Online Learning from Human Feedback with Applications to Exoskeleton Gait Optimization

Thesis by
Ellen Novoseller

In Partial Fulfillment of the Requirements for the
Degree of
Doctor of Philosophy

CALIFORNIA INSTITUTE OF TECHNOLOGY
Pasadena, California

2021
Defended November 30th, 2020

Ellen Novoseller
ORCID: 0000-0001-5263-0598

ACKNOWLEDGEMENTS

I am deeply grateful to my advisors, Professors Joel Burdick and Yisong Yue, for their support along my PhD journey. This thesis would not have been possible without their constant advice, insights, patience, guidance, and encouragement.

I would also like to thank my committee members, Professors Aaron Ames, Dorsa Sadigh, and Richard Murray, for taking time out of their busy schedules to provide valuable suggestions and advice.

Also, I am extremely grateful to everyone with whom I have collaborated (in no particular order): Maegan Tucker, Kejun Li, Myra Cheng, Claudia Kann, Richard Cheng, Yibing Wei, Erdem Bıyık, Jeffrey Edlund, Charles Guan, Atli Kosson, Solveig Einarsdottir, Sonia Moreno, and Professors Aaron Ames, Yanan Sui, Dorsa Sadigh, and Dimitry Sayenko. I have learned a tremendous amount from you, and this work would never have been possible if I had not had the opportunity to work together with you.

I would like to thank all of my colleagues in Joel's and Yisong's groups; I have been really fortunate to get to know you during my time at Caltech. I am also lucky to have many great friends who have been there for me over the last six years and made grad school more enjoyable.

Finally, I am grateful to my family for their love and for believing in me, particularly my husband David, my parents, and my brother Michael.

ABSTRACT

Systems that intelligently interact with humans could improve people's lives in numerous ways and in numerous settings, such as households, hospitals, and workplaces. Yet, developing algorithms that reliably and efficiently personalize their interactions with people in real-world environments remains challenging. In particular, one major difficulty lies in adapting to human-in-the-loop feedback, in which an algorithm makes sequential decisions while receiving online feedback from humans; throughout this interaction, the algorithm seeks to optimize its decision-making quality, as measured by the utility of its performance to the human users. Such algorithms must balance between exploration and exploitation: on one hand, the algorithm must select uncertain strategies to fully explore the environment and the interacting human's preferences, while on the other hand, it must exploit the empirically-best-performing strategies to maximize its cumulative performance.

Learning from human feedback can be difficult, as people are often unreliable in specifying numerical scores. In contrast, humans can often more accurately provide various types of qualitative feedback, for instance pairwise preferences. Yet, sample efficiency is a significant concern in human-in-the-loop settings, as qualitative feedback is less informative than absolute metrics, and algorithms can typically pose only limited queries to human users. Thus, there is a need to create theoretically-grounded online learning algorithms that efficiently, reliably, and robustly optimize their interactions with humans while learning from online qualitative feedback.

This dissertation makes several contributions to algorithm design for human-in-the-loop learning. Firstly, this work develops the Dueling Posterior Sampling (DPS) algorithmic framework, a model-based, Bayesian approach for online learning in the settings of preference-based reinforcement learning and generalized linear dueling bandits. DPS is developed together with a theoretical regret analysis framework, and yields competitive empirical performance in a range of simulations. Additionally, this thesis presents the CoSpar and LineCoSpar algorithms for sample-efficient, mixed-initiative learning from pairwise preferences and coactive feedback. CoSpar and LineCoSpar are both deployed in human subject experiments with a lower-body exoskeleton to identify optimal, user-preferred exoskeleton walking gaits. This work presents the first demonstration of preference-based learning for optimizing dynamic crutchless exoskeleton walking for user comfort, and makes progress toward customizing exoskeletons and other assistive devices for individual users.

PUBLISHED CONTENT AND CONTRIBUTIONS

Novoseller, Ellen R. et al. "Dueling posterior sampling for preference-based reinforcement learning." In: *Conference on Uncertainty in Artificial Intelligence (UAI)*. PMLR. 2020, pp. 1029–1038. URL: http://proceedings.mlr.press/v124/novoseller20a.html.

E.R.N. contributed to the conception of the project, developing the algorithm, performing the theoretical analysis, conducting the simulation experiments, and writing the manuscript.

Tucker, Maegan, Myra Cheng, et al. "Human preference-based learning for high-dimensional optimization of exoskeleton walking gaits." In: *IEEE International Conference on Intelligent Robots and Systems (IROS)*. 2020. URL: https://arxiv.org/pdf/2003.06495.pdf.

E.R.N. contributed to the conception of the project, developing the algorithm, providing ongoing mentorship and direction for conducting the simulations, conducting the exoskeleton experiments, and writing the manuscript.

Tucker, Maegan, Ellen R. Novoseller, et al. "Preference-based learning for exoskeleton gait optimization." In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2020. DOI: 10.1109/ICRA40945.2020.9196661. URL: https://ieeexplore.ieee.org/document/9196661.

E.R.N. contributed to the conception of the project, developing the algorithm, conducting the simulation and exoskeleton experiments, analyzing the experimental results, and writing the manuscript.

CONTENTS

Acknowledgements . . . iii
Abstract . . . iv
Published Content and Contributions . . . v
Contents . . . v
List of Figures . . . viii
List of Tables . . . xvii
Chapter I: Introduction . . . 1
1.1 Motivation . . . 2
1.2 The Bandit and Reinforcement Learning Problems . . . 3
1.3 Human-in-the-Loop Learning . . . 4
1.4 Lower-Body Exoskeletons for Mobility Assistance . . . 6
1.5 Contributions . . . 8
1.6 Organization . . . 10
Chapter II: Background . . . 11
2.1 Bayesian Inference for Parameter Estimation . . . 11
2.2 Gaussian Processes . . . 16
2.3 Entropy, Mutual Information, and Kullback-Leibler Divergence . . . 22
2.4 Bandit Learning . . . 24
2.5 Dueling Bandits . . . 36
2.6 Episodic Reinforcement Learning . . . 41
Chapter III: The Preference-Based Generalized Linear Bandit and Reinforcement Learning Problem Settings . . . 48
3.1 The Generalized Linear Dueling Bandit Problem Setting . . . 48
3.2 The Preference-Based Reinforcement Learning Problem Setting . . . 51
3.3 Comparing the Preference-Based Generalized Linear Bandit and RL Settings . . . 54
Chapter IV: Dueling Posterior Sampling for Preference-Based Bandits and Reinforcement Learning . . . 55
4.1 The Dueling Posterior Sampling Algorithm . . . 55
4.2 Additional Notation . . . 58
4.3 Posterior Modeling for Utility Inference and Credit Assignment . . . 58
4.4 Theoretical Analysis . . . 64
4.5 Empirical Performance of DPS . . . 95
4.6 Discussion . . . 102
Chapter V: Mixed-Initiative Learning for Exoskeleton Gait Optimization . . . 103
5.1 Introduction . . . 103
5.2 Background on the Atalante Exoskeleton and Gait Generation for Bipedal Robots . . . 106
5.3 The CoSpar Algorithm for Preference-Based Learning . . . 109

5.4 Simulation Results for CoSpar . . . 114
5.5 Deployment of CoSpar in Human Subject Exoskeleton Experiments . 117
5.6 The LineCoSpar Algorithm for High-Dimensional Preference-Based Learning . . . 119
5.7 Performance of LineCoSpar in Simulation . . . 125
5.8 Deployment of LineCoSpar in Human Subject Exoskeleton Experiments . . . 127
5.9 Discussion . . . 129
Chapter VI: Conclusions and Future Directions . . . 131
6.1 Conclusion . . . 131
6.2 Future Work . . . 132
Bibliography . . . 135
Appendix A: Models for Utility Inference and Credit Assignment . . . 149
A.1 Bayesian Linear Regression . . . 149
A.2 Gaussian Process Regression . . . 149
A.3 Gaussian Process Preference Model . . . 155
Appendix B: Proofs of Asymptotic Consistency for Dueling Posterior Sampling 158
B.1 Facts about Convergence in Distribution . . . 160
B.2 Asymptotic Consistency of the Transition Dynamics in DPS in the Preference-Based RL Setting . . . 160
B.3 Asymptotic Consistency of the Utilities in DPS . . . 171
B.4 Asymptotic Consistency of the Selected Policies in DPS . . . 190
Appendix C: Additional Details about the Dueling Posterior Sampling Experiments in the Linear and Logistic Dueling Bandit Settings . . . 192
C.1 Baselines: Sparring with Upper Confidence Bound (UCB) Algorithms 192
C.2 Hyperparameter Optimization . . . 196
Appendix D: Additional Details about the Dueling Posterior Sampling Experiments in the Preference-Based RL Setting . . . 199

LIST OF FIGURES

Number *Page*

1.1 The Atalante exoskeleton, designed by Wandercraft (Wandercraft, n.d.). 8

4.1 Comparison of Poisson disk sampling and uniform random sampling over the surface of the 3-dimensional unit sphere. Both plots show 100 samples. While the uniformly random samples often cluster together, the Poisson disk samples are more uniformly spaced over the sphere's surface. 84

4.2 Cumulative regret and estimated information ratio values in the linear bandit setting with relative Gaussian feedback over pairs of actions. Values are plotted over the entire learning process for three representative experimental repetitions (colors are identical for corresponding experiment runs between the two plots). These three experiments use $d = 3$, $\lambda = 1$, $\sigma = 1$, and $A = 100$. 85

4.3 Scatter plots of the maximum estimated information ratio value in each of the 960 relative Gaussian bandit simulations (each dot corresponds to one of these simulations). All 960 trials are shown in each of the three plots. Values are color-coded according to: a) the action space size A, b) σ, and c) λ. 86

4.4 Estimating the information ratio with different numbers of MCMC warm-up steps. In each of the five DPS runs with 100 learning iterations each, the information ratio is independently estimated six times per learning iteration, once with each of the following numbers of warm-up samples: $\{0, 5, 10, 50, 100, 500\}$. On the plot, each color corresponds to one of the five simulation runs (i.e., for each run, the six different numbers of warm-up samples are plotted in the same color). Clearly, as the similarly-colored lines mostly overlap, the number of warm-up samples does not heavily impact the information ratio estimates. 87

4.5 Corner plots comparing posterior samples from the Laplace approximation and MCMC. The prior over utility vectors $\overline{\boldsymbol{r}} = [r_1, r_2, r_3]^T$ is only supported on $||\overline{\boldsymbol{r}}||_2 \leq 1$. a)-b) Case 1: $\overline{\boldsymbol{r}} = [0.7, 0.5, 0.2]^T$. The true posterior is close to Gaussian, so the two approximations appear similar. c)-d) Case 2: $\overline{\boldsymbol{r}} = [0.8, 0.5, 0.1]^T$, which is closer to the boundary of allowable utility vectors, $||\overline{\boldsymbol{r}}||_2 = 1$. Because the true posterior is highly non-Gaussian, the two approximations are visibly different (compare r_1 versus r_2 in the plots). Importantly, the MCMC posterior respects the constraint $||\overline{\boldsymbol{r}}||_2 \leq 1$, while the Laplace approximation does not. 89

4.6 Corner plots comparing posterior samples from the Laplace approximation and MCMC under small numbers of samples. The Laplace and MCMC posteriors are visibly different, as the utility posterior is highly non-Gaussian. The preference data was generated by running DPS with $d = 3, c = 4, \lambda = 1$, and $A = 10$. 90

4.7 Cumulative regret and estimated information ratio values in the preference-based linear bandit setting with posterior modeling via MCMC. Values are plotted over the learning process for three representative experimental repetitions (colors are identical for corresponding experimental runs between the two plots). The experiments in this plot use $d = 10$, $\lambda = 1$, and $A = 10$. 91

4.8 Scatter plot of the maximum information ratio estimates when running DPS in the linear dueling bandit setting with MCMC posterior inference. Each dot corresponds to one of 120 simulation runs, and values are color-coded according to the action space size, A. 91

4.9 Cumulative regret and information ratio estimates in the RL setting with relative feedback, known dynamics, and posterior inference via relative Gaussian feedback. Values are plotted over the learning process for three representative experimental repetitions (colors are identical for each experimental run between the two plots). These experiments utilize $\lambda = 1$, $\sigma = 10$, and MDP structure 1 (see Table 4.1), for which $S = 3$, $A = 2$, and $h = 4$. 93

4.10 Scatter plots of the maximum information ratio estimates in each of the 600 relative Gaussian RL simulations (each dot corresponds to one of the 600 simulations). The x-axis labels indicate the MDP labels 0-4 (from Table 4.1) and the values of $d = SA$. All 600 trials are shown in both of the two plots. Values are color-coded according to: a) σ and b) λ. In all cases, the information ratios are significantly less than d. 93

4.11 Cumulative regret and estimated information ratio values in the preference-based RL setting with known dynamics and posterior inference via MCMC. Values are plotted over the learning process for three representative experimental repetitions (colors are identical for corresponding simulation runs between the two plots). These experiments use $\lambda = 1$ and MDP structure 4 (see Table 4.1), for which $S = 3, A = 3, h = 2$. The value of c is set to $2\sqrt{2}h = 4\sqrt{2}$. 94

4.12 Scatter plot of the maximum information ratio estimates for the preference-based RL setting with known dynamics and MCMC posterior inference. Each dot corresponds to one of the 80 simulation runs, with the maximum taken over the 1,000 learning iterations. The x-axis labels indicate the MDP structure labels (from Table 4.1) and the values of $d = SA$. In all cases, the information ratios are significantly less than d. 95

4.13 Empirical performance of DPS in the generalized linear dueling bandit setting (mean ± std over 100 runs). The plots show DPS with utility inference via both Bayesian logistic and linear regression, and under both logistic (a-i) and linear (j-l) user feedback noise. The plots normalize the rewards for each simulation run such that the best and worst actions in $\mathcal{A}$ have rewards of 1 and 0, respectively. For both baselines, the hyperparameters were optimized independently in each of the 12 cases, while for linear and logistic DPS, the plots depict results for a single set of well-performing hyperparameters. DPS is competitive against both baselines and is robust to the utility inference method and to noise in the preference feedback. 98

4.14 Empirical performance of DPS; each simulated environment is shown under the two least-noisy user preference models evaluated. The plots show DPS with three credit assignment models: Gaussian process regression (GPR), Bayesian linear regression, and a Gaussian process preference model. PSRL is an upper bound that receives numerical rewards, while EPMC is a baseline. Plots display the mean +/- one standard deviation over 100 runs of each algorithm tested. Results from the remaining user noise parameters are plotted in Appendix D. For RiverSwim and Random MDPs, normalization is with respect to the total reward achieved by the optimal policy. Overall, DPS performs well and is robust to the choice of credit assignment model. 101

5.1 Atalante Exoskeleton with and without a user. The user is wearing a mask to measure metabolic expenditure. 104

5.2 Human subject experiments with the LineCoSpar algorithm exploring six exoskeleton gait parameters: step length, step duration, step width, maximum step height, pelvis roll, and pelvis pitch. 106

