u',a}, \pi_T)) + \alpha(n_T(E_{u,a}, \pi_T))\right)}{T}
\end{aligned}$$

Here the last inequality follows from the fact that by definition of $\delta_u(p_t)$, $u(\delta_u(p_t), p_t) \geq u(a', p_t)$ for all $a' \in \mathcal{A}$ (including $a' = \delta_{u'}(p_t)$).

The simplified bound follows from the concavity of α and the fact that for every utility function u, the events $\{E_{u,a}\}_{a\in\mathcal{A}}$ are disjoint, which implies that $\sum_{a\in\mathcal{A}} n_T(E_{u,a}, \pi_T) \leq T$. ■

Appendix K. Proofs From Section F.1

Theorem 61 *Fix a collection of L-Lipschitz utility functions $\mathcal{U}_L$, an action set $\mathcal{A} = \{1, \ldots, K\}$, a collection of groups $\mathcal{G} \in 2^{\mathcal{X}}$, and any transcript π_T. Let $\mathcal{E} = \{E_{u,a,G} : u \in \mathcal{U}_L, a \in \mathcal{A}, G \in \mathcal{G}\}$ be the set of binary events defined as $E_{u,a,G}(x_t, p_t) = 1$ if and only if $x_t \in G$ and $\delta_u(p_t) = a$. Then if π_T is α-unbiased with respect to $\mathcal{E}$, for every $u \in \mathcal{U}_L$, the straightforward decision maker with utility function u has $\mathcal{G}$-groupwise swap regret at most:*

$$\max_{\phi:\mathcal{A}\to\mathcal{A}} r(\pi_T, u, \phi, G) \leq \frac{2L\sum_{a\in\mathcal{A}}\alpha(n_T(E_{u,a,G}, \pi_T))}{T_G(\pi_T)}$$

for every group $G \in \mathcal{G}$. If α is concave, then this bound is at most:

$$\max_{\phi:\mathcal{A}\to\mathcal{A}} r(\pi_T, u, \phi, G) \leq \frac{2LK\alpha(T_G(\pi_T)/K)}{T_G(\pi_T)}.$$

Proof [Proof of Theorem 61] Fix any $\phi : \mathcal{A} \to \mathcal{A}$, $u \in \mathcal{U}_L$, and $G \in \mathcal{G}$. We need to upper bound $r(\pi_T, u, \phi, G)$. Using the linearity of $u(a, \cdot)$ in its second argument for all $a \in \mathcal{A}$, we can write:

$$\begin{aligned}
r(\pi_T, u, \phi, G) &= \frac{1}{T_G(\pi_T)}\sum_{t:x_t\in G} u(\phi(\delta_u(p_t)), y_t) - u(\delta_u(p_t), y_t) \\
&= \frac{1}{T_G(\pi_T)}\sum_{a\in\mathcal{A}}\sum_{t:\delta_u(p_t)=a, x_t\in G} u(\phi(a), y_t) - u(a, y_t) \\
&= \sum_{a\in\mathcal{A}}\frac{1}{T_G(\pi_T)}\sum_{t=1}^{T} E_{u,a,G}(x_t, p_t)\left(u(\phi(a), y_t) - u(a, y_t)\right) \\
&= \sum_{a\in\mathcal{A}}\left(u\left(\phi(a), \frac{1}{T_G(\pi_T)}\sum_{t=1}^{T} E_{u,a,G}(x_t, p_t)y_t\right) - u\left(a, \frac{1}{T_G(\pi_T)}\sum_{t=1}^{T} E_{u,a,G}(x_t, p_t)y_t\right)\right) \\
&\leq \sum_{a\in\mathcal{A}}(u\left(\phi(a), \frac{1}{T_G(\pi_T)}\sum_{t=1}^{T} E_{u,a,G}(x_t, p_t)p_t\right) - u\left(a, \frac{1}{T_G(\pi_T)}\sum_{t=1}^{T} E_{u,a,G}(x_t, p_t)p_t\right)
\end{aligned}$$

$$
\begin{aligned}
&+ \quad \frac{2L\alpha(n_T(E_{u,a,G},\pi_T))}{T_G(\pi_T)}) \\
&\leq \quad \sum_{a\in\mathcal{A}} \left(\frac{2L\alpha(n_T(E_{u,a,G},\pi_T))}{T_G(\pi_T)} \right) \\
&= \quad \frac{2L\sum_{a\in\mathcal{A}}\alpha(n_T(E_{u,a},\pi_T))}{T_G(\pi_T)}
\end{aligned}
$$

Here the first inequality follows from the α-unbiasedness condition and the L-Lipschitzness of u: indeed, for every a' (and in particular for $a' \in \{a, \phi(a)\}$) we have

$$
\left| u\left(a', \frac{1}{T_G(\pi_T)}\sum_{t=1}^{T} E_{u,a,G}(x_t,p_t)p_t\right) - u\left(a', \frac{1}{T_G(\pi_T)}\sum_{t=1}^{T} E_{u,a,G}(x_t,p_t)y_t\right)\right|
$$

$$
\leq L|\frac{1}{T_G(\pi_T)}\sum_t (p_t - y_t)E_{u,a,G}(p_t)|_\infty = \frac{L}{T_G(\pi_T)}\max_{i\in[d]}|\sum_t (p_{t,i}-y_{t,i})E_{u,a,G}(p_t)| \leq \frac{\alpha(n_T(E_{u,a,G},\pi_T))L}{T_G(\pi_T)}.
$$

The 2nd inequality follows from the fact that by definition, whenever $\delta_u(p_t) = a$ (and hence whenever $E_{u,a}(p_t) = 1$), $u(a, p_t) \geq u(a', p_t)$ for all $a' \in \mathcal{A}$, and the fact that by Lemma 8, the levelsets of δ_u are convex, and hence $u(a, \frac{1}{T_G(\pi_T)}\sum_{t=1}^{T} E_{u,a,G}(x_t,p_t)p_t) \geq u(a', \frac{1}{T_G(\pi_T)}\sum_{t=1}^{T} E_{u,a,G}(x_t,p_t)p_t)$ for all a'.