5.3 Leftmost: Cost of transport (COT) for the compass-gait biped at different step lengths and a fixed 0.2 m/s velocity. Remaining plots: posterior utility estimates of CoSpar ($n = 2$, $b = 0$; without coactive feedback) after varying iterations of learning (posterior mean +/- 2 standard deviations). The plots each show three posterior samples, which lie in the high-confidence region (mean +/- 2 standard deviations) with high probability. The posterior utility estimate quickly converges to identifying the optimal action. 115

5.4 CoSpar models two-dimensional utility functions using preference data. a) Example of a synthetic 2D objective function. b) Utility model posterior learned after 150 iterations of CoSpar in simulation ($n = 1$; $b = 1$; coactive feedback). CoSpar prioritizes identifying and exploring the optimal region, rather than learning a globally-accurate utility landscape. 116

5.5 CoSpar simulation results on 100 2D synthetic objective functions, comparing CoSpar with and without coactive feedback for three settings of the parameters n and b (see Algorithm 13). Mean +/- standard error of the objective values achieved over the 100 repetitions. The maximal and minimal objective function values are normalized to 0 and 1. We see that coactive feedback always helps, and that $n = 2$, $b = 0$—which receives the fewest preferences—performs worst. . . . 117

5.6 Experimental results for optimizing step length with three subjects (one row per subject). Columns 1-4 illustrate the evolution of the preference model posterior (mean +/- standard deviation), shown at various trials. CoSpar converges to similar but distinct optimal gaits for different subjects. Column 5 depicts the subjects' blind ranking of the three gaits executed after 20 trials. The rightmost column displays the experimental trials in chronological order, with the background depicting the posterior preference mean at each step length. CoSpar draws more samples in the region of higher posterior preference. . . . 119

5.7 Experimental results from two-dimensional feature spaces (top row: step length and duration; bottom row: step length and width). Columns 1-4 illustrate the evolution of the preference model's posterior mean. Column 4 also shows the subject's blind rankings of the three gaits executed after 20 trials. Column 5 depicts the experimental trials in chronological order, with the background as in Figure 5.6. CoSpar draws more samples in the region of higher posterior preference. . . . 120

5.8 Experimental phase diagrams of the left leg joints over 10 seconds of walking. The gaits shown correspond to the maximum, mean, and minimum preference posterior values for both of subject 1's 2D experiments. For instance, Subject 1 preferred gaits with longer step lengths, as shown by the larger range in sagittal hip angles in the phase diagram. 120

5.9 Curse of dimensionality for CoSpar. Average time per iteration of CoSpar versus LineCoSpar. The y-axis is on a logarithmic scale. For LineCoSpar, the time is roughly constant in the number of dimensions d, while the runtime of CoSpar increases exponentially. For $d = 4$, the duration of a CoSpar iteration is inconvenient in the human-in-the-loop learning setting, and for $d \geq 5$, it is intractable. 121

5.10 Convergence to higher objective values on standard benchmarks. Mean objective value ± standard deviation using H3 and H6, averaged over 100 runs. Compared to CoSpar, LineCoSpar converges to sampling actions with higher objective values at a faster rate, as it employs an improved sampling approach and link function. It is intractable to run CoSpar on a 6-dimensional action space. 126

5.11 Robustness to noisy preferences. Mean objective value ± standard deviation of the action $\boldsymbol{x}_{\max}$ with the highest posterior utility. This is averaged over 100 runs of LineCoSpar on H6 with varying preference noise, as quantified by c_h. Higher performance correlates with less noise (lower c_h). The algorithm is robust to noise up to a certain degree ($c_h \leq 0.5$). 126

5.12 Coactive feedback improves convergence of LineCoSpar. Mean objective value ± standard deviation of the sampled actions using random objective functions. Results are averaged over 1,000 runs of LineCoSpar over 100 randomly-generated six-dimensional functions ($d = 6$; 10 runs per synthetic function). The sampled actions converge to high objective values in relatively few iterations, and coactive feedback accelerates this process. 127

5.13 LineCoSpar experimental procedure. After setup of the subject-exoskeleton system, subjects were queried for preferences between all consecutive gait pairs, along with coactive feedback, in 30 gait trials (in total, at most 29 pairwise preferences and 30 pieces of coactive feedback). After these 30 trials, the subject unknowingly entered the validation portion of the experiment, in which he/she validated the posterior-maximizing gait, $\boldsymbol{x}_{\max}$, against four randomly-selected gaits. 128

5.14 Exploration versus exploitation in the LineCoSpar human trials. Each row depicts the distribution of a particular gait parameter's values across all gaits that the subject tested. Each dimension is discretized into 10 bins. Note that the algorithm explores different parts of the action space for each subject. These visitation frequencies exhibit a statistically-significant correlation with the posterior utilities across these regions (Pearson's p-value = 1.22e-10). 129

C.1 Hyperparameter sensitivity of DPS in the linear and logistic dueling bandit settings (mean ± std over 20 runs). The plots show DPS with utility inference via both Bayesian linear (a-c) and logistic (d-f) regression. The learning curves compare several sets of hyperparameters among those evaluated, for several different action space dimensionalities d and levels of logistic user feedback noise c. The plots normalize the rewards for each simulation run such that the best action in $\mathcal{A}$ has a reward of 1, while the worst action has a reward of 0. Overall, DPS performs well and is robust to the hyperparameter values to a certain degree. 198

D.1 Empirical performance of DPS in the RiverSwim environment. Plots display mean +/- one standard deviation over 100 runs of each algorithm tested. Normalization is with respect to the total reward achieved by the optimal policy. Overall, DPS performs well and is robust to the choice of credit assignment model. 200

D.2 Empirical performance of DPS in the Random MDP environment. Plots display mean +/- one standard deviation over 100 runs of each algorithm tested. Normalization is with respect to the total reward achieved by the optimal policy. Overall, DPS performs well and is robust to the choice of credit assignment model. 203

D.3 Empirical performance of DPS in the Mountain Car environment. Plots display mean +/- one standard deviation over 100 runs of each algorithm tested. Overall, DPS performs well and is robust to the choice of credit assignment model. 203

D.4 Empirical performance of DPS in the RiverSwim environment for different hyperparameter combinations. Plots display mean +/- one standard deviation over 30 runs of each algorithm tested with logistic user noise and $c = 0.001$. Overall, DPS is robust to the choice of hyperparameters. The hyperparameter values depicted in each plot are (from left to right): for Bayesian linear regression, $(\sigma, \lambda) = \{(0.5, 0.1), (0.5, 10), (0.1, 0.1), (0.1, 10), (1, 0.1)\}$; for GP regression, $(\sigma_f^2, \sigma_n^2) = \{(0.1, 0.001), (0.1, 0.1), (0.01, 0.001), (0.001, 0.0001), (0.5, 0.1)\}$; for Bayesian logistic regression (special case of the GP preference model), $(\lambda, \alpha) = \{(1, 1), (30, 1), (20, 0.5), (1, 0.5), (30, 0.1)\}$; and additionally for the GP preference model, $c \in \{0.5, 1, 2, 5, 13\}$. See Table D.2 for the values of any hyperparameters not specifically mentioned here. 205

D.5 Empirical performance of DPS in the Random MDP environment for different hyperparameter combinations. Plots display mean +/- one standard deviation over 30 runs of each algorithm tested with logistic user noise and $c = 0.001$. Overall, DPS is robust to the choice of hyperparameters. The hyperparameter values depicted in each plot are (from left to right): for Bayesian linear regression, $(\sigma, \lambda) = \{(0.1, 10), (0.1, 0.1), (0.05, 0.01), (0.5, 20), (1, 10)\}$; for GP regression, $(\sigma_f^2, \sigma_n^2) = \{(0.05, 0.0005), (0.001, 0.0001), (0.05, 0.1), (0.001, 0.0005), (1, 0.1)\}$; for Bayesian logistic regression (special case of the GP preference model), $(\lambda, \alpha) = \{(0.1, 0.01), (1, 0.01), (0.1, 1), (30, 0.1), (5, 0.5)\}$; and additionally for the GP preference model, $c \in \{1, 10, 15, 19, 100\}$. See Table D.3 for the values of any hyperparameters not specifically mentioned here. 206

D.6 Empirical performance of DPS in the Mountain Car environment for different hyperparameter combinations. Plots display mean +/- one standard deviation over 30 runs of each algorithm tested with logistic user noise and $c = 0.001$. Overall, DPS is robust to the choice of hyperparameters. The hyperparameter values depicted in each plot are (from left to right): for Bayesian linear regression, $(\sigma, \lambda) = \{(10, 1), (10, 10), (30, 0.001), (0.001, 10), (0.1, 0.1)\}$; for GP regression, $(\sigma_f^2, l, \sigma_n^2) = \{(0.01, 2, 10^{-5}), (0.01, 1, 10^{-5}), (0.1, 2, 0.01), (1, 2, 0.001), (0.001, 3, 10^{-6})\}$; for Bayesian logistic regression (special case of the GP preference model), $(\lambda, \alpha) = \{(0.0001, 0.01), (0.1, 0.01), (0.0001, 0.0001), (0.001, 0.0001), (0.001, 0.01)\}$; and additionally for the GP preference model, $c \in \{10, 300, 400, 700, 1000\}$. See Table D.4 for the values of any hyperparameters not specifically mentioned here. 207

LIST OF TABLES

Number *Page*

4.1 The labels 0-4 are assigned to the five MDP structures used in the RL information ratio simulations. For each of the five MDP structures, this table specifies the number of states S, the number of actions A, and the episode horizon h. For each (S, A, h) triple, the total numbers of deterministic policies and of distinct policy pairs (on which information ratio computations depend) are also displayed. . . 92

5.1 Gait parameters optimizing LineCoSpar's posterior mean ($\boldsymbol{x}_{\max}$) for each able-bodied subject. 130

C.1 Hyperparameters in the linear and logistic dueling bandit experiments. For the linear and logistic UCB algorithms, the hyperparameters were optimized individually for each dimension d and preference noise level. For each d, the best-performing hyperparameters are listed in the following order: logistic noise with $c = 0.01$, logistic noise with $c = 0.1$, logistic noise with $c = 1$, and linear noise with $c = 4$. For each of the two versions of DPS, a single set of hyperparameters was found that performed well across the different values of d and noise levels considered. Hyperparameters were optimized by performing 20 simulation runs of each candidate set of values. 197

D.1 Hyperparameters for the EPMC baseline algorithm (Wirth and Fürnkranz, 2013a). Each table element shows the best-performing α/η values for the corresponding simulation environment and type of simulated user feedback (logistic or linear noise). For preference feedback with logistic noise, values of c are given in parentheses; larger values correspond to noisier preference feedback. 201

D.2 Credit assignment hyperparameters tested for the RiverSwim Environment. 202

D.3 Credit assignment hyperparameters tested for the Random MDP Environment. 202

D.4 Credit assignment hyperparameters tested for the Mountain Car Environment. 204

Chapter 1

INTRODUCTION

We are quickly entering an era where digital and cyberphysical systems are increasingly autonomous and can adapt to the specific needs of individuals. In particular, there is an increasing focus on personalized medicine (Sui, Yue, and Burdick, 2017; Sui, Zhuang, et al., 2018), autonomous driving (Basu, Bıyık, et al., 2019; Basu, Yang, et al., 2017), and personalized education (Jain, Thiagarajan, et al., 2020). Furthermore, intelligent robotic assistive devices that can adapt to each user's limitations could give increased independence to individuals with disabilities or limited mobility (Harib et al., 2018; Donati et al., 2016).

Yet, creating principled algorithms that efficiently and reliably personalize their interactions with people in real-world settings remains challenging. In particular, one difficulty lies in adapting to human feedback via human-in-the-loop learning, in which the algorithm seeks to optimize its interactions with people while receiving sequential feedback from users.

The problem of sequential interaction with an unknown environment has been studied extensively in the contexts of bandit learning and reinforcement learning (RL). In these settings, an agent takes actions in the environment and typically receives a reward signal reflecting the quality of those actions. The agent's learning goal is to maximize its cumulative reward over time, or equivalently, to minimize its *regret*, that is, gap between the algorithm's total reward and the total reward obtained by executing the optimal action sequence.

To achieve low regret, algorithms must balance between exploration and exploitation: locating optimal actions requires exploring actions with uncertain rewards, while maximizing cumulative reward also requires exploiting the empirically-best-performing actions.

In practice, people are typically unreliable in specifying numerical scores, but can give more accurate qualitative information, for instance correctly answering preference queries of the form, "Do you prefer trial A or B?" (Yue, Broder, et al., 2012), or suggesting improvements to the presented trials, for instance via the coactive feedback learning paradigm (Shivaswamy and Joachims, 2015).

This dissertation tackles the problem of online learning from sequential, qualitative human feedback in order to minimize regret with respect to user satisfaction. Learning from qualitative feedback is challenging, as such data contains less information than numerical rewards; for instance, a pairwise preference query yields only a single bit of information. Furthermore, human-in-the-loop learning algorithms can collect only limited data in practice, as each trial involves interacting with a human. Human patience, energy, resources, and attention have limits. For instance, the total time available in a clinical laboratory that can be directed to the process of adapting an assistive medical device to a patient may be quite limited due to cost. For these reasons, sample efficiency is a significant concern when learning from online human feedback.

This thesis presents the Dueling Posterior Sampling (DPS) algorithmic framework for online learning in the preference-based RL and generalized linear bandit settings, together with a concurrently-developed theoretical analysis framework for bounding the regret. This model-based approach learns a Bayesian posterior over the *utility function* underlying the user's feedback, which quantifies the user's satisfaction with the agent's task performance. Furthermore, this work presents the CoSpar framework for mixed-initiative learning in the bandit setting. The CoSpar algorithm is deployed on a robotic exoskeleton platform to optimize exoskeleton walking gaits in response to online human feedback, and achieves state-of-the-art performance in personalized exoskeleton gait optimization.

1.1 Motivation

Robots that interact intelligently with humans could improve people's lives in many capacities, from performing everyday household tasks to improving surgical outcomes for patients to assisting individuals with mobility impairments. Such devices must interact with human users intuitively and adapt to people's preferences quickly, reliably, and safely.

Many online learning algorithms operate within the multi-armed bandits or reinforcement learning (RL) frameworks, which formalize the problem of learning to maximize reward feedback over time. Yet, many bandit and RL algorithms rely upon receiving a numerical reward signal, and humans are often unreliable in specifying such absolute reward feedback. In practice, human-specified scores may drift over time (Payne et al., 1993; Brochu, Brochu, and de Freitas, 2010), while misspecified rewards can exhibit unintended side effects—for example, a robot might knock over a

vase to clean an area more quickly—and reward hacking, in which the agent "games" the objective function, finding a solution that achieves a high reward but perverts the human's intent (Amodei et al., 2016). For instance, in OpenAI's widely-cited boat racing example (Clark and Amodei, 2016), the agent fails to finish the race course as intended; rather, it maximizes its reward by turning in a large circle repeatedly, each time hitting three point-yielding targets just as they regenerate, crashing into other objects, and catching on fire. In another example (Popov et al., 2017), an agent fails to complete a simulated robotic block stacking task because it is rewarded for raising the vertical coordinate of the block's ground-facing side, and learns to do so by flipping the block upside down (instead of picking it up).

While humans can have difficulty with assigning accurate numerical scores, however, people *can* often give qualitative feedback more reliably. Examples include preference-based feedback, which poses queries of the form, "Do you prefer trial A or B?" (Yue, Broder, et al., 2012; Sui, Zhuang, et al., 2017; Sui, Zoghi, et al., 2018); coactive feedback, in which the user suggests an improvement following each trial (Shivaswamy and Joachims, 2015; Shivaswamy and Joachims, 2012; Raman et al., 2013); and ordinal feedback, in which users assign each trial to one of several discrete, ordered categories (e.g., "bad," "neutral," and "good") (Chu and Ghahramani, 2005a).

Thus, there is an increasing need to develop systematic algorithmic frameworks for integrating and learning from different modalities of user feedback, that are both theoretically-grounded and practical in real-world settings.

1.2 The Bandit and Reinforcement Learning Problems

This thesis considers human-in-the-loop learning problems that build upon the standard bandit and reinforcement learning (RL) problem settings. Bandits and RL are both sequential decision-making problems in which a learning agent takes actions in an environment while seeking to maximize a numerical reward signal.

Firstly, in the standard multi-armed bandit setting (Robbins, 1952; Agrawal and Goyal, 2012) an agent sequentially selects actions, and after each action it receives a numerical reward reflecting the quality of that action. The agent's goal is to minimize its regret, that is, the gap between the algorithm's total reward and the reward obtained by repeatedly selecting the optimal action. Achieving low regret requires both choosing the empirically-most-promising actions (exploitation) and selecting actions with uncertain rewards, in case the optimal action has not yet been

discovered (exploration).

In the RL setting, meanwhile, the agent similarly takes actions and observes rewards, but in addition to generating rewards, these actions alter the environment's underlying state. After each step, the agent not only receives the reward signal, but also observes the environment's state. If the environment only has one state, then RL reduces to the bandit problem. With more than one environment state, however, the decision-making process must account for the state transition dynamics as well as the rewards.

A number of strategies have been proposed for tackling the exploration-exploitation trade-off in both the bandit and RL settings, including upper confidence bound algorithms that predict reward information optimistically (Auer, Cesa-Bianchi, and Fischer, 2002; Dann and Brunskill, 2015; Abbasi-Yadkori, Pál, and Szepesvári, 2011); posterior sampling, which learns a Bayesian model over the environment and samples from it to select actions (Agrawal and Goyal, 2012; Agrawal and Jia, 2017; Abeille and Lazaric, 2017; Osband, Russo, and Van Roy, 2013); and information-theoretic approaches, which trade-off between choosing actions with high estimated rewards and actions expected to yield significant new information (Russo and Van Roy, 2014a; Kirschner and Krause, 2018; Nikolov et al., 2018).

1.3 Human-in-the-Loop Learning

This work considers several different paradigms of online learning from humans. Unlike in the standard bandit and RL settings, the feedback signal is no longer numerical, but rather qualitative. For example, in the dueling bandit setting (Yue, Broder, et al., 2012; Yue and Joachims, 2011; Sui, Zhuang, et al., 2017; Sui, Zoghi, et al., 2018), the agent chooses at least two actions in each learning iteration and receives feedback in the form of pairwise preferences between the selected actions. The notion of regret can be adapted to the dueling bandits problem: similarly to the numerical-reward setting, the agent must balance between exploration and exploitation to minimize the gap between its performance and that achieved by repeatedly selecting the most-preferred actions.

Meanwhile, in the episodic preference-based RL setting, the agent executes trajectories of interaction with the environment, and receives pairwise preference feedback revealing which of the trajectories are preferred. The RL problem is more challenging than the bandit problem, since instead of selecting individual actions, the agent selects (potentially-stochastic) *policies* that govern action selection as a function of

the environment's state. Thus, RL policies can be viewed analogously to actions in the bandit setting. The environment's dynamics then stochastically translate the agent's policies to the observed trajectories. While in the standard RL problem, the agent receives rewards after every action, in preference-based RL, the agent only receives preferences between entire trajectories of interaction.

Learning from trajectory-level preferences is in general a very challenging problem, as information about the rewards is sparse (often just one bit), is only relative to the pair of trajectories being compared, and does not explicitly include information about actions within the trajectories. One approach is to infer the utilities of individual state-action pairs by solving the *temporal credit assignment problem* (Akrour, Schoenauer, and Sebag, 2012; Zoghi, Whiteson, Munos, et al., 2014; Szörényi et al., 2015; Christiano et al., 2017; Wirth, Fürnkranz, and Neumann, 2016; Wirth, Akrour, et al., 2017), i.e., determining which of the encountered states and actions are responsible for the trajectory-level preference feedback.

Preference-based learning algorithms have seen success in a number of domains. In the bandit setting, preference-based algorithms have been deployed in several real-world applications, including optimizing spinal cord injury therapy in clinical trials (Sui, Zhuang, et al., 2018; Sui, Yue, and Burdick, 2017; Sui and Burdick, 2014), learning search result rankings in the information retrieval setting (Yue, Finley, et al., 2007; Radlinski and Joachims, 2005), and optimizing parameters in computer graphics design (Brochu, de Freitas, and Ghosh, 2008; Brochu, Brochu, and de Freitas, 2010).

Preference-based RL algorithms, meanwhile, have demonstrated successful performance applications including Atari games and the Mujoco environment (Christiano et al., 2017; Ibarz et al., 2018), learning human preferences for autonomous driving (Sadigh et al., 2017; Bıyık, Huynh, et al., 2020), and selecting a robot's controller parameters (Kupcsik, Hsu, and Lee, 2018; Akrour, Schoenauer, Sebag, and Souplet, 2014). Yet, there remains a lack of formal frameworks for theoretical analysis of preference-based RL algorithms.

Much of the existing work in preference-based RL focuses on a different setting from that considered in this thesis. While this work is concerned with online regret minimization, several existing algorithms instead minimize preference queries to the user (Christiano et al., 2017; Wirth, Fürnkranz, and Neumann, 2016). Such algorithms typically assume that many simulations can be executed inexpensively between preference queries. In many domains, however, experimentation is as time-intensive as

preference elicitation; for instance, in adaptive experiment design and human-robot interaction, it could be infeasible to accurately simulate the environment.

As mentioned previously, sample complexity is an important concern when learning from humans. Data collection is expensive, as the algorithm can only pose a limited number of queries to the user. For instance, Brochu, de Freitas, and Ghosh (2008) assert that "requiring more than 50 user queries in a real application would be unacceptable." One approach toward acquiring more data under a fixed trial budget is to learn simultaneously from more than one type of user feedback, for instance via mixed-initiative systems.

In particular, this work considers combining preferences with coactive feedback, in which the user gives suggestions in response to each action that the algorithm selects. Coactive feedback has been applied to web search and movie recommendation tasks in simulation (Shivaswamy and Joachims, 2015), as well as to optimize trajectory planning in robotic manipulation tasks such as grocery store checkout (Jain, Sharma, et al., 2015).

1.4 Lower-Body Exoskeletons for Mobility Assistance

In the United States alone, various forms of paralysis affect nearly 5.4 million people (Armour et al., 2016), with major causes including stroke, spinal cord injury, multiple sclerosis, and cerebral palsy. While this group includes people with a range of movement difficulties, in particular (in the United States), there are currently approximately 300,000 individuals with severe spinal cord injuries (*Facts and Figures at a Glance* 2019), while every year, more than 795,000 people have a stroke (*Stroke Facts* 2020). Such individuals could benefit significantly from assistive devices that help to replace or restore lost motor function.

Lower-body exoskeletons are wearable assistive devices that can restore mobility to people suffering from lower-body mobility impairments such as paralysis.[1] Exoskeleton-assisted walking can benefit patients in several ways. Firstly, while wheelchairs are limited by obstacles such as stairs, exoskeletons are comparatively less encumbered. Furthermore, continuous wheelchair use can result in pressure sores and loss of bone mass. For instance, Goemaere et al. (1994) found that paraplegic patients who regularly perform passive weightbearing standing with the aid of a standing device have better-preserved bone mass in comparison to control sub-

[1] While exoskeletons can also be designed to augment function for healthy humans (Dollar and Herr, 2008), this dissertation is concerned with exoskeletons for assisting mobility-impaired individuals.

jects. In fact, a survey of wheelchair users and healthcare professionals working with mobility-impaired individuals (Wolff et al., 2014) identified "health benefits" as the most highly-recommended reason for using an exoskeleton. Respondents further indicated a number of health benefits associated with exoskeleton use, including pressure relief, increased circulation, improved bone density, improved bowel and bladder function, and reduced risk of orthostatic hypotension.

Finally, multiple studies have demonstrated that exoskeletons have significant potential to assist in patient rehabilitation following spinal cord injuries and strokes (Gad et al., 2017; Donati et al., 2016; Mehrholz et al., 2017). In Donati et al. (2016), eight patients with chronic paraplegia spent 12 months training with an exoskeleton, and regained voluntary motor control below the level of their spinal cord injuries; four of these patients were upgraded from a complete to an incomplete paraplegia classification. Mehrholz et al. (2017) found that the combination of exoskeleton-assisted gait training with physiotherapy improved stroke patients' recovery of independent walking compared to physiotherapy alone.

To improve users' quality of life, exoskeleton devices must be comfortable, affordable, safe, and enable users to engage in everyday activities. For lower-body exoskeletons, designing comfortable walking gaits is a significant challenge, due to the high cost of human trials, the enormous space of possible walking gaits, and the need to ensure safety during learning. Furthermore, exoskeleton walking must be personalized to each individual user.

This thesis applies human-in-the-loop learning to optimize gait parameters for the Atalante lower-body exoskeleton, designed by the French company Wandercraft, to identify user-personalized gaits that maximize comfort. The exoskeleton is pictured in Figure 1.1.

The Atalante exoskeleton, first introduced in Harib et al. (2018), has 18 degrees of freedom and 12 actuated joints: three at each hip, one at each knee, and two in each ankle. Gurriet, Finet, et al. (2018) describe the device's mechanical components and control architecture in detail. Importantly, while other exoskeletons require users to rely upon crutches for balance and stability (Dollar and Herr, 2008), Atalante facilitates dynamically-stable, crutchless walking by leveraging the partial hybrid zero dynamics framework to formally generate stable gaits (Gurriet, Finet, et al., 2018).

Figure 1.1: The Atalante exoskeleton, designed by Wandercraft (Wandercraft, n.d.).

1.5 Contributions

This dissertation presents three main contributions to algorithm design for human-in-the-loop learning. Firstly, it develops the Dueling Posterior Sampling (DPS) framework for preference-based learning in the RL and generalized linear bandit settings. Secondly, this thesis presents the CoSpar and LineCoSpar learning frameworks for sample-efficient, mixed-initiative learning in the Gaussian process bandits regime. Thirdly, the CoSpar and LineCoSpar algorithms are deployed on the Atalante lower-body exoskeleton to learn personalized, user-preferred walking gaits. Each of these contributions is further described below.

The DPS algorithm uses preference-based posterior sampling to tackle the regret minimization problem in the Bayesian regime. Posterior sampling (Thompson, 1933) is a Bayesian model-based approach to balancing exploration and exploitation, which enables the algorithm to efficiently learn models of both the environment's state transition dynamics and reward function. Previous work on posterior sampling in RL (Osband, Russo, and Van Roy, 2013; Gopalan and Mannor, 2015; Agrawal and Jia, 2017; Osband and Van Roy, 2017) is focused on learning from absolute rewards, while DPS extends posterior sampling to both elicit and learn from trajectory-level preference feedback.

To elicit preference feedback, at every episode of learning, DPS draws two independent samples from the posterior to generate two trajectories to duel against each other. This approach is inspired by the SelfSparring algorithm proposed for the bandit setting (Sui, Zhuang, et al., 2017), but has a quite different theoretical analysis,

due to the need to incorporate structured preference feedback over RL trajectories and featurized actions.

DPS learns from preference feedback by internally maintaining a Bayesian model over the environment. In the RL setting, this approach models both the transition dynamics and rewards over state-action pairs. The reward model explains the preferences by solving the *temporal credit assignment problem* to determine which of the encountered states and actions are responsible for the trajectory-level preferences. This thesis presents several Bayesian approaches to credit assignment, including via Bayesian linear regression and two Gaussian process-based methods.

In the bandit setting, the DPS framework provides a low-dimensional structure for relating actions and user preferences in terms of the actions' features (for instance, the features could represent RL policy parameters). Thus, DPS can learn over large or infinite action spaces, which would be infeasible for a learning algorithm that models the actions' rewards independently of each other.

Although the study of preference-based RL has seen increased interest in recent years (Christiano et al., 2017; Wirth, Akrour, et al., 2017; Ibarz et al., 2018), it remains an open challenge to design formal frameworks that admit tractable theoretical analysis. This thesis develops DPS concurrently with an analysis framework for characterizing regret convergence in the episodic RL setting, based upon information-theoretic techniques for bounding the Bayesian regret of posterior sampling (Russo and Van Roy, 2016). The analysis depends upon upper-bounding a quantity called the *information ratio*, which balances between the expected instantaneous regret and the information gained about the optimal action. This work conjectures an upper-bound for the information ratio under a linear credit assignment model, which is supported by an extensive set of simulations that estimate the information ratio empirically. This result leads to a Bayesian no-regret rate for DPS under credit assignment via Bayesian linear regression.

In summary, concurrently with developing the DPS algorithm, this work mathematically integrates Bayesian credit assignment and preference elicitation within the conventional posterior sampling framework, evaluates several credit assignment models, and presents a regret analysis framework. Finally, experiments demonstrate that DPS delivers competitive performance empirically.

This thesis also presents the CoSpar and LineCoSpar frameworks for efficient, mixed-initiative learning, and applies them to optimize exoskeleton gaits for user

comfort. This work was conducted jointly with Dr. Aaron Ames' research group, and in particular with Maegan Tucker and Myra Cheng. Firstly, the CoSpar algorithm relies on Gaussian process modeling and posterior sampling to infer each action's utility to the user and to select actions that achieve low regret under sequential feedback. To increase sample efficiency, CoSpar integrates preference and coactive feedback within a mixed-initiative system. CoSpar was experimentally deployed in human subject trials with the Atalante exoskeleton to determine users' preferred walking parameters within a gait library. CoSpar successfully identified users' preferred exoskeleton step lengths, durations, and widths, while also providing insights into the users' preferences for certain gaits.