For any utility function u and group G, the events $\{E_{u,a,G}\}_{a\in\mathcal{A}}$ are disjoint, and so for any u, G, and any π_T, $\sum_{a\in\mathcal{A}} n_T(E_{u,a,G},\pi_T) \leq T_G(\pi_T)$. Therefore, whenever α is a concave function (as it is for the algorithm we give in this paper, and as it is for essentially any reasonable bound), the term $\sum_{a\in\mathcal{A}}\alpha(n_T(E_{u,a,G},\pi_T))$ evaluates to at most $K\alpha(T_G(\pi_T)/K)$. ■

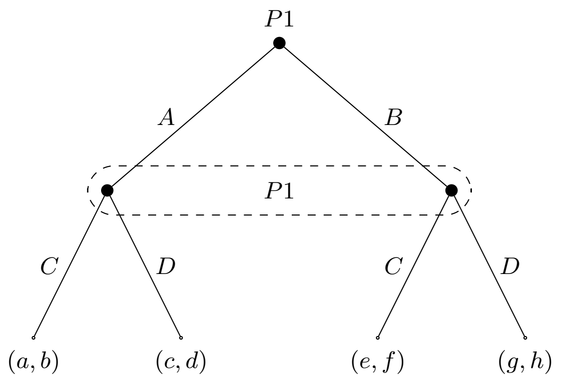
Appendix L. Additional Details from Section F.2

L.1. A Best-Response Oracle for Leaf-Form Strategies

In Section F.2, we reduced the problem of obtaining subsequence regret in an extensive-form game to the problem of optimizing over the set of all leaf-form strategies for that game, to find the learner's best response to any collection of opponent strategies. Here, we present a best-response oracle that efficiently returns a learner's best response for the subset of games for which the learner has *perfect recall* and *path recall*. While perfect recall is a well-studied and standard property in the literature, we introduce path recall as a new but related property specific to the requirements of our best-response oracle.

Definition 88 *A player i has* perfect recall *in an extensive-form game G if, for any information set $I \in \mathcal{I}_i$ and any two nodes $x, y \in I$, for any node $x' \in I' \in \mathcal{I}_i$ that is a predecessor of x, there is also a node $y' \in I'$ that precedes y, such that the action played at x' on the path to x and the action played at y' on the path to y are the same. If all players have perfect recall, then the game G itself is said to have perfect recall.*

Intuitively, a player has perfect recall if the partition of their nodes into information sets does not lose them any information about their past actions, i.e. the full history of a player's own actions is deducible from their current information set.



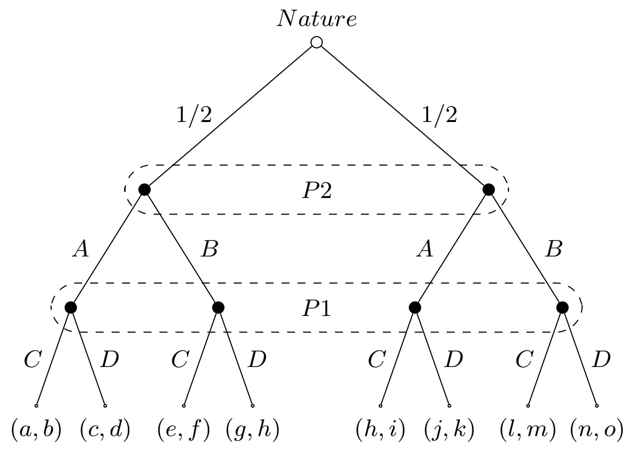


Figure 1: Examples of extensive-form games. In the game on the left, Player 1 has path recall, but not perfect recall. In the game on the right, Player 1 has perfect recall, but not path recall.

Definition 89 *A player i has* path recall *in an extensive-form game G if, for any information set $I \in \mathcal{I}_i$ and any two nodes $x, y \in I$,*

- *For any node $x' \in I' \in \mathcal{I}_i$ that is a predecessor of x, there is also a node $y' \in I'$ that precedes y.*
- *For any node h such that $\rho(h) \neq i$, if h is a predecessor of x, then h is a predecessor of y.*

Intuitively, a player has path recall if the partition of their nodes into information sets does not lose them any information about the past sequence of their own information sets and opponents' nodes along the path to any information set, i.e. the full history of the player's information sets and opponents' nodes reached so far is deducible from their current information set. Thus, perfect recall allows us to reason about the past in terms of one's own actions, while path recall allows us to reason about the past in terms of the sequence of learner's information sets and opponents' nodes passed so far. A game can satisfy perfect recall and not path recall, and vice versa (see Figure 1). In conjunction, however, these properties imply a useful structure to the game which we can exploit.

Lemma 90 *Define an extensive-form game G for which player i has perfect recall and path recall. For any node h such that $\rho(h) \neq i$, define C_h as the set of all children of h, i.e.*

$$C_h = \{h' \in \mathcal{H} \mid h' = c(h, a) \text{ for some } a \in \mathcal{A}(\Pi(h))\}.$$

For any of player i's information sets $I' \in \mathcal{I}_i$, either all nodes of I' are contained in C_h, or none of them are.

Proof Fix a node h played by someone other than player i and an information set $I' \in \mathcal{I}_i$. Assume by way of contradiction that there exist nodes h_1 and h_2 in I' such that $h_1 \in C_h$ and $h_2 \notin C_h$. By definition of path recall, h must be an ancestor of h_2. Since $h_2 \notin C_h$, there must be some node h_2' on the path between h and h_2. If $\rho(h_2') \neq i$, this contradicts the assumption of path recall (since

h'_2 is an ancestor of h_2 but not of h_1). If $\rho(h'_2) = i$, then in order for path recall to be satisfied, some ancestor h'_1 of h_1 must be in the same information set as h'_2. Since any ancestor of h_1 is also an ancestor of h'_2, h'_1 and h'_2 being in the same information set violates the assumption of perfect recall. Thus, for any node h such that $\rho(h) \neq i$, if any node in I' is in C_h, then all nodes of I' must be in C_h. ■

Lemma 91 *Define an extensive-form game G for which player i has perfect recall and path recall. For any information set I such that $\rho(I) = i$ and action $a \in \mathcal{A}(I)$, define $C_{I,a}$ as the set of all nodes resulting from playing action a at nodes in I, i.e.*

$$C_{I,a} = \{h' \in \mathcal{H} \mid h' = c(h, a) \text{ for some } h \in I\}.$$

For any information set $I' \in \mathcal{I}_i$, either all nodes of I' are contained in $C_{I,a}$, or none of them are.

Proof The proof parallels that of 90. Fix any two information sets $I, I' \in \mathcal{I}_i$, and any action $a \in \mathcal{A}(I)$. Assume by way of contradiction that there exist nodes h_1 and h_2 in I' such that $h_1 \in C_{I,a}$ and $h_2 \notin C_{I,a}$. By definition of perfect recall, h_2 must be a descendant of some node in I along the path taken after playing action a. That is, h_2 is the descendant of some node $h'_2 \in C_{I,a}$. if $\rho(h'_2) \neq i$, then path recall is not satisfied, since h'_2 is an ancestor of h_2 but not of h_1. If $\rho(h'_2) = i$, then in order for path recall to be satisfied, some ancestor h'_1 of h_1 must be in the same information set as h'_2. However, this means that h'_1, an ancestor of a node in I (call it h''_1), and h'_2, a descendant of a node in I, are in the same information set. In order for perfect recall to be satisfied, there must be some ancestor of h''_1 (a node in I) that is also in I, which itself violates the property of perfect recall. Thus, if any node in I' is in $C_{I,a}$, then all nodes of I' must be in $C_{I,a}$. ■

For any game that satisfies perfect recall and path recall for the learner, the children of any of the learner's information sets (or any opponent's nodes) is a union of learner's information sets and opponents' nodes. Thus, we can reason inductively about the leaf-form strategies restricted by any subtree of the game in terms of the leaf-form strategies restricted by the children of the root node of the subtree (or the information set rooting the subtree). This gives us a natural recursive algorithm able to return the optimal leaf-form strategy at any subtree rooted either at a learner's information set, or at an opponent's node, given a collection of opponents' strategies (in terms of a payoff-weighted reachability vector).

Firstly, note that Algorithm 7 returns a set of leaf nodes rather than a leaf-node strategy. We use this representation for ease of proof, with the observation that it is trivial to convert a set of leaf nodes to a leaf-node strategy, i.e. some $\overrightarrow{s} \in \{0,1\}^{|\mathcal{Z}|}$. For any subset of leaf nodes $s \in 2^{\mathcal{Z}}$, we use $\overrightarrow{s}$ to represent the corresponding vector representation. Secondly, Algorithm 7 takes a set of nodes I as input. In practice, this will either be a single node played by an opponent (or a terminal node), or an information set played by the learner (player i).

For notational purposes in following proofs, for any deterministic behavioral strategy $\pi \in \Pi_i$, let s_π be the corresponding leaf-form strategy. For any set of nodes $I \in 2^{\mathcal{H}}$, let $\pi|_I$ be the restriction of π to nodes in the subtree rooted at I, and let $s_\pi|_I$ be the restriction of s_π to the collection of leaf nodes reachable from I, that is, $z \in s_\pi|_I$ iff $z \in s_\pi$ and z is a descendant of some node in I.