This thesis also describes the LineCoSpar algorithm, which incorporates techniques from high-dimensional Gaussian process Bayesian optimization (Kirschner, Mutny, et al., 2019) to jointly optimize over larger numbers of gait parameters. LineCoSpar delivers robust performance in simulation, and in human subject experiments, the algorithm jointly optimizes user preferences over six exoskeleton gait parameters. To our knowledge, LineCoSpar is the first algorithm for high-dimensional Gaussian process learning under preference feedback. Furthermore, this line of work demonstrates the first application of preference-based learning for optimizing dynamic crutchless walking. The results present progress for customizing exoskeleton walking for user comfort, as well as for gaining an understanding of the mechanisms underlying comfortable walking in an exoskeleton.

1.6 Organization

This dissertation is organized as follows. Chapter 2 reviews background material on Bayesian inference, Gaussian processes, information theory, the bandit and RL settings, and learning from qualitative feedback. Chapter 3 outlines the preference-based bandit and RL settings considered within the DPS framework, and Chapter 4 presents the DPS algorithm along its theoretical analysis and experiments. Chapter 5 discusses mixed-initiative learning for exoskeleton gait optimization. Finally, Chapter 6 draws conclusions and considers possible avenues for future extension.

Chapter 2

BACKGROUND

This chapter reviews background material that will be helpful for understanding the following chapters. First, Sections 2.1-2.3 review mathematical concepts in Bayesian inference, Gaussian processes, and information theory, respectively. Next, Section 2.4 discusses bandit learning and Section 2.5 reviews preference-based bandit learning. Finally, Section 2.6 surveys work in RL and preference-based RL.

2.1 Bayesian Inference for Parameter Estimation

Probability Theory

This subsection briefly introduces the concepts of a probability space, conditional probability, and Bayes' Theorem. A detailed exposition of these topics can be found in Grimmett and Stirzaker (2001).

Probability theory formalizes the notion of chance. For instance, the outcome of an experiment may be unknown in advance, and could be influenced by a number of random factors. To analyze such a situation, one can begin by listing all of the possible experimental outcomes, also known as events:

Definition 1 (Sample space)**.** *A* sample space *is the set of all possible outcomes of an experiment, and is denoted by* Ω*.*

For example, if the experiment consists of flipping a coin, then there are two possible outcomes: heads and tails, denoted by H and T, respectively. Then, the sample space is given by $\Omega = \{H, T\}$. In another example, if the experiment involves asking a person for a pairwise preference between two options ("Do you prefer A or B?"), then the sample space includes the two possible preferences that the person could give: $\Omega = \{A \succ B, B \succ A\}$, where $x \succ y$ denotes a preference for x over y.

In order to later define probabilities over collections of events, one must define event collections that are closed under the operation of taking countable unions. Such a collection of subsets of Ω is called a *σ-field*, and is defined as follows:

Definition 2 (σ-field)**.** *A* σ-field *is a collection* $\mathcal{F}$ *of subsets of* Ω *that satisfies the following three conditions:*

A) The empty set belongs to $\mathcal{F}$, that is, $\emptyset \in \mathcal{F}$.

B) For events $A_1, A_2, \ldots$, if $A_1, A_2, \ldots \in \mathcal{F}$, then $\bigcup_{j=1}^{\infty} A_j \in \mathcal{F}$.

C) For any event $A \in \mathcal{F}$, its complement $\Omega \setminus A$ also belongs to $\mathcal{F}$.

For example, the smallest σ-field associated with any sample space Ω is the collection: $\mathcal{F} = \{\emptyset, \Omega\}$. Meanwhile, the *power set* of Ω is the set of all possible subsets of Ω, and clearly must always be a σ-field. For instance, the power set of the earlier coin toss example, with $\Omega = \{H, T\}$, is $\mathcal{F} = \{\emptyset, \Omega, H, T\}$.

One can then define a *probability measure* over the members of a σ-field $\mathcal{F}$:

Definition 3 (Probability measures and spaces)**.** *A* probability measure *P on the tuple $(\Omega, \mathcal{F})$ is a function $P : \mathcal{F} \longrightarrow [0, 1]$ that satisfies the following requirements:*

A) The empty set has zero probability, and the entire sample space has a probability of one: $P(\emptyset) = 0$ and $P(\Omega) = 1$.

B) If $A_1, A_2, \ldots$ is a collection of members of $\mathcal{F}$ that are disjoint, that is, $A_i \cap A_j = \emptyset$ for all pairs j, k such that $j \neq k$, then:

$$P\left(\bigcup_{j=1}^{\infty} A_j\right) = \sum_{j=1}^{\infty} P(A_j).$$

A probability space *is a triple $(\Omega, \mathcal{F}, P)$ consisting of a sample space Ω, a σ-field $\mathcal{F}$ of subsets of Ω, and a probability measure P on $(\Omega, \mathcal{F})$.*

In many cases, it is useful to express an event's probability given the occurrence of a second event. This is formalized via the notion of *conditional probability*:

Definition 4 (Conditional probability)**.** *For two events A and B, if $P(B) > 0$, then the* conditional probability *that event A occurs given that event B occurs is defined as:*

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}. \tag{2.1}$$

Symmetrically, it holds that $P(B \mid A)P(A) = P(B \cap A)$. By substituting this fact into the right-hand side of Eq. (2.1), one obtains Bayes' Theorem, which states that for any two events A and B:

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}. \tag{2.2}$$

Bayesian Inference

Assume that we aim to estimate a vector $\boldsymbol{\theta} \in \mathbb{R}^d$ of unknown parameters belonging to a model. One can use Bayesian inference to probabilistically estimate $\boldsymbol{\theta}$ given the combination of data and any prior knowledge about $\boldsymbol{\theta}$. In Bayes' Theorem, given by Eq. (2.2), replacing A by the model parameters $\boldsymbol{\theta}$ and B by the observed dataset $\mathcal{D}$ yields:

$$P(\boldsymbol{\theta} \mid \mathcal{D}) = \frac{P(\mathcal{D} \mid \boldsymbol{\theta})P(\boldsymbol{\theta})}{P(\mathcal{D})}. \tag{2.3}$$

In Eq. (2.3), the quantity $P(\boldsymbol{\theta} \mid \mathcal{D})$ is known as the *model posterior*: this distribution yields the probability of each possible value of $\boldsymbol{\theta}$ conditioned upon the observed dataset. Meanwhile, $P(\boldsymbol{\theta})$ is known as the *prior*, and represents the prior belief about $\boldsymbol{\theta}$ before data is observed. The term $P(\mathcal{D} \mid \boldsymbol{\theta})$ is called the *likelihood*, and quantifies the probability of observing a dataset $\mathcal{D}$ given particular model parameters $\boldsymbol{\theta}$. Finally, the denominator of Eq. (2.3) contains the *evidence*, that is, the probability of the dataset, $P(\mathcal{D})$.

The *maximum a posteriori estimate* $\hat{\boldsymbol{\theta}}$, or MAP estimate, is defined as the mode of the posterior distribution. This is the parameter vector $\boldsymbol{\theta}$ which maximizes $P(\boldsymbol{\theta} \mid \mathcal{D})$:

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, P(\boldsymbol{\theta} \mid \mathcal{D}) = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, \frac{P(\mathcal{D} \mid \boldsymbol{\theta})P(\boldsymbol{\theta})}{P(\mathcal{D})} \overset{(a)}{=} \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, P(\mathcal{D} \mid \boldsymbol{\theta})P(\boldsymbol{\theta}),$$

where (a) holds because the evidence $P(\mathcal{D})$ does not depend on $\boldsymbol{\theta}$. In many posterior modeling situations, it is sufficient to use that $P(\boldsymbol{\theta} \mid \mathcal{D}) \propto P(\mathcal{D} \mid \boldsymbol{\theta})P(\boldsymbol{\theta})$, rather than explicitly calculating the evidence.

Posterior Inference with Conjugate Priors

When the prior and posterior belong to the same family of probability distributions, the prior $P(\boldsymbol{\theta})$ is said to be a *conjugate prior* for the likelihood, $P(\mathcal{D} \mid \boldsymbol{\theta})$. In such cases, the posterior $P(\boldsymbol{\theta} \mid \mathcal{D})$ can be straightforwardly calculated in closed-form. This section lists several examples of conjugate prior and likelihood pairs, which will appear at various future points in this dissertation.

In Murphy (2007), the author derives several conjugate priors for the Gaussian likelihood. Notably, the Gaussian distribution is self-conjugate, such that a Gaussian prior and likelihood result in a Gaussian posterior. For instance, consider the problem of estimating the mean μ of a univariate Gaussian distribution with known variance σ^2; in this case, the unknown model parameter vector is $\boldsymbol{\theta} = \mu \in \mathbb{R}$. A Gaussian prior on μ takes the form $\mu \mid \mu_0, \sigma_0 \sim \mathcal{N}(\mu_0, \sigma_0^2)$ for specified prior parameters μ_0

and σ_0. The likelihood assumes that observations are centered at μ with Gaussian noise of variance σ^2, such that given μ and σ, the i^{th} observation x_i is distributed as $x_i \mid \mu, \sigma \sim \mathcal{N}(\mu, \sigma^2)$. Then, for observations $\mathcal{D} = \{x_1, \ldots, x_N\}$, the full likelihood expression is:

$$P(\mathcal{D} \mid \mu, \sigma) = \prod_{i=1}^{N} p(x_i \mid \mu) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2\sigma^2}(x_i - \mu)^2}. \tag{2.4}$$

Under this prior and likelihood, Murphy (2007) shows that the posterior is also Gaussian, with distribution:

$$p(\mu \mid \mathcal{D}, \sigma^2, \mu_0, \sigma_0) = \mathcal{N}\left(\frac{1}{N\sigma_0^2 + \sigma^2}\left(\sigma^2\mu_0 + \sigma_0^2 \sum_{i=1}^{N} x_i\right), \frac{\sigma^2\sigma_0^2}{N\sigma_0^2 + \sigma^2}\right).$$

Next, consider Gaussian observations for which both the mean and variance are a priori unknown. In this case, the likelihood of the i^{th} observation x_i can be written as $x_i \sim \mathcal{N}(\mu, \lambda^{-1})$, where μ and the precision $\lambda := \sigma^{-2}$ are both unknown. The prior probabilities for μ and λ can be jointly modeled via a normal-gamma distribution:

$$\begin{aligned} &\mu, \lambda \sim NG(\mu_0, \kappa_0, \alpha_0, \beta_0), \text{ or equivalently,} \\ &\quad \lambda \sim \text{Gamma}(\alpha_0, \text{rate} = \beta_0), \ \ \mu \sim \mathcal{N}(\mu_0, (\kappa_0\lambda)^{-1}). \end{aligned}$$

It can be shown (Murphy, 2007) that under this normal-gamma prior and the Gaussian likelihood in Eq. (2.4), the posterior also takes a normal-gamma form, with the following parameters:

$$\begin{aligned} &p(\mu, \lambda \mid \mathcal{D}, \mu_0, \kappa_0, \alpha_0, \beta_0) = NG(\mu_N, \kappa_N, \alpha_N, \beta_N), \text{ where:} \\ &\quad \mu_N = \frac{\kappa_0\mu_0 + N\overline{x}}{\kappa_0 + N}, \ \ \kappa_N = \kappa_0 + N, \\ &\quad \alpha_N = \alpha_0 + \frac{N}{2}, \qquad \beta_N = \beta_0 + \frac{1}{2}\sum_{i=1}^{N}(x_i - \overline{x})^2 + \frac{\kappa_0 N(\overline{x} - \mu_0)^2}{2(\kappa_0 + N)}, \text{ and} \\ &\quad \overline{x} = \frac{1}{N}\sum_{i=1}^{N} x_i. \end{aligned}$$

While Gaussian distributions have infinite support, the beta distribution can serve as a conjugate prior to model variables with support on the $[0, 1]$ interval, such as probabilities. Consider binomial data consisting of a series of Bernoulli trials

with an unknown success probability, $\theta \in [0, 1]$. After any number of trials N, one observes some number W of positive results ("wins") and some number F of negative results ("failures"); thus, $\mathcal{D} = \{W, F\}$, where $W + F = N$. Note that the Bernoulli distribution is a special case of the binomial distribution, in which $N = 1$.

Because the success probability θ must lie within $[0, 1]$, the beta distribution is a reasonable prior: $\theta \mid \alpha_0, \beta_0 \sim \text{Beta}(\alpha_0, \beta_0)$, for some prior hyperparameters $\alpha_0, \beta_0 > 0$. The beta prior is conjugate to the Bernoulli and binomial likelihoods (Murphy, 2012), such that θ has the following exact posterior:

$$p(\theta \mid \mathcal{D}, \alpha_0, \beta_0) = \text{Beta}(\alpha_0 + W, \beta_0 + F).$$

In a final example, the Dirichlet distribution is a conjugate prior for the multinomial likelihood. This fact generalizes the conjugacy of the beta and binomial distributions to model multinomial data, in which each trial has one of K possible outcomes for some $K \geq 2$. Each trial exhibits outcome k with probability $\theta_k \in [0, 1]$, and so the vector of unknown model parameters is $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_K]^T$. The dataset $\mathcal{D}$ records the number of trials N_k corresponding to each outcome: $\mathcal{D} = \{N_1, \ldots, N_K\}$.

One can place a Dirichlet prior distribution upon $\boldsymbol{\theta}$. This distribution is supported over the $K - 1$ standard simplex, that is, over $x_1, \ldots, x_K \geq 0$ such that $\sum_{k=1}^{K} x_k = 1$. With the prior $\boldsymbol{\theta} \mid \alpha_1, \ldots, \alpha_K \sim \text{Dirichlet}(\alpha_1, \ldots, \alpha_K)$, for hyperparameters $\alpha_1, \ldots, \alpha_K > 0$, $\boldsymbol{\theta}$ has the following model posterior:

$$p(\boldsymbol{\theta} \mid \mathcal{D}, \alpha_1, \ldots, \alpha_K) = \text{Dirichlet}(\alpha_1 + N_1, \ldots, \alpha_K + N_K).$$

Approximate Posterior Inference

In many real-world situations, model posteriors not only lack convenient conjugate forms, but are analytically intractable and do not have closed-form representations. Fortunately, there are many techniques for approximating model posteriors, including the Laplace approximation (Murphy, 2012), expectation propagation (Minka, 2001; Minka and Lafferty, 2002), variational inference (Jordan, 1999; Wainwright and Jordan, 2008), and Markov Chain Monte Carlo (Metropolis et al., 1953; Murphy, 2012). The remainder of this subsection reviews the Laplace approximation method in detail.