Theorem 92 *Fix an extensive-form game G, a player i, a set of nodes I representing either one of player i's information sets or a single node of another player, and a payoff-weighted reachability*

```
// Base case - terminal nodes
if $I = \{h\} \cap h \in \mathcal{Z}$ then
  return ($\{h\}, \overrightarrow{v}(h)$)  // return payoff-weighted probability of
  reaching terminal node $h$ and the corresponding leaf-form
  strategy
else
  if $I \in \mathcal{I}_i$ then
    // Player $i$'s information set
    for $a \in \mathcal{A}(I)$ do
      Initialize $s_a = \emptyset, v_a = 0$.
      for $h' \in C_{I,a}$ such that $\rho(h') \neq i$ do
        Let $(s, v)$ = Best-Response-Leaf-Form($\{h'\}, \overrightarrow{v}, i, G$)
        $s_a = s_a \cup s, v_a = v_a + v$
      for $I' \in C_{I,a}$ such that $\rho(I') = i$ do
        Let $(s, v)$ = Best-Response-Leaf-Form($I', \overrightarrow{v}, i, G$)
        $s_a = s_a \cup s, v_a = v_a + v$
    Let $a_{\max} = \max_{a \in \mathcal{A}(I)} v_a$
    return $(s_{a_{\max}}, v_{a_{\max}})$
  else if $I = \{h\}$ and $\rho(h) \neq i$ then
    // Opponent node
    Initialize $s_I = \emptyset, v_I = 0$.
    for $h' \in C_h$ such that $\rho(h') \neq i$ do
      Let $(s, v)$ = Best-Response-Leaf-Form($\{h'\}, \overrightarrow{v}, i, G$)
      $s_I = s_I \cup s, v_I = v_I + v$
    for $I' \in C_h$ such that $\rho(I') = i$ do
      Let $(s, v)$ = Best-Response-Leaf-Form($I', \overrightarrow{v}, i, G$)
      $s_I = s_I \cup s, v_I = v_I + v$
    return $(s_I, v_I)$
```

Algorithm 7: Best-Response-Leaf-Form($I, \overrightarrow{v}, i, G$)

vector $v \in \mathbb{R}^{|\mathcal{Z}|}$. If player i has both perfect recall and path recall in G, then the backwards induction algorithm (Algorithm 7) returns an optimal leaf-form strategy s_I that maximizes the player's expected payoff over the set of all restrictions of leaf-node strategies to the subtree rooted at I, and the corresponding expected payoff v_I. That is,

$$s_I = \underset{s_\pi|_I, \pi \in \Pi_i}{\text{argmax}} \langle \overrightarrow{v}, \overrightarrow{s}_\pi|_I \rangle$$
$$v_I = \langle \overrightarrow{v}, \overrightarrow{s_I} \rangle.$$

Proof We proceed by induction. Our base case is when I consists of a single terminal node (some $h \in \mathcal{Z}$). In this case, no more actions are to be played and since there must exist some strategy $\pi \in \Pi_i$ such that h is reachable, there is exactly one leaf-form strategy restricted to the current sub-tree: $s = \{h\}$. So Algorithm 7 returns the only viable (and thus optimal) strategy, and the corresponding expected payoff for player i is $\overrightarrow{v}(h)$.

If I is not just a terminal node, there are two cases to consider; when I is an information set playable by player i, and when I consists of a single node playable by some opponent. When $I \notin \mathcal{I}_i$, $I = \{h\}$ such that player i does not control the action taken at h. Define $S_I = \{s_\pi|_I \mid \pi \in \Pi_i\}$ as the collection of all leaf-node strategies restricted to the current sub-tree (rooted at h). Consider C_h, the set of children of node h. By Lemma 90, C_h can be written as a disjoint union $\cup_{k=1}^m I_k$ such that each I_k is either an information set played by player i or a single node played by some opponent. For any strategy $\pi \in \Pi_i$, the restricted strategy $\pi|_I$ is described entirely in terms of strategies restricted by each of the sets I_k. That is, for any information set $I' \in \mathcal{I}_i$ within the current subtree, $\pi|_I$ is defined as follows:

$$\pi|_I(I') = \pi|_C(I')$$

where C is the unique I_k which roots the subtree containing I'. Further,

$$\langle \overrightarrow{v}, \overrightarrow{s}_\pi|_I \rangle = \sum_{k=1}^{m} \langle \overrightarrow{v}, \overrightarrow{s}_\pi|_{I_k} \rangle$$

since each terminal node in the subtree rooted at I will be in the subtree of exactly one of the subtrees rooted at each of the subsets I_k. Therefore, we can optimize for s_I by separately optimizing for each of the strategies s_{I_k}. By the inductive hypothesis, Algorithm 7 applied to each of these sets of nodes will return the optimal strategies, and the resulting leaf-node strategy (i.e. the set of reachable leaf nodes via this strategy) will be the union of all the resulting s_{I_k}. The corresponding expected payoff will accordingly be the sum of the childrens' payoffs.

Finally, we must consider the case when $I \in \mathcal{I}_i$. For any action $a \in \mathcal{A}(I)$, the set $C_{I,a}$ (using Lemma 91) of children of nodes in I reached upon taking action a can be written as a disjoint union $\cup_{k=1}^m I_k$ of information sets played by player i and single nodes played by opponents. Therefore, the maximum expected payoff *conditioned on taking action* a at I is the sum of the maximum expected payoffs with respect to each of the I_k making up $C_{I,a}$ - and by the inductive hypothesis, the payoff returned by Algorithm 7 for eack I_k is the payoff corresponding to the optimal leaf-form strategy restricted to I_k. Having computed the maximum expected payoff received for taking each possible action at I, selecting the optimal distribution over actions at I is straightforward - we deterministically choose the single action which maximizes this value ex post.

As Algorithm 7 implements the logic described above in each case, the proof is complete. ■

Running Algorithm 7 on the root node of any game G satisfying the necessary properties will return a best-response leaf-form strategy for the learner along with the corresponding expected payoff.

L.2. Improving Algorithm Efficiency

The algorithm for obtaining subsequence regret has computational efficiency depending largely on two factors - the length of the vector predicted at each round (upon which the algorithm making unbiased predictions has a linear dependence) and the efficiency of the best-response oracle. The best-response oracle described in the previous section (Algorithm 7) uses the payoff-weighted reachability vector v_t with length $|\mathcal{Z}|$, the number of terminal nodes in the game being played. This representation was chosen in order to be able to describe the expected payoff of a learner's strategy as a linear function of the strategy. For extensive-form games where there is a more compact representation that satisfies this linearity property and is still sufficiently informative in order for the algorithm to compute a best-response, the general principle described in Section F.2 still works -

now with the algorithms introduced in Section C running in time linear with respect to this smaller vector dimension. Below, we describe a class of extensive-form games for which the length of the vector prediction scales with the number of the learner's information sets, as opposed to the number of terminal nodes, and for which the best-response oracle described in the previous section (with very minor alterations) still outputs best-responses.

At a high level, Algorithm 7 reasons about the maximum possible payoff at any node of a tree in terms of the payoffs of its children, and, starting from the terminal nodes, backtracks through the tree to choose, at each of the learner's information sets, the action which maximizes expected payoff. Thus, any linear representation of the opponents' (predicted) strategy which allows us to compute the expected payoff for each action at each of the learner's information sets will suffice.

Definition 93 *An information set $I \in \mathcal{I}_i$ of player i is a* terminal information set *if for each node $h \in I$, none of the descendants of h are playable by player i.*

Definition 94 *The* predecessor function *$P_i : \mathcal{Z} \to \mathcal{I}_i \cup \emptyset$ for player i maps each terminal node z to the last information set of player i that is played along the path from the root to z. If no such information set exists, then $P_i(z) = \emptyset$.*

For any game G where $P_i(z)$ is a terminal information set for each $z \in \mathcal{Z}$[10] (assuming the learner is player i), rather than predict the probability of reaching each terminal node, we can predict v as the expected payoff for taking each viable action at each of the learner's terminal information sets. Correspondingly, instead of picking a leaf-form strategy, the learner can pick a "terminal information set" strategy π indexed by pairs of terminal information sets I and actions $a \in \mathcal{A}(I)$, such that $\pi_{I,a} = 1$ iff action a is taken at information set I. Then, the net payoff using strategy π is $\langle \pi, v \rangle$, where $v_{I,a}$ is the expected payoff from taking action a at information set I, and we once again have a linear optimization problem, now in dimension scaling linearly with the number of information sets and available actions at each of these information sets. Since the payoff for each non-terminal information set is deducible from the payoffs at each terminal information set (since the path to any leaf node must cross some terminal information set), the same Algorithm 7, modified only to consider terminal information sets as base cases instead of terminal nodes, will be able to compute a best-response.

L.3. Connecting Subsequence Regret to Other Forms of Regret in Extensive-Form Games

To define subsequence regret in a similar framework as causal regret, or more generally, Φ-regret, we must define deviations in terms of behavioral strategies, rather than in terms of leaf-form strategies. In the rest of this section, we work only with behavioral strategies, and we assume efficient conversion of leaf-form strategies to behavioral strategies via an oracle.

Definition 95 *Fix a collection of binary events $\mathcal{E}$ and behavioral strategy space S_i. Fix some $E \in \mathcal{E}$. A subsequence deviation is a function $\phi_E : S_i \to S_i$ such that*

$$\phi_E(s_i) = \begin{cases} s_i', & E = 1 \\ s_i, & \textit{otherwise} \end{cases}$$

Let Φ_{sub} be the set of all possible subsequence deviations.

10. The games shown in Figure 1 are examples of games satisfying this property. Figure 2 is an example of game that does not satisfy this property.

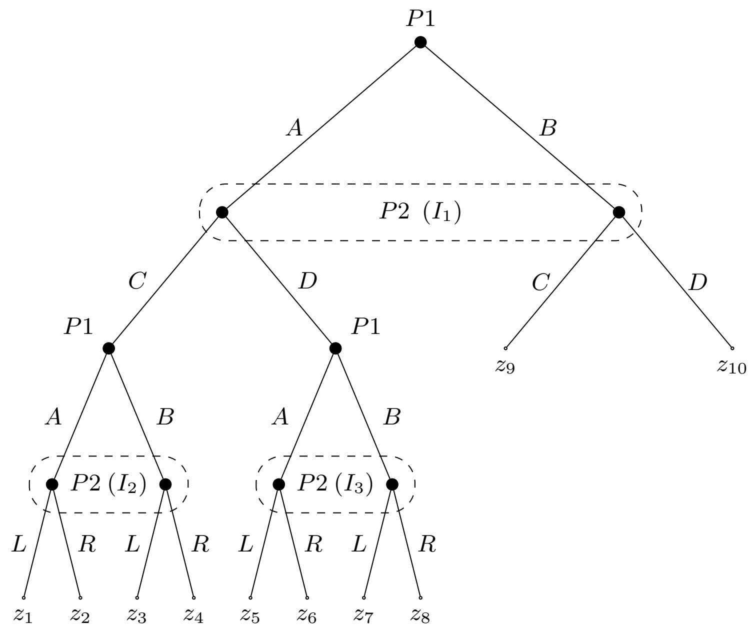


Figure 2: A two-player extensive-form game. Of Player 2's three information sets, only I_2 and I_3 are terminal. Note that $P_2(z_9) = P_2(z_{10}) = I_1$, which is not a terminal information set.