The Laplace approximation to the posterior uses a second-order Taylor approximation to model the posterior distribution as Gaussian. To derive the Laplace

approximation, first note that the posterior $p(\boldsymbol{\theta} \mid \mathcal{D})$ can be written as follows:

$$p(\boldsymbol{\theta} \mid \mathcal{D}) = \frac{p(\boldsymbol{\theta}, \mathcal{D})}{P(\mathcal{D})} = \frac{e^{-[-\log p(\boldsymbol{\theta}, \mathcal{D})]}}{P(\mathcal{D})} = \frac{e^{-E(\boldsymbol{\theta})}}{P(\mathcal{D})}, \tag{2.5}$$

where $E(\boldsymbol{\theta}) := -\log p(\boldsymbol{\theta}, \mathcal{D})$. Performing a second-order Taylor approximation of $E(\boldsymbol{\theta})$ about the MAP estimate, $\hat{\boldsymbol{\theta}} = \operatorname{argmax}_{\boldsymbol{\theta}} p(\boldsymbol{\theta} \mid \mathcal{D})$, yields:

$$E(\boldsymbol{\theta}) \approx E(\hat{\boldsymbol{\theta}}) + (\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^T \left(\nabla_{\boldsymbol{\theta}} E(\boldsymbol{\theta}) \big|_{\hat{\boldsymbol{\theta}}} \right) + \frac{1}{2} (\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^T \left(\nabla^2_{\boldsymbol{\theta}} E(\boldsymbol{\theta}) \big|_{\hat{\boldsymbol{\theta}}} \right) (\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}). \tag{2.6}$$

The second term in Eq. (2.6) must be zero, because:

$$\hat{\boldsymbol{\theta}} = \operatorname*{argmax}_{\boldsymbol{\theta}} p(\boldsymbol{\theta} \mid \mathcal{D}) = \operatorname*{argmax}_{\boldsymbol{\theta}} \frac{e^{-E(\boldsymbol{\theta})}}{P(\mathcal{D})} = \operatorname*{argmax}_{\boldsymbol{\theta}} e^{-E(\boldsymbol{\theta})} = \operatorname*{argmin}_{\boldsymbol{\theta}} E(\boldsymbol{\theta}), \tag{2.7}$$

and $\nabla_{\boldsymbol{\theta}} E(\boldsymbol{\theta}) = 0$ in the place at which $E(\boldsymbol{\theta})$ is minimized. Thus, Eq. (2.6) becomes:

$$E(\boldsymbol{\theta}) \approx E(\hat{\boldsymbol{\theta}}) + \frac{1}{2} (\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^T H (\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}),$$

where $H := \nabla^2_{\boldsymbol{\theta}} E(\boldsymbol{\theta}) \big|_{\hat{\boldsymbol{\theta}}}$. Substituting this Taylor approximation into Eq. (2.5) yields:

$$p(\boldsymbol{\theta} \mid \mathcal{D}) = \frac{e^{-E(\boldsymbol{\theta})}}{P(\mathcal{D})} \approx \frac{e^{-E(\hat{\boldsymbol{\theta}})} e^{-\frac{1}{2}(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^T H (\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})}}{P(\mathcal{D})}.$$

The numerator of this fraction is an unnormalized Gaussian density with mean $\hat{\boldsymbol{\theta}}$ and covariance H^{-1}. In order for the expression to be a normalized probability density, one must set $P(\mathcal{D}) = e^{-E(\hat{\boldsymbol{\theta}})} (2\pi)^{\frac{d}{2}} |H|^{-\frac{1}{2}}$, where recall that d is the length of $\boldsymbol{\theta}$. Therefore, $p(\boldsymbol{\theta} \mid \mathcal{D}) \approx \mathcal{N}(\hat{\boldsymbol{\theta}}, H^{-1})$, where $H := \nabla^2_{\boldsymbol{\theta}} E(\boldsymbol{\theta}) \big|_{\hat{\boldsymbol{\theta}}}$.

Note that H must be positive semidefinite for the approximated posterior covariance matrix, H^{-1}, to be well-defined. Convexity of $E(\boldsymbol{\theta}) = -\log p(\boldsymbol{\theta}, \mathcal{D})$ over the domain of $\boldsymbol{\theta}$ is a necessary and sufficient condition for H to be positive semidefinite for any $\hat{\boldsymbol{\theta}}$. Thus, $E(\boldsymbol{\theta})$ must be convex in order for the Laplace approximation to be applicable. The MAP estimate in Eq. (2.7) can therefore be computed using any convex optimization solver.

2.2 Gaussian Processes

Gaussian processes (GPs) provide a flexible Bayesian approach for modeling smooth, nonparametric functions and for specifying probability distributions over spaces of such functions. Rasmussen and Williams (2006) give an excellent introduction to the use of Gaussian process methods in machine learning. This section reviews

the definition of a GP, as well as GP regression for modeling both numerical and preference data.

As stated in Rasmussen and Williams (2006), a Gaussian process is defined as a collection of random variables such that any finite subset of them have a joint Gaussian distribution. In our case, these random variables represent the values of an unknown function $f : \mathcal{A} \to \mathbb{R}$ evaluated over points $\boldsymbol{x} \in \mathcal{A}$, where $\mathcal{A} \subset \mathbb{R}^d$ is the domain of f. Then, the GP is fully specified by a mean function $\mu : \mathcal{A} \to \mathbb{R}$ and a covariance function $\mathcal{K} : \mathcal{A} \times \mathcal{A} \to \mathbb{R}_{\geq 0}$. For any $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{A}$, the mean and covariance functions are denoted as $\mu(\boldsymbol{x})$ and $\mathcal{K}(\boldsymbol{x}, \boldsymbol{x}')$, respectively. Then, if the function $f : \mathcal{A} \to \mathbb{R}$ is distributed according to this GP:

$$\begin{aligned} f(\boldsymbol{x}) &\sim \mathcal{GP}(\mu(\boldsymbol{x}), \mathcal{K}(\boldsymbol{x}, \boldsymbol{x}')), \text{ where:} \\ \mu(\boldsymbol{x}) &:= \mathbb{E}[f(\boldsymbol{x})], \text{ and} \\ \mathcal{K}(\boldsymbol{x}, \boldsymbol{x}') &:= \mathbb{E}[(f(\boldsymbol{x}) - \mu(\boldsymbol{x}))(f(\boldsymbol{x}') - \mu(\boldsymbol{x}'))]. \end{aligned}$$

Over any collection of points in the domain $\mathcal{A}$, the mean and covariance functions fully define the joint Gaussian distribution of f's values. To illustrate, define $X := [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N]^T \in \mathbb{R}^{N \times d}$ as a matrix in which $\boldsymbol{x}_i \in \mathcal{A}$ for each $i \in \{1, \ldots, N\}$. For this collection of points in $\mathcal{A}$, let $\mu(X) := [\mu(\boldsymbol{x}_1), \ldots, \mu(\boldsymbol{x}_N)]^T$ be a vector of the mean function values at each row in X, and let $K(X, X) \in \mathbb{R}^{N \times N}$ be such that $[K(X, X)]_{ij} = \mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j)$. The matrix $K(X, X)$ is called the Gram matrix. Furthermore, let $f(X) := [f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_N)]^T$ contain the values of f at each point in X. Then,

$$f(X) \sim \mathcal{N}(\mu(X), K(X, X)).$$

Thus, the GP is fully specified by its mean and covariance functions, μ and $\mathcal{K}$. Furthermore, the specific choices of μ and $\mathcal{K}$ define a distribution over the space of functions on $\mathcal{A}$. A common choice of μ is $\mu(\boldsymbol{x}) := 0$ for all $\boldsymbol{x}$: assuming a flat prior is reasonable without problem-specific knowledge, and setting it to zero simplifies notation. This dissertation will only consider zero-mean GP priors, in which $\mu(\boldsymbol{x}) := 0$. Non-zero GP mean functions are discussed in Chapter 2.7 of Rasmussen and Williams (2006), and can be useful in a number of situations, for instance to express prior information or to enhance model interpretability.

The covariance function $\mathcal{K}$, also called the kernel, defines the covariance between pairs of points in $\mathcal{A}$. The choice of $\mathcal{K}$ determines the strength of links between points in $\mathcal{A}$ at various distances from each other, and therefore characterizes the

smoothness and structure of the functions f sampled from the GP. This thesis will utilize the squared exponential kernel, which is defined as follows:

$$\mathcal{K}_{se}(\boldsymbol{x}_i, \boldsymbol{x}_j) := \sigma_f^2 \exp\left(-\frac{1}{2l^2}||\boldsymbol{x}_i - \boldsymbol{x}_j||^2\right),$$

where σ_f^2 is the signal variance and l is the lengthscale. The variables σ_f and l are hyperparameters, which must be either obtained from expert domain knowledge or learned, for instance via evidence maximization or cross-validation (discussed in Rasmussen and Williams, 2006). The signal variance, σ_f^2, reflects the prior belief about the function f's amplitude, while the lengthscale l captures f's sensitivity across distances in $\mathcal{A}$. Note that $\mathcal{K}_{se}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is maximized when $\boldsymbol{x}_i = \boldsymbol{x}_j$, and that as the distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ increases, $\mathcal{K}_{se}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ decays exponentially.

The choice of the kernel and settings of its hyperparameter values influence the types of functions likely to be sampled from the GP prior, and can convey expert knowledge. It is assumed that nearby points in the domain $\mathcal{A}$ have similar target values in f. In GP learning, the covariance kernel dictates this notion of nearness or similarity. Aside from the squared exponential kernel, other common types of kernels include the linear, periodic, and Matérn kernels.

When $d > 1$ (recall that d is the dimension of $\mathcal{A}$), the squared exponential kernel can be modified such that the lengthscale differs between dimensions; this is called the automatic relevance determination (ARD) kernel, and will also appear later in this thesis. It is defined as follows:

$$\mathcal{K}_{se,ARD}(\boldsymbol{x}_i, \boldsymbol{x}_j) := \sigma_f^2 \exp\left(-\frac{1}{2}\sum_{k=1}^{d}\left(\frac{[\boldsymbol{x}_i]_k - [\boldsymbol{x}_j]_k}{l_k}\right)^2\right),$$

where $[\boldsymbol{x}]_k$ denotes the k^{th} element of $\boldsymbol{x}$, and $l_1, \ldots, l_d$ are d lengthscale hyperparameters. Under the ARD kernel, functions sampled from the GP vary more slowly in some dimensions than others. If the lengthscales $l_1, \ldots, l_d$ are learned from data, then their resultant values yield insight into the relevance of each dimension (or feature) in the domain $\mathcal{A}$ to the output function; hence, the name "automatic relevance determination."

Importantly, an arbitrary function of two inputs will generally not be a valid covariance function. For a valid kernel, the Gram matrix $K(X, X)$ must be positive semidefinite for any set of points X. Such a kernel, for which the Gram matrix is guaranteed to be positive semidefinite, is called a positive semidefinite kernel. A

more complete discussion of this topic can be found in either Chapter 4 of Rasmussen and Williams (2006) or Chapter 12 of Wainwright (2019).

Every positive semidefinite kernel function $\mathcal{K}(\cdot,\cdot)$ corresponds to a unique reproducing kernel Hilbert space (RKHS) $\mathcal{H}_\mathcal{K}$ of real-valued functions associated with that kernel (Wainwright, 2019). The RKHS satisfies two properties: 1) for any fixed $\boldsymbol{x} \in \mathcal{A}$, the function $\mathcal{K}(\cdot,\boldsymbol{x})$ belongs to $\mathcal{H}_\mathcal{K}$, and 2) for any fixed $\boldsymbol{x} \in \mathcal{A}$, the function $\mathcal{K}(\cdot,\boldsymbol{x})$ satisfies the reproducing property, namely, that $< f, \mathcal{K}(\cdot,\boldsymbol{x}) >_{\mathcal{H}_\mathcal{K}} = f(\boldsymbol{x})$ for all $f \in \mathcal{H}_\mathcal{K}$ (Wainwright, 2019). The RKHS norm, $||f||_{\mathcal{H}_\mathcal{K}}$, captures information about the smoothness of functions f in the RKHS; f becomes smoother as $||f||_{\mathcal{H}_\mathcal{K}}$ decreases (Kanagawa et al., 2018).

Gaussian Process Regression

Gaussian process regression performs Bayesian inference on an unknown function f with a GP prior, when noisy observations of f are available. This subsection outlines the procedure for GP regression when the observations have Gaussian noise. The material is summarized from Chapter 2.3 of Rasmussen and Williams (2006), and additional details may be found therein.

First, assume that f has a GP prior: $f(\boldsymbol{x}) \sim \mathcal{GP}(\mu(\boldsymbol{x}), \mathcal{K}(\boldsymbol{x},\boldsymbol{x}'))$ for some μ and $\mathcal{K}$. Additionally, evaluations of f are observed, such that y_i denotes an observation corresponding to the i^{th} data point location $\boldsymbol{x}_i \in \mathcal{A}$. In most real-world situations, such observations are noisy. Assume that the observation noise is additive, Gaussian, independent, and identically-distributed. Then, $y_i = f(\boldsymbol{x}_i) + \varepsilon_i$, where $\varepsilon_i \sim \mathcal{N}(0, \sigma_n^2)$. The variable σ_n^2 is called the noise variance, and is an additional hyperparameter (alongside the kernel hyperparameters) which must either be learned from data or given by domain knowledge. Note that a noise variance of $\sigma_n^2 = 0$ indicates noise-free observations.

GP regression makes predictions about the values of f, given a dataset of N potentially-noisy function evaluations, $\{(\boldsymbol{x}_i, y_i) \mid i = 1,\ldots,N\}$. Let $\boldsymbol{y} := [y_1,\ldots,y_N]^T$ be the vector of observations. Recall that X is a matrix in which each row corresponds to an observation location: $X = [\boldsymbol{x}_1,\ldots,\boldsymbol{x}_N]^T$. Let $X_* \in \mathbb{R}^{M\times d}$ be a matrix in which each row corresponds to a point in $\mathcal{A}$ at which the value of f is to be predicted, and let $\boldsymbol{f}_* \in \mathbb{R}^M$ be a vector containing the values of f at the locations in X_*. Notationally, for any two collections X_A and X_B of points in $\mathcal{A}$, where $X_A = \left[\boldsymbol{x}_1^{(A)},\ldots,\boldsymbol{x}_n^{(A)}\right]^T \in \mathbb{R}^{n\times d}$, $X_B = \left[\boldsymbol{x}_1^{(B)},\ldots,\boldsymbol{x}_m^{(B)}\right]^T \in \mathbb{R}^{m\times d}$, define $K(X_A, X_B) \in \mathbb{R}^{n\times m}$ as a matrix such

that the ij^{th} element is $[K(X_A, X_B)]_{ij} = \mathcal{K}\left(\boldsymbol{x}_i^{(A)}, \boldsymbol{x}_j^{(B)}\right)$. Then, $\boldsymbol{y}$ and $\boldsymbol{f}_*$ have the following joint Gaussian distribution:

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{f}_* \end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} K(X,X) + \sigma_n^2 I & K(X, X_*) \\ K(X_*, X) & K(X_*, X_*) \end{bmatrix}\right), \tag{2.8}$$

where I is the N-dimensional identity matrix.