Translating the events defined by Theorem 74 to a set of subsequence deviations, we obtain:

Corollary 96 *Fix an information set I', subsequent action $a' \in \mathcal{A}(I)$, strategy s_i', events $\mathcal{E} = \{E_{I,a} : I \in \mathcal{I}_i, a \in \mathcal{A}(I)\}$, and transcript π_T. Let a* subsequence deviation *$\phi_{I',a'}$ of strategy $s_i \in S_i$ return a strategy such that at each information set $I \in \mathcal{I}$,*

$$\phi_{I',a'}(s_i)(I) = \begin{cases} s_i'(I), & E_{I',a'} = 1 \\ s_i(I), & \textit{otherwise} \end{cases}$$

Let Φ_{sub} be the collection of all $\phi_{I',a'}$. If the learner has $(\mathcal{E}, \alpha)$-subsequence regret, then the learner has Φ-regret with respect to the class of subsequence deviations bounded by:

$$\max_{\phi_{I',a'} \in \Phi_{sub}} r(\pi_T, u, \phi_{I',a'}) \leq \alpha$$

The proof of this follows straight from definition – for each event $E_{I',a'}$, the learner deviates to some fixed strategy s_i' in the transcript whenever $E_{I',a'} = 1$. To show that no subsequence regret over events $\mathcal{E}$ implies no causal regret, we will define a related, but slightly different notion of a causal deviation that we will use to directly compare to a subsequence deviation known as a *reachable informed causal deviation.*

Definition 97 *Fix an information set I', subsequent action $a' \in \mathcal{A}(I)$, and strategy s_i'. A* reachable informed causal deviation *ϕ of strategy $s_i \in S_i$ returns a strategy such that at each information set*

$I \in \mathcal{I}$,

$$\phi(s_i) = \begin{cases} s_i'(I), & I \succeq I', s_i(I') = a', I' \textit{ reachable under } s_i \\ s_i(I), & \textit{otherwise} \end{cases}$$

Let $\Phi_{\textit{r-causal}} = \{\phi_{I,a,s} : I \in \mathcal{I}_i, a \in \mathcal{A}(I), s \in S_i\}$ be the set of all reachable informed causal deviations.

Lemma 98 *Fix an extensive-form game G and any transcript π_T. Let the action space $\mathcal{A} = S_i$ be the set of all deterministic behavioral strategies of player i. Then, for all $\phi_{I,a,s} \in \Phi_{\textit{r-causal}}$ and $\phi'_{I,a,s} \in \Phi_{\textit{causal}}$,*

$$r(\pi_T, u, \phi_{I,a,s}) = r(\pi_T, u, \phi'_{I,a,s})$$

Proof Suppose the transcript of behavioral strategies in π_T is $(s_1, s_2, \dots s_T)$. Observe that for all strategies s_k where $k \in [T]$ such that the trigger information set I is reachable under s_k, $\phi_{I,a,s}(s_k) = \phi'_{I,a,s}(s_k)$, by definition.

Thus, the modified strategies differ only between strategies s_j such that I is not reachable under s_j. When I is not reachable under s_j, $\phi_{I,a,s}(s_j)(I') = s_j(I')$ and $\phi'_{I,a,s}(s_j)(I') = s_j(I')$ for all $I' \in \mathcal{I}_i$ except for information sets $I' \succeq I$. However, since I is not reachable under s_j, play will never reach I, and thus the regret must also be equal between the two modified strategies. ■

Lemma 99 *The set of reachable informed causal deviations $\Phi_{\textit{r-causal}}$ is a subset of the subsequence deviations $\Phi_{\textit{sub}}$ defined by events $\mathcal{E} = \{E_{I,a} : I \in \mathcal{I}_i, a \in \mathcal{A}(I)\}$ in Theorem 74.*

Proof Fix any $\phi_{I',a',s'} \in \Phi_{\text{r-causal}}$ with trigger sequence (I', a') and trigger deviation s'. By definition,

$$\phi_{I',a',s'}(s) = \begin{cases} s'(I), & I \succeq I', s(I') = a', I' \text{ is reachable under } s \\ s(I), & \text{otherwise} \end{cases}$$

Clearly, when I' is reachable under s and $s(I') = a$, and thus $E_{I,a} = 1$. Thus, $\phi_{I',a',s'}(s)$ can be expressed as a subsequence deviation:

$$\phi_{I',a',s'}(s) = \phi^{sub}_{I',a',s^!}(s) = \begin{cases} s^!(I), & E_{I,a} = 1 \\ s(I), & \text{otherwise} \end{cases}$$

where $s^!(I)$ is the behavioral strategy equal to $s'(I)$ at all information sets $I \succeq I'$, and equal to $s(I)$ otherwise. ■

Theorem 76 *Fix an extensive-form game G and transcript π_T. If the algorithm has $\mathcal{E}$-subsequence regret with event set $\mathcal{E} = \{E_{I,a} : I \in \mathcal{I}_i, a \in \mathcal{A}(I)\}$ bounded by:*

$$\max_{E \in \mathcal{E}, \phi \in \Phi_{\textit{Ext}}} r(\pi_T, u, \phi, E) \leq \alpha,$$

then we have that causal regret, or Φ-regret with respect to the set $\Phi_{\textit{causal}}$, is bounded by:

$$\max_{\phi \in \Phi_{\textit{causal}}} r(\pi_T, u, \phi) \leq \alpha.$$

Proof We can apply Corollary 96, showing that if we have $\mathcal{E}$-subsequence regret at most α, we must have Φ-regret with respect to Φ_{sub} bounded by:

$$\max_{\phi \in \Phi_{\text{sub}}} r(\pi_T, u, \phi) \leq \alpha$$

Next, by Lemma 98, we can safely consider only the set of reachable causal deviations $\Phi_{\text{r-causal}}$ as relevant to causal regret. By Lemma 99, we have that $\Phi_{\text{r-causal}} \subseteq \Phi_{\text{sub}}$, and thus, we must have

$$\max_{\phi \in \Phi_{\text{causal}}} r(\pi_T, u, \phi) = \max_{\phi \in \Phi_{\text{r-causal}}} r(\pi_T, u, \phi) \leq \max_{\phi^{sub} \in \Phi_{\text{sub}}} r(\pi_T, u, \phi) \leq \alpha$$

■

Remark 100 *The set of linear-swap deviations [26] is not a superset of the deviations that can be captured by subsequence regret over a fixed collection of events $\mathcal{E}$. In their paper, the authors give an example of a nonlinear swap (see their Example E.3), where strategy A_1B_2 is swapped to A_1B_1, and strategy A_2B_1 is swapped to A_1B_1. A simplified game tree is given below this remark. This can be represented with subsequence deviations with event set $\mathcal{E} = \{E_{1,2}, E_{2,1}\}$, where $E_{i,j} = 1$ if A_i is played at information set I_1 and B_j is played at information set I_2, deviating to the fixed strategy A_1B_1 whenever an event occurs.*

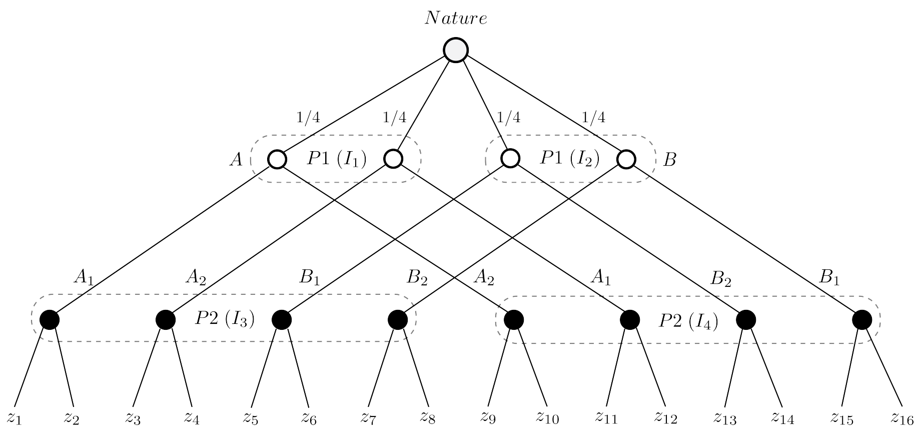


Figure 3: A simplified game tree given in [26], Example E.3.

Remark 101 *The deviations of the behavioral deviation landscape [74] are not a superset of the deviations that can be captured by subsequence regret over a fixed collection of events $\mathcal{E}$. For example, instantiating an event set $\mathcal{E} = \{E_{I_1,I_2,I_3,a_3} : I_1, I_2, I_3 \in \mathcal{I}_i, a_3 \in \mathcal{A}(I_3)\}$ such that $E_{I_1,I_2,I_3,a_3} = 1$ if a strategy makes I_1, I_2 reachable under the learner's strategy but also plays the action a_3 at information set I_3 does not correspond to any deviation type enumerated within the behavioral deviation landscape.*

Appendix M. Proofs From Section F.2

Theorem 63 *Fix a base action set $B = \{1, 2, \dots n\}$, prediction space $\mathcal{C} = [-1, 1]^n$, an action set $\mathcal{D} \subseteq 2^B$, a collection of events $\mathcal{E}$, and utility function $u(S_t, p_t) = \sum_{i \in S_t} p_{t,i}$. Let $\mathcal{I} = \{I_{b,E} : b \in B, E \in \mathcal{E}\}$ be the set of binary events defined as $I_{b,E}(p_t) = 1$ if and only if $b \in \delta_u(p_t)$ and $E(\pi_{t-1}, x_t, p_t) = 1$. Then if π_T is α-unbiased with respect to $\mathcal{I} \cup \mathcal{E}$, the straightforward decision maker with utility function u has $\mathcal{E}$-subsequence regret at most:*

$$\max_{E \in \mathcal{E}, \phi \in \Phi_{Ext}} r(\pi_T, u, \phi, E) \leq \sum_{b \in B} \frac{\alpha(n_T(E, \pi_T)) + \alpha(n_T(I_{b,E}, \pi_T))}{T}$$