By the standard method for obtaining a conditional distribution from a joint Gaussian distribution (e.g., see Appendix A in Rasmussen and Williams, 2006), the distribution of $\boldsymbol{f}_*$ conditioned upon the observations $\boldsymbol{y}$ is also Gaussian:

$$\begin{aligned} \boldsymbol{f}_* \mid X, \boldsymbol{y}, X_* &\sim \mathcal{N}(\mu_{\boldsymbol{f}_*}, \Sigma_{\boldsymbol{f}_*}), \text{ where:} \\ \mu_{\boldsymbol{f}_*} &= K(X_*, X)[K(X,X) + \sigma_n^2 I]^{-1}\boldsymbol{y} \\ \Sigma_{\boldsymbol{f}_*} &= K(X_*, X_*) - K(X_*, X)[K(X,X) + \sigma_n^2 I]^{-1}K(X, X_*). \end{aligned}$$

Thus, the posterior over $\boldsymbol{f}_*$ can be computed in closed-form using the data (X, y), the kernel and its hyperparameters, and the noise variance σ_n^2. Note that the prior, likelihood, and posterior are all Gaussian; this is because the Gaussian distribution is self-conjugate, similarly to the case of univariate Gaussian data discussed in Section 2.1.

Gaussian Process Regression for Modeling Preference Data

In addition to regression, GPs have been applied toward a number of other modeling domains, for instance binary classification (Rasmussen and Williams, 2006) and ordinal regression (Chu and Ghahramani, 2005a). In particular, this subsection details a GP approach first proposed in Chu and Ghahramani (2005b) for Bayesian modeling of pairwise preference data, which will be applied later in this dissertation. Preference data (further discussed in Sections 2.5-2.6) can be useful when labels are given by humans: often, people cannot give reliable numerical scores, but *can* accurately respond to queries of the form, "Do you prefer A or B?"

Chu and Ghahramani (2005b) propose a GP-based approach in which preferences are given between elements of a space $\mathcal{A} \subset \mathbb{R}^d$, and each element $\boldsymbol{x} \in \mathcal{A}$ is assumed to have a latent utility $f(\boldsymbol{x})$ to the human user assigning the preferences. In other words, there exists a latent utility function $f : \mathcal{A} \longrightarrow \mathbb{R}$ that explains the preference data. A GP prior with an appropriate kernel $\mathcal{K}$ is placed over f: $f(\boldsymbol{x}) \sim \mathcal{GP}(\boldsymbol{0}, \mathcal{K}(\boldsymbol{x}, \boldsymbol{x}'))$.

Next, assume that $\mathcal{A}$ is a finite set with cardinality $A := |\mathcal{A}|$. The elements of $\mathcal{A}$ can then be indexed as $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_A$, and the values of f at the points in $\mathcal{A}$ can be written

in vectorized form: $\boldsymbol{f} := [f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), \ldots, f(\boldsymbol{x}_A)]^T$. The GP prior over $\boldsymbol{f}$ can be expressed as follows:

$$P(\boldsymbol{f}) = \frac{1}{(2\pi)^{A/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}\boldsymbol{f}^T\Sigma^{-1}\boldsymbol{f}\right),$$

where $\Sigma \in \mathbb{R}^{A\times A}$, $[\Sigma]_{ij} = \mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j)$, and $\mathcal{K}$ is a kernel function of choice, for instance the squared exponential kernel.

The dataset $\mathcal{D}$ consists of N pairwise preferences between points in $\mathcal{A}$: $\mathcal{D} = \{\boldsymbol{x}_{k1} \succ \boldsymbol{x}_{k2} \mid k = 1, \ldots, N\}$, where $\boldsymbol{x}_{k1}, \boldsymbol{x}_{k2} \in \mathcal{A}$ and $\boldsymbol{x}_{k1} \succ \boldsymbol{x}_{k2}$ indicates that the user prefers action $\boldsymbol{x}_{k1} \in \mathcal{A}$ to action $\boldsymbol{x}_{k2} \in \mathcal{A}$ in the k^{th} preference.

In modeling the likelihood $P(\mathcal{D} \mid \boldsymbol{f})$, the user's feedback is assumed to reflect the underlying utilities given by $\boldsymbol{f}$, but corrupted by i.i.d. Gaussian noise: when presented with action $\boldsymbol{x}_i$, the user determines her internal valuation $y(\boldsymbol{x}_i) = f(\boldsymbol{x}_i) + \varepsilon_i$, where $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$. Note that the preference noise parameter, σ, is a hyperparameter that must either be learned from data or set via domain knowledge. The likelihood of a specific preference given $\boldsymbol{f}$ is given by:

$$P(\boldsymbol{x}_{k1} \succ \boldsymbol{x}_{k2} \mid \boldsymbol{f}) = P(y(\boldsymbol{x}_{k1}) > y(\boldsymbol{x}_{k2}) \mid f(\boldsymbol{x}_{k1}), f(\boldsymbol{x}_{k2})) = \Phi\left(\frac{f(\boldsymbol{x}_{k1}) - f(\boldsymbol{x}_{k2})}{\sqrt{2}\sigma}\right),$$

where Φ is the standard normal cumulative distribution function, and $y(\boldsymbol{x}_{kj}) = f(\boldsymbol{x}_{kj}) + \varepsilon_{kj}$, $j \in \{1, 2\}$. Thus, the full expression for the likelihood is:

$$P(\mathcal{D}|\boldsymbol{f}) = \prod_{k=1}^{N} \Phi\left(\frac{f(\boldsymbol{x}_{k1}) - f(\boldsymbol{x}_{k2})}{\sqrt{2}\sigma}\right).$$

Intuitively, the user can easily give a reliable preference between two options with disparate utilities (i.e., $f(\boldsymbol{x}_{k1})$ and $f(\boldsymbol{x}_{k2})$ are far apart), while the closer the utilities of the two options, the more difficult it becomes for the user to give a correct preference. If the utilities of $\boldsymbol{x}_{k1}$ and $\boldsymbol{x}_{k2}$ are equal, then the user cannot distinguish between them at all, and $P(\boldsymbol{x}_{k1} \succ \boldsymbol{x}_{k2} \mid \boldsymbol{f}) = \Phi(0) = 0.5$.

The full model posterior takes the following form:

$$\begin{aligned} P(\boldsymbol{f} \mid \mathcal{D}) &\propto P(\mathcal{D} \mid \boldsymbol{f})P(\boldsymbol{f}) \\ &= \frac{1}{(2\pi)^{A/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}\boldsymbol{f}^T\Sigma^{-1}\boldsymbol{f}\right) \prod_{k=1}^{N} \Phi\left[\frac{f(\boldsymbol{x}_{k1}) - f(\boldsymbol{x}_{k2})}{\sqrt{2}\sigma}\right]. \end{aligned}$$

Unlike the posterior for GP regression with numerical rewards, discussed previously, the above posterior is non-Gaussian and lacks a tractable conjugate form. Because its

negative log-likelihood is convex, however, one can use the Laplace approximation (see Section 2.1) to approximate $P(\boldsymbol{f} \mid \mathcal{D})$ as a multivariate Gaussian distribution.

2.3 Entropy, Mutual Information, and Kullback-Leibler Divergence

This section defines several information-theoretic measures—namely, entropy, mutual information, and the Kullback-Leibler divergence—and states some of their properties, which will appear later in this dissertation. All of the facts below are proven for discrete random variables in Chapter 2 of Cover and Thomas (2012), and all random variables in this section are likewise discrete.

In the following, X is a random variable that takes values over a finite set $\mathcal{X}$ ($\mathcal{X}$ is called the alphabet of X); similarly, Y and Z are random variables with respective alphabets $\mathcal{Y}$ and $\mathcal{Z}$.

Firstly, *entropy* is a measure of a random variable's uncertainty. The Shannon entropy of X is defined as:

$$H(X) := -\sum_{x \in \mathcal{X}} P(X = x) \log P(X = x).$$

The entropy $H(X)$ is non-negative and upper-bounded in terms of $|\mathcal{X}|$:

Fact 1 (Non-negativity and upper-boundedness of entropy)**.** *It holds that:*

$$0 \leq H(X) \leq \log |\mathcal{X}|.$$

Note that $H(X) = 0$ when the distribution of X is a point mass at a particular element of $\mathcal{X}$—i.e., the distribution is maximally certain—while $H(X) = \log |\mathcal{X}|$ when $P(X = x) = \frac{1}{|\mathcal{X}|}$ for all $x \in \mathcal{X}$, i.e., the distribution of X is maximally uncertain.

The entropy of X conditioned upon $Y = y$ is defined as:

$$H(X \mid Y = y) := -\sum_{x \in \mathcal{X}} P(X = x \mid Y = y) \log P(X = x \mid Y = y),$$

and the conditional entropy of X given Y is:

$$\begin{aligned} H(X \mid Y) &= \sum_{y \in \mathcal{Y}} P(Y = y) H(X \mid Y = y) \\ &= -\sum_{y \in \mathcal{Y}} P(Y = y) \sum_{x \in \mathcal{X}} P(X = x \mid Y = y) \log P(X = x \mid Y = y) \\ &= \mathbb{E}_Y \left[-\sum_{x \in \mathcal{X}} P(X = x \mid Y) \log P(X = x \mid Y) \right]. \end{aligned}$$

Next, the Kullback-Leibler divergence is an (asymmetric) measure of distance between two probability distributions. The Kullback-Leibler divergence between two probability mass functions $\{P(X = x) \mid x \in \mathcal{X}\}$ and $\{Q(X = x) \mid x \in \mathcal{X}\}$ is denoted by $D(P \mid\mid Q)$ and is defined as:

$$D(P(X) \mid\mid Q(X)) = \sum_{x \in \mathcal{X}} P(X = x) \log \left(\frac{P(X = x)}{Q(X = x)} \right), \tag{2.9}$$

where by convention, $0 \log \frac{0}{0} = 0$, $0 \log \frac{0}{q} = 0$, and $p \log \frac{p}{0} = \infty$. Importantly, the Kullback-Leibler divergence is always non-negative:

Fact 2 (Non-negativity of Kullback-Leibler divergence)**.**

$$D(P(X) \mid\mid Q(X)) \geq 0,$$

with equality if and only if $P(X = x) = Q(X = x)$ *for all* $x \in \mathcal{X}$.

The *mutual information* of two random variables X and Y quantifies the amount of information learned about one of the variables when observing the value of the other. It is defined as:

$$\begin{aligned} I(X;Y) :&= \sum_{x \in \mathcal{X}, y \in \mathcal{Y}} P(X = x, Y = y) \log \left(\frac{P(X = x, Y = y)}{P(X = x)P(Y = y)} \right) \\ &= D(P(X,Y) \mid\mid P(X)P(Y)) = \mathbb{E}_{X,Y} \left[\log \left(\frac{P(X,Y)}{P(X)P(Y)} \right) \right]. \end{aligned}$$

Because the mutual information can be expressed as a Kullback-Leibler divergence, the following fact holds:

Fact 3 (Non-negativity of mutual information)**.** $I(X;Y) \geq 0$*, and* $I(X;Y) = 0$ *if and only if* X *and* Y *are independent.*

One can show that the mutual information can be written as a difference of entropies:

Fact 4 (Entropy reduction form of mutual information)**.** *The mutual information between* X *and* Y *can be written as* $I(X;Y) = H(X) - H(X \mid Y)$*. Conditioned on a third random variable,* Z*, the mutual information between* X *and* Y *can be written:* $I(X;Y \mid Z) = H(X \mid Z) - H(X \mid Y, Z)$.

The mutual information between X and a collection of random variables $Y_1, \ldots, Y_n$ can be simplified via the *chain rule for mutual information*:

Fact 5 (Chain rule for mutual information)**.**

$$I(X;(Y_1,\ldots,Y_n)) = \sum_{i=1}^{n} I(X;Y_i \mid Y_1 \ldots, Y_{i-1}).$$

Next, the following result from Russo and Van Roy (2016) expresses the mutual information in terms of a Kullback-Leibler divergence:

Fact 6 (Mutual information in terms of Kullback-Leibler divergence)**.** *The mutual information between X and Y can be expressed as:*

$$I(X;Y) = \mathbb{E}_X[D(P(Y \mid X) \mid\mid P(Y))] = \sum_{x \in \mathcal{X}} P(X = x) D(P(Y \mid X = x) \mid\mid P(Y)).$$

Finally, Russo and Van Roy (2016) applies Pinsker's inequality to relate the Kullback-Leibler divergence to a difference of expectations:

Fact 7 (Relating the Kullback-Leibler divergence to a difference of expectations; Fact 9 in Russo and Van Roy, 2016)**.** *Let P and Q be two probability distributions such that P is absolutely continuous with respect to Q, let X be any random variable taking values on the set $\mathcal{X}$, and let $g : \mathcal{X} \longrightarrow \mathbb{R}$ be any function such that* $\sup g - \inf g \leq 1$. *Then, with $\mathbb{E}_P$ and $\mathbb{E}_Q$ denoting the expectations under P and Q:*

$$D(P \mid\mid Q) \geq 2\big(\mathbb{E}_P[g(X)] - \mathbb{E}_Q[g(X)]\big)^2.$$

2.4 Bandit Learning

The multi-armed bandit problem, first studied in Robbins (1952), is a sequential decision-making problem in which an agent interacts with an environment by taking a series of actions in the environment. In each step, the agent observes a reward corresponding to the action that it selected. This section first reviews the K-armed stochastic bandit problem, and then discusses several extensions of this setting.

In the K-armed bandit setting (Auer, Cesa-Bianchi, and Fischer, 2002), the environment consists of an action space $\mathcal{A}$ with finite cardinality, $|\mathcal{A}| = A$. The actions are indexed such that $\mathcal{A} = \{1, \ldots, A\}$. In each learning iteration i, the agent selects a specific action $a_i \in \mathcal{A}$ and observes a reward $R_{a_i,i}$ corresponding to action a_i. The rewards for a particular action $a \in \{1, \ldots, A\}$ at each time step are denoted by $R_{a,1}, R_{a,2}, \ldots$, and are assumed to be independent and identically distributed (i.i.d.) samples from the distribution ν_a. Importantly, rewards for different actions are independent, that is, R_{a,i_1} and R_{a',i_2} are independent for any $i_1, i_2 \geq 1$ and for

any $a \neq a'$ such that $a, a' \in \mathcal{A}$; thus, observing action a's reward does not reveal information about any other actions' rewards.[1]

Denote μ_a to be the expected reward when selecting action a: $\mu_a := \mathbb{E}[R_{a,i}]$ for all i. Then, the index of the optimal action is $a^* := \text{argmax}_{a \in \{1,\ldots,A\}} \mu_a$. Let μ^* be its corresponding expected reward: $\mu^* := \max_{a \in \{1,\ldots,A\}} \mu_a = \mu_{a^*}$.

The agent seeks to maximize its cumulative reward over time: $\sum_{i=1}^{T} R_{a_i,i}$. This is equivalent to minimizing the *regret*, or the gap between the algorithm's total reward and the total reward accumulated by repeatedly choosing the optimal action a^*. The regret is defined as follows over T learning iterations:

$$\text{Reg}(T) := \sum_{i=1}^{T} [\mu^* - \mu_{a_i}] = \mu^* T - \sum_{i=1}^{T} \mu_{a_i} = \sum_{i=1}^{T} \Delta_{a_i},$$

where $\Delta_a := \mu^* - \mu_a$ is the expected per-iteration contribution to the regret when choosing action a, and is also known as the expected *instantaneous regret* under action a. In expectation,

$$\mathbb{E}[\text{Reg}(T)] := \mathbb{E}\left[\sum_{i=1}^{T} \Delta_{a_i}\right] = \sum_{a=1}^{A} \Delta_a \mathbb{E}[T_a(T)],$$

where $T_a(t)$ is the number of times that action a is selected in the first t steps, and where the expectation is taken over any randomness in the action selection strategy and in the rewards (which could affect future action selection).