Proof [Proof of Theorem 63] Fix any $\phi \in \Phi_{\text{Ext}}$ and $E \in \mathcal{E}$. We need to upper bound $r(\pi_T, u, \phi, E)$. Using the linearity of $u(S_t, p_t) = \sum_{i \in S_t} p_{t,i}$ in its second argument for all $S_t \in \mathcal{A}$, and observing that $\phi(\cdot)$ is a constant strategy modification function (i.e. $\phi(D) = D'$ for all $D \in \mathcal{D}$), we can write:

$$\begin{aligned}
r(\pi_T, u, \phi, E) &= \frac{1}{T}\sum_{t=1}^{T} E(x_t, p_t, \pi_{t-1}) \cdot (u(\phi(\delta_u(p_t)), y_t) - u(\delta_u(p_t), y_t)) \\
&= \frac{1}{T}\sum_{t=1}^{T} E(x_t, p_t, \pi_{t-1}) \cdot \big(u(D', y_t) - u(\delta_u(p_t), y_t)\big) \\
&= \frac{1}{T}\sum_{t=1}^{T} E(x_t, p_t, \pi_{t-1}) \cdot \sum_{b \in D'} y_{t,b} - \frac{1}{T}\sum_{t=1}^{T} E(x_t, p_t, \pi_{t-1}) \sum_{b \in \delta_u(p_t)} y_{t,b} \\
&= \frac{1}{T}\left(\sum_{b \in D'}\sum_{t=1}^{T} E(x_t, p_t, \pi_{t-1}) y_{t,b} - \sum_{b \in B}\sum_{t=1}^{T} I_{b,E}(p_t) \cdot y_{t,b}\right) \\
&\leq \frac{1}{T}\left(\sum_{b \in B}\sum_{t=1}^{T} E(x_t, p_t, \pi_{t-1}) \cdot p_{t,b} - I_{b,E}(p_t) \cdot p_{t,b}\right) \\
&\quad + \sum_{b \in B} \frac{\alpha(n_T(E, \pi_T)) + \alpha(n_T(I_{b,E}, \pi_T))}{T} \\
&\leq \sum_{b \in B} \frac{\alpha(n_T(E, \pi_T)) + \alpha(n_T(I_{b,E}, \pi_T))}{T}
\end{aligned}$$

Here the first inequality follows the α-unbiasedness condition over $\mathcal{I}$: for every $b \in B$, we have

$$\left|\sum_{t=1}^{T} (p_{t,b} - y_{t,b}) I_{b,E}(p_t)\right| \leq \alpha(n_T(I_{b,E}, \pi_T)).$$

Similarly, α-unbiasedness over $\mathcal{E}$ gives us, for every $b \in B$:

$$\left|\sum_{t=1}^{T} (p_{t,b} - y_{t,b}) E(x_t, p_t, \pi_{t-1})\right| \leq \alpha(n_T(E, \pi_T)).$$

The 2nd inequality follows from the fact that by definition of the straightforward decision maker, $u(\delta_u(p_t), p_t) \geq u(D', p_t)$ for any $D' \in \mathcal{D}$, and the second term in the previous expression can be simplified:

$$\sum_{b \in B} \sum_{t=1}^{T} I_{b,E}(p_t) \cdot p_{t,b} = \sum_{t=1}^{T} \sum_{b \in \delta_u(p_t)} E(x_t, p_t, \pi_{t-1}) \cdot p_{t,b} = \sum_{t=1}^{T} E(x_t, p_t, \pi_{t-1}) \cdot u(\delta_u(p_t), p_t).$$

■

Corollary 102 *Fix any collection of events $\mathcal{E}$. The canonical Predict-Then-Act algorithm parameterized with $\mathcal{C} = [-1, 1]^n, \mathcal{A} = \mathcal{D}, \mathcal{E}' = \mathcal{I} \cup \mathcal{E}$, where $\mathcal{I} = \{I_{b,E} : b \in B, E \in \mathcal{E}\}$, obtains expected subsequence regret at each round $t \leq T$ at most:*

$$\mathbb{E}_{\pi_t}\left[\max_{E \in \mathcal{E}, \phi \in \Phi_{Ext}} r(\pi_t, u, \phi, E)\right] \leq O\left(\frac{n\sqrt{\ln(n|\mathcal{E}|T)}}{\sqrt{t}}\right).$$

We can implement the Predict-Then-Act algorithm using Algorithm 3, with runtime in every round $t \in [T]$ polynomial in n and t.

Proof From Theorem 14, at round t the predictions of the canonical algorithm with event set $\mathcal{E}'$ have expected bias bounded by

$$\begin{aligned}\mathbb{E}_{\pi^t}\left[\alpha(T, n_t(E', \pi_t))\right] &= O\left(\ln(n|\mathcal{E}'|T) + \sqrt{\ln(n|\mathcal{E}'|T) \cdot n_t(E', \pi_t)}\right) \\ &= O\left(\ln(n|\mathcal{E}|T) + \sqrt{\ln(n|\mathcal{E}|T) \cdot n_t(E', \pi_t)}\right)\end{aligned}$$

for each event $E' \in \mathcal{E}$. Since $n_t(E', \pi_t) \leq t$, we can plug this into Theorem 63 to get that the canonical algorithm has $\mathcal{E}$-subsequence regret bounded by

$$\max_{E \in \mathcal{E}, \phi \in \Phi_{\text{Ext}}} r(\pi_T, u, \phi, E) \leq O\left(\frac{n\sqrt{\ln(n|\mathcal{E}|T)}}{\sqrt{t}}\right).$$

■