The literature on the stochastic bandit problem can be divided into two approaches toward analyzing regret, as discussed in Kaufmann, Cappé, and Garivier (2012): frequentist and Bayesian. In the frequentist setting, the reward distributions for each action are assumed to be fixed and deterministic. Meanwhile, under the Bayesian viewpoint, the environment variables are assumed to be drawn from a prior over all possible environments. In particular, in the K-armed bandit setting, the environment is determined by the distributions $\{\nu_1, \ldots, \nu_A\}$. While the frequentist perspective assumes that $\{\nu_1, \ldots, \nu_A\}$ are fixed, unknown distributions, a Bayesian framework would assume that each distribution ν_a is defined by a parameter vector $\boldsymbol{\theta}_a$, which is drawn from a prior distribution: $\boldsymbol{\theta}_a \sim p(\boldsymbol{\theta}_a)$. The Bayesian regret is:

$$\text{BayesReg}(T) := \mathbb{E}\left[\sum_{i=1}^{T} \Delta_{a_i}\right] = \sum_{a=1}^{A} \Delta_a \mathbb{E}[T_a(T)],$$

[1] There are many versions of the bandit setting in which different actions' rewards *are* dependent on one another; some of these will be discussed later.

where in addition to randomness in the rewards and in the action selection scheme, the expectation is taken with respect to randomness in the environment, captured by $\boldsymbol{\theta}_a \sim p(\boldsymbol{\theta}_a)$ for $a \in \{1, \ldots, A\}$.

The regret minimization problem in the bandit setting was first formulated in Robbins (1952) (regret is called "loss" in that work).

Bandit Algorithms

The first multi-armed bandit algorithm with an asymptotically-optimal regret guarantee was proposed in Lai and Robbins (1985). This seminal work demonstrated that under mild assumptions on the reward distributions, within T total steps, any suboptimal action $a \neq a^*$ is selected an asymptotically lower-bounded number of times, $T_a(T)$:

$$\mathbb{E}[T_a(T)] \geq \frac{\log T}{D(\nu_a \,||\, \nu^*)}, \tag{2.10}$$

where $\nu^* = \nu_{a^*}$ is the reward distribution of the optimal action a^* and $D(p \,||\, q)$ is the Kullback-Leibler divergence between continuous probability distributions p and q, defined as:

$$D(p \,||\, q) := \int_{-\infty}^{\infty} p(x) \log\left(\frac{p(x)}{q(x)}\right) dx,$$

where p and q are continuous distributions over the real numbers. In addition, Lai and Robbins propose an algorithm for which $\mathbb{E}[T_a(T)] \leq \left(\frac{1}{D(\nu_a \,||\, \nu^*)} + o(1)\right) \log T$, where $o(1) \to 0$ as $T \to \infty$. Comparing with Eq. (2.10), one can see that this approach is asymptotically optimal, and furthermore, that the optimal action a^* is asymptotically selected exponentially more often than any suboptimal action.

While Lai and Robbins (1985) demonstrate that one can achieve logarithmic regret asymptotically, Gittins (1974) and Gittins (1979) proposed a provably-optimal dynamic programming strategy for the Bayesian setting with known priors and time-discounted rewards, in which the algorithm seeks to maximize $\mathbb{E}[\sum_{i=1}^{T} \gamma^{i-1} R_{a_i,i}]$ for a discount factor $\gamma \in (0, 1)$. This computationally-intensive strategy uses dynamic programming to compute dynamic allocation indices, or Gittins indices, for each action at every step. Later, Auer, Cesa-Bianchi, and Fischer (2002) were first to propose computationally-efficient algorithms for achieving logarithmic regret in finite time; their analysis only requires that the reward distributions have bounded support. The following subsections detail several regret minimization strategies for the bandit setting.

Tackling the Exploration-Exploitation Trade-Off

Every algorithm in the bandit setting must balance between selecting actions thought to yield high rewards and exploring actions whose outcomes are uncertain. There are numerous strategies in the literature for tackling this trade-off between exploration and exploitation; among others, these include upper confidence bound algorithms, posterior sampling, and information-theoretic approaches. Each of these three algorithmic design methodologies is reviewed below.

Upper-Confidence Bound Algorithms

Upper confidence bound algorithms leverage the principle of *optimism in the face of uncertainty*, in which actions are selected as if all rewards are equal to their maximum statistically-plausible values. The idea first appeared in Lai and Robbins (1985), in which an upper confidence index is associated with each action; the algorithm alternates between selecting the action whose confidence index is largest and selecting the "leader," that is, the action with the largest estimated mean among sufficiently-observed actions. The procedure for computing the confidence indices is computationally inefficient, however, and utilizes the entire history of rewards observed for a given action.

Subsequent upper confidence bound approaches, for example the UCB1 and UCB2 algorithms introduced in Auer, Cesa-Bianchi, and Fischer (2002), select an action a_{i+1} at step $i+1$ as follows:

$$a_{i+1} = \underset{a \in \{1,\ldots,A\}}{\operatorname{argmax}} \left[\hat{\mu}_{a,i} + \hat{\sigma}_{a,i}\right],$$

where $\hat{\mu}_{a,i}$ is the empirical mean reward observed for action a after i steps, while $\hat{\sigma}_{a,i}$ estimates the uncertainty in action a's reward. Thus, the first term prioritizes selecting the actions that appear most promising, while the second term favors actions with more uncertain reward distributions. There are a number of possible definitions for $\hat{\sigma}_{a,i}$. For instance, the UCB1 algorithm uses $\hat{\sigma}_{a,i} = \sqrt{\frac{2 \log i}{T_a(i)}}$, where $T_a(i)$ is the number of times that action a has been selected at step i. The UCB1 and UCB2 algorithms have expected regret values of $O\left(\sum_{a:\mu_a<\mu^*} \frac{\log T}{\Delta_a} + \sum_{j=1}^{A} \Delta_j\right)$ and $O\left(\sum_{a:\mu_a<\mu^*} \left(\frac{\log(\Delta_a^2 T)}{\Delta_a} + \frac{1}{\Delta_a}\right)\right)$, respectively.

More recently, a number of UCB-type algorithms have improved upon the regret bounds given in Auer, Cesa-Bianchi, and Fischer (2002). These include Audibert, Munos, and Szepesvári (2009), Audibert and Bubeck (2010), Garivier and Cappé (2011), and Kaufmann, Cappé, and Garivier (2012).

Posterior Sampling

Posterior sampling, also known as Thompson sampling, is a Bayesian alternative to the upper confidence bound approach. First proposed by Thompson (1933), posterior sampling selects each action according to its probability of being optimal. The method iterates between maintaining a posterior distribution over possible environments and sampling from this posterior to inform action selection. Intuitively, when the algorithm is less certain, it samples from diffuse distributions encouraging exploration, while when it becomes more certain of the optimal action, it samples from peaked distributions leading to exploitation.

In the strictly Bayesian setting, posterior sampling assumes that each action has a prior probability distribution over its reward, which given reward observations, is updated to obtain a model posterior over the rewards (e.g., Russo and Van Roy, 2016). Importantly, however, the sampling distributions need not be exact Bayesian posteriors, and in a number of cases are not (e.g., Abeille and Lazaric, 2017; Agrawal and Goyal, 2013; Sui, Zhuang, et al., 2017). In particular, Abeille and Lazaric (2017) explain that posterior sampling does not necessarily need to sample from an actual Bayesian posterior: in fact, any probability distribution satisfying appropriate concentration and anti-concentration inequalities will achieve low regret. As a result, although posterior sampling is an apparently-Bayesian approach, it can enjoy bounded regret in the frequentist setting as well as the Bayesian setting.

To illustrate posterior sampling, consider the Bernoulli bandits problem, in which the actions yield binary rewards: $R_{a,i} \in \{0, 1\}$ for all $a \in \{1, \ldots, A\}$, $i \geq 1$. A reward of 1 is regarded as a "success," while a reward of 0 is a "failure." This is analogous to a game in which each action tosses a corresponding coin with unknown probability of landing on heads. The player selects one coin to flip per step, and receives a reward of 1 if it lands on heads, and 0 if it lands on tails. The player's total reward is thus the total number of times that a coin toss results in heads.

Let $\theta_a = P(r_{a,i} = 1)$ denote action a's success probability (analogously, a coin's probability of landing on heads). The Beta distribution can be used to model the algorithm's belief over $\{\theta_1, \ldots, \theta_A\}$:

$$\theta_a \sim \text{Beta}(\alpha_a, \beta_a),\ a \in \{1, \ldots, A\},$$

where α_a, β_a are prior parameters. Without any prior information about the rewards, one could set $\alpha_a = \beta_a = 1$ for each action $a \in \{1, \ldots, A\}$; the Beta$(1, 1)$ prior distribution is equivalent to a uniform distribution over the interval $[0, 1]$.

Algorithm 1 Posterior Sampling for Bernoulli Bandits (Agrawal and Goyal, 2012)

For each action $a = 1, 2, \cdots, A$, set $W_a = 0$, $F_a = 0$.
for $i = 1, 2, \ldots$ **do**
    For each action $a = 1, 2, \cdots, A$, sample $\theta_a \sim \text{Beta}(W_a + 1, F_a + 1)$
    Execute action $a_i := \text{argmax}_a\, \theta_a$, and observe reward $r_i \in \{0, 1\}$
    $W_{a_i} \leftarrow W_{a_i} + r_i$, $F_{a_i} \leftarrow F_{a_i} + 1 - r_i$
end for

The Beta distribution is a convenient choice of prior: firstly, it is supported only on the interval $[0, 1]$, which is a necessary condition for modeling probabilities, and secondly, it is conjugate to the Bernoulli and Binomial distributions. Thus, if a Beta prior is coupled with Binomial data, then the posterior also has a Beta distribution. For each action, the corresponding data consists of the numbers of successes and failures observed for that action, and thus is naturally modeled via the Binomial distribution. In particular, after observing W_a successes, or wins, and F_a failures, or losses, one can show that the conjugate posterior takes the following form:

$$\theta_a \sim \text{Beta}(\alpha_a + W_a, \beta_a + F_a),\ a \in \{1, \ldots, A\}. \tag{2.11}$$

Algorithm 1 details the posterior sampling algorithm for this Bernoulli bandit problem, as given in Agrawal and Goyal (2012). The variables W_a and F_a keep track of the numbers of successes and failures observed for each action. The algorithm then alternates between sampling from the actions' posteriors in Eq. (2.11), selecting the action a_i with the highest sampled reward, and updating the posterior parameters W_{a_i} and F_{a_i} given the observed reward.

Agrawal and Goyal (2012) also propose a more general posterior sampling algorithm for the K-armed bandit setting, in which rewards are not necessarily Bernoulli, but rather can lie in the entire $[0, 1]$ interval. In this procedure, detailed in Algorithm 2, the algorithm similarly maintains variables W_a and F_a for each action and draws samples from the posterior in Eq. (2.11). In contrast, however, after observing each reward $\tilde{r}_i \in [0, 1]$, the algorithm samples a Bernoulli random variable r_i with success probability $\tilde{r}_i$, such that $r_i \sim \text{Bernoulli}(\tilde{r}_i)$, and updates the counts W_{a_i} and F_{a_i} just as in Algorithm 1. Note that Algorithm 1 is a special case of Algorithm 2, and also that it is straightforward to adapt Algorithms 1 and 2 such that different actions have different prior parameters α_a and β_a.

A number of studies demonstrate that posterior sampling is not only empirically competitive with upper confidence bound approaches, but it often outperforms them

Algorithm 2 Posterior Sampling for general stochastic bandits (Agrawal and Goyal, 2012)

For each action $a = 1, 2, \cdots, A$, set $W_a = 0$, $F_a = 0$.
for $i = 1, 2, \ldots$ **do**
    For each action $a = 1, 2, \cdots, A$, sample $\theta_a \sim \text{Beta}(W_a + 1, F_a + 1)$
    Execute action $a_i := \text{argmax}_a \, \theta_a$, and observe reward $\tilde{r}_i \in \{0, 1\}$
    Sample $r_i \sim \text{Bernoulli}(\tilde{r}_i)$
    $W_{a_i} \leftarrow W_{a_i} + r_i$, $F_{a_i} \leftarrow F_{a_i} + 1 - r_i$
end for

(Chapelle and Li, 2011; Granmo, 2010; May and Leslie, 2011; May, Korda, et al., 2012). Granmo (2010) proves that Algorithm 1 with $A = 2$ is asymptotically consistent, that is, its probability of selecting the optimal action approaches 1.

Agrawal and Goyal (2012) derive the first regret analysis for posterior sampling. Firstly, for $A = 2$, the authors demonstrate that posterior sampling has an expected regret of $O\left(\frac{\log T}{\Delta} + \frac{1}{\Delta^3}\right)$, where $\Delta = |\mu_1 - \mu_2|$. Secondly, for any number of actions A, the expected regret of posterior sampling is upper-bounded by $O\left(\left(\sum_{a:\mu_a<\mu^*} \frac{1}{\Delta_a^2}\right)^2 \log T\right)$, where recall that $\Delta_a := \mu^* - \mu_a$. These bounds have a finite-time logarithmic dependence in the total time T, which matches the lower bound for this setting.

Meanwhile, Russo and Van Roy (2016) present an information-theoretic perspective on bounding the Bayesian regret of posterior sampling algorithms. In particular, this work introduces a quantity called the *information ratio*, denoted by Γ_i:

$$\Gamma_i := \frac{\mathbb{E}[\mu^* - \mu_{a_i} \mid \mathcal{H}_i]^2}{I(a^*; (a_i, R_{a_i,i}) \mid \mathcal{H}_i)}, \tag{2.12}$$

where $\mathcal{H}_i := \{a_1, R_{a_1,1}, \ldots, a_{i-1}, R_{a_{i-1},i-1}\}$ is the history of actions and rewards observed during the first $i-1$ time-steps of running the algorithm. The numerator of Γ_i is the squared expected instantaneous regret given the history, while the denominator is the mutual information between the optimal action a^* and the information gained during the current iteration i (see Section 2.3 for the definition of mutual information). Smaller values of Γ_i are preferred: intuitively, if the information ratio is small, then in each iteration, the algorithm must either incur little instantaneous regret or gain a significant amount of information about the optimal action. Thus, a bounded information ratio directly balances between exploration and exploitation.

Russo and Van Roy (2016) upper-bound the Bayesian regret of the bandit problem

in terms of the information ratio:

$$\text{BayesReg}(T) \leq \sqrt{\overline{\Gamma} H(a^*) T}, \tag{2.13}$$

where $H(a^*)$ is the entropy of the optimal action, and $\overline{\Gamma}$ is a uniform upper-bound upon the information ratio. By Fact 1, $H(a^*) \leq \log |\mathcal{A}| = \log A$. Russo and Van Roy (2016) demonstrate that in the K-armed bandit problem with A actions, posterior sampling achieves $\overline{\Gamma} \leq \frac{A}{2}$. This results in the following Bayesian regret bound for posterior sampling in the K-armed bandit setting:

$$\text{BayesReg}(T) \leq \sqrt{\frac{1}{2} AT \log A}.$$

This regret bound has an optimal dependence upon the number of actions A and time steps T neglecting logarithmic factors, as Bubeck and Liu (2013) show that for any T and A, there exists a prior over the environment such that $\text{BayesReg}(T) \geq \frac{1}{20}\sqrt{AT}$. Beyond the K-armed bandit problem, Russo and Van Roy (2016) analyze the information ratio of posterior sampling in several more problem settings, one of which will be discussed later.

Information-Theoretic Regret Minimization Approaches

This section discusses several information-theoretic approaches to guiding action selection in the bandit problem. Firstly, the knowledge gradient algorithm makes a series of greedy decisions targeted at improving decision quality. The approach, first proposed in Gupta and Miescke (1996) and further studied in Frazier, Powell, and Dayanik (2008) and Ryzhov, Powell, and Frazier (2012), at each step selects the action predicted to maximally improve the expected reward of an exploit-only strategy (i.e., a strategy which terminates learning and repeatedly chooses the predicted optimal action). As discussed in Russo and Van Roy (2014a), however, the knowledge gradient algorithm lacks general theoretical guarantees, and one can construct cases in which it fails to identify the optimal action.

Secondly, Russo and Van Roy (2014a) propose the *information-directed sampling* framework, which uses the information ratio Γ_i, defined in Eq. (2.12), to guide action selection in the Bayesian setting. At each time step i, information-directed sampling selects the action which minimizes the information ratio, Γ_i. The minimum-possible Γ_i cannot exceed the upper-bound for Γ_i derived for posterior sampling in Russo and Van Roy (2016). Therefore, using the relationship between regret and Γ_i given by Eq. (2.13), one can argue that the regret bounds for posterior sampling proven

in Russo and Van Roy (2016) also hold for information-directed sampling. Empirically, however, information-directed sampling often outperforms posterior sampling (Russo and Van Roy, 2014a). As discussed in Russo and Van Roy (2014a), under certain types of information structures, information-directed sampling can be more sample efficient than upper confidence bound or posterior sampling approaches.

While the analyses in Russo and Van Roy (2016) and Russo and Van Roy (2014a) are restricted to the Bayesian regret setting, Kirschner and Krause (2018) define a frequentist counterpart to the information ratio, termed the *regret-information ratio*, and leverage it to develop a version of information-directed sampling with frequentist regret guarantees.

Bandits with Structured Reward Functions

Strategies for the K-armed bandit setting assume that different actions' outcomes are independent; as a result, regret bounds depend polynomially on the number of actions, and algorithms become highly-inefficient in large action spaces. While such strategies are infeasible for learning over exponentially or infinitely-large actions sets, a number of efficient learning algorithms have been developed for cases where rewards have a lower-dimensional structure. This subsection reviews work corresponding to several such reward structures.

For example, in the linear bandits problem, each action is represented by a d-dimensional feature vector. The expected reward for executing action $\boldsymbol{x} \in \mathcal{A} \subset \mathbb{R}^d$ is denoted by $R(\boldsymbol{x})$, and is a linear combination of these features:

$$\mathbb{E}[R(\boldsymbol{x})] = \overline{\boldsymbol{r}}^T \boldsymbol{x},$$

where $\overline{\boldsymbol{r}} \in \mathbb{R}^d$ is an unknown model parameter vector defining the rewards. Note that the action space $\mathcal{A} \subset \mathbb{R}^d$ could potentially be infinite in this setting.

A number of works study upper confidence bound approaches for the linear bandits problem (Dani, Hayes, and Kakade, 2008; Rusmevichientong and Tsitsiklis, 2010; Abbasi-Yadkori, Pál, and Szepesvári, 2011) and derive frequentist regret bounds. At a high level, these approaches maintain a confidence region $\mathcal{R}_i$ which contains $\overline{\boldsymbol{r}}$ with high probability at each step i. Actions are then selected optimistically, such that at each step i, the algorithm selects action $\boldsymbol{x}_i$ via:

$$\boldsymbol{x}_i = \underset{\boldsymbol{r} \in \mathcal{R}_i, \boldsymbol{x} \in \mathcal{A}}{\operatorname{argmax}} \boldsymbol{r}^T \boldsymbol{x}. \tag{2.14}$$

Among confidence-based linear bandit algorithms, smaller confidence regions $\mathcal{R}_i$ result in tighter regret bounds. In particular, Abbasi-Yadkori, Pál, and Szepesvári (2011) introduced a novel martingale concentration inequality that enables a vastly-improved regret guarantee in comparison to prior work, for instance in Dani, Hayes, and Kakade (2008) and Rusmevichientong and Tsitsiklis (2010). In particular, Abbasi-Yadkori, Pál, and Szepesvári (2011) develop the OFUL algorithm for the linear bandit problem, and prove that with probability at least $1-\delta$, its regret satisfies $\text{Reg}(T) \leq O(d \log T \sqrt{T} + \sqrt{dT \log(T/\delta)})$, where δ is an algorithmic parameter. The authors further prove a problem-dependent regret bound: with probability at least $1-\delta$, $\text{Reg}(T) \leq O\left(\frac{\log(1/\delta)}{\Delta}(\log T + d \log\log T)^2\right)$, where Δ (as defined in Dani, Hayes, and Kakade, 2008) can be intuitively understood as the gap in reward between the best and second-best corner decisions.[2] Formally, Δ is defined in terms of the set of extremal points $\mathcal{E}$, which consists of all the points in $\mathcal{A}$ that cannot be expressed as a convex combination of any other points in $\mathcal{A}$. The optimal action $\boldsymbol{x}^* = \text{argmax}_{\boldsymbol{x}\in\mathcal{A}} \overline{\boldsymbol{r}}^T \boldsymbol{x}$ is guaranteed to belong to $\mathcal{E}$. Then, defining $\mathcal{E}_- := \mathcal{E} \setminus \boldsymbol{x}^*$, the gap is given by:

$$\Delta = \sup_{\boldsymbol{x}\in\mathcal{E}_-} \overline{\boldsymbol{r}}^T \boldsymbol{x}^* - \overline{\boldsymbol{r}}^T \boldsymbol{x}.$$

Meanwhile, posterior sampling algorithms for linear bandits (Agrawal and Goyal, 2013; Abeille and Lazaric, 2017) place a prior upon $\overline{\boldsymbol{r}}$, $\overline{\boldsymbol{r}} \sim p(\boldsymbol{r})$, and maintain a model posterior over $\overline{\boldsymbol{r}}$, $p(\boldsymbol{r} \mid \mathcal{D})$, where $\mathcal{D}$ is the evolving dataset of agent-environment interactions. For instance, this can be performed via Bayesian linear regression. In each iteration, the algorithm samples $\tilde{\boldsymbol{r}} \sim p(\boldsymbol{r} \mid \mathcal{D})$, and chooses the optimal action as if $\tilde{\boldsymbol{r}}$ is the model parameter vector: $\boldsymbol{x}_i = \text{argmax}_{\boldsymbol{x}\in\mathcal{A}} \tilde{\boldsymbol{r}}^T \boldsymbol{x}$.

Agrawal and Goyal (2013) and Abeille and Lazaric (2017) both provide high-probability frequentist regret bounds for posterior sampling-based algorithms for the linear bandits problem. Both of these analyses upper-bound the T-step regret by $\tilde{O}(d^{3/2}\sqrt{T})$ with high probability, where the $\tilde{O}$-notation hides logarithmic factors. More specifically, Agrawal and Goyal (2013) prove that with probability at least $1-\delta$, $\text{Reg}(T) \leq O(d^{3/2}\sqrt{T}(\log T \sqrt{\log T + \log(1/\delta)}))$ and in addition, that $\text{Reg}(T) \leq O(d\sqrt{T \log A}(\log T \sqrt{\log T + \log(1/\delta)}))$.

Notably, in the linear bandit setting, posterior sampling is more computationally efficient than confidence-based approaches; the latter involve solving the optimization problem in Eq. (2.14), which is more costly than updating a model posterior

[2] Note that Δ is only well-defined for polytopic action spaces, and not, e.g., for spherical ones.

and sampling from it via the procedure in Agrawal and Goyal (2013) or Abeille and Lazaric (2017).

While the preceding analyses upper-bound the frequentist regret with high probability, Russo and Van Roy (2016) and Dong and Van Roy (2018) derive Bayesian regret bounds for the linear bandits problem by upper-bounding the information ratio for this setting. In particular, Dong and Van Roy (2018) derive a Bayesian regret bound of $\text{BayesReg}(T) \leq d\sqrt{T \log\left(3 + \frac{3\sqrt{2T}}{d}\right)}$, which is competitive with other results for the Bayesian linear bandits setting, for instance the Bayesian regret analysis in Russo and Van Roy (2014b).

More generally, the linear bandit problem is a special case of the generalized linear bandit setting, in which the reward feedback $R(\boldsymbol{x})$ for each action $\boldsymbol{x} \in \mathcal{A} \subset \mathbb{R}^d$ is generated according to:

$$\mathbb{E}[R(\boldsymbol{x})] = g(\overline{\boldsymbol{r}}^T \boldsymbol{x}),$$

where $\overline{\boldsymbol{r}} \in \mathbb{R}^d$ is an unknown model parameter vector, and g is a monotonically-increasing link function satisfying appropriate conditions; for instance, a sigmoidal g results in the logistic bandit problem. For this setting, Filippi et al. (2010) provide a regret bound for a confidence set approach, while Abeille and Lazaric (2017) bound the regret for posterior sampling and Dong and Van Roy (2018) bound the Bayesian regret for posterior sampling, again by deriving an upper bound for the problem's information ratio.

Meanwhile, in the Gaussian process bandit setting (Srinivas et al., 2010; Chowdhury and Gopalan, 2017), the reward function is assumed to be an arbitrary, non-parametric, smooth function of the action space features. More formally, $\mathbb{E}[R(\boldsymbol{x})] = f(\boldsymbol{x})$ for $\boldsymbol{x} \in \mathcal{A} \subset \mathbb{R}^d$, and a Gaussian process prior is placed over the unknown function f, such that $f(\boldsymbol{x}) \sim \mathcal{GP}(\boldsymbol{0}, \mathcal{K}(\boldsymbol{x}, \boldsymbol{x'}))$. Smoothness assumptions about f are encoded through the choice of kernel, $\mathcal{K}$; intuitively, actions with similar feature vectors have similar rewards. Then, given reward observations, a Gaussan process model posterior $P(\boldsymbol{f} \mid \mathcal{D})$ is fit to the data, $\mathcal{D}$. Srinivas et al. (2010) introduced the Gaussian Process Upper Confidence Bound (GP-UCB) algorithm for this setting, and proved the first sub-linear regret guarantee for Gaussian process bandits. Chowdhury and Gopalan (2017) proposed a confidence-based approach with tighter regret bounds than Srinivas et al. (2010), and additionally introduced and derived a regret bound for GP-TS, a posterior sampling Gaussian process bandit algorithm. Because Gaussian processes can be represented in an infinite-dimensional linear

space defined by the kernel function $\mathcal{K}$, the Gaussian process bandit setting can be viewed as a generalization of the generalized linear bandit setting discussed previously.

Other structural assumptions in the literature include bandits with Lipschitz-continous reward functions (Kleinberg, Slivkins, and Upfal, 2008) and convex reward functions (Saha and Tewari, 2011; Hazan and Levy, 2014).

Related Problem Settings

Beyond the standard bandit problem, a number of related problem settings have been studied. For instance, the active learning problem focuses exclusively on learning an accurate model rather than maximizing the cumulative utility of decision-making (Golovin and Krause, 2011; Sadigh et al., 2017; Houlsby et al., 2011; Brochu, de Freitas, and Ghosh, 2008). Similarly, Bayesian optimization (Brochu, Cora, and de Freitas, 2010; Hoffman, Brochu, and de Freitas, 2011) aims to identify the optimum of an expensive-to-evaluate function within a limited budget of queries, rather than minimize the regret:

$$\max_{\boldsymbol{x} \in \mathcal{A} \subset \mathbb{R}^d} f(\boldsymbol{x}).$$

Bayesian optimization is closely related to the Gaussian process bandit setting, as many Bayesian optimization algorithms use Gaussian process methods to model the objective function. Furthermore, approaches such as GP-UCB (Srinivas et al., 2010), which have bounded regret in the multi-armed bandit setting, are also applicable to Bayesian optimization; as Brochu, Cora, and de Freitas (2010) point out, a Bayesian optimization problem can be cast as a bandit problem, such that its objective function f becomes the bandit problem's reward function. Then, any bandit algorithm with sublinear regret, for instance GP-UCB, must converge to identifying the objective function's optimum. Other approaches to Bayesian optimization include maximizing the expected improvement over the highest objective value observed so far (Jones, Schonlau, and Welch, 1998), and entropy search (Hennig and Schuler, 2012), which selects points to minimize the entropy, or uncertainty, of f.

Another related problem is the adversarial bandit setting (Saha and Tewari, 2011; Hazan and Levy, 2014), in which rewards are selected by an adversary rather than stochastically from fixed probability distributions, as in the stochastic bandit setting.

Meanwhile, the agent can receive reward information in many different ways. While in the bandit setting, the agent only observes the reward corresponding to the specific action selected, in the case of expert advice, all actions' rewards are revealed in every

iteration (Cesa-Bianchi and Lugosi, 2006). In the semi-bandit feedback setting (Neu and Bartók, 2013; Russo and Van Roy, 2016), the algorithm selects a subset of the action space in each iteration, and the environment reveals the selected actions' total reward.

Environment feedback can also take various forms in addition to numerical rewards. In particular, the following subsection will discuss learning from pairwise preference feedback and review several other forms of feedback.

2.5 Dueling Bandits

In a number of real-world situations, absolute numerical feedback is unavailable. For instance, as discussed in Chapter 1, humans often cannot reliably specify numerical scores. Yet, in many cases, people *can* more accurately provide various forms of qualitative feedback, and in particular, much of this thesis focuses on online learning from pairwise preferences. In the bandit setting, preference-based learning is called the *dueling bandits problem* and was first formalized in Yue, Broder, et al. (2012). The dueling bandit setting is a sequential optimization problem, in which the learning agent selects actions and receives relative pairwise preference feedback between them. In other words, the agent poses queries of the form, "Do you prefer A or B?" The dueling bandits paradigm formalizes the problem of online regret minimization through preference-based feedback.

The dueling bandits setting is defined by a tuple, $(\mathcal{A}, \phi)$, in which $\mathcal{A}$ is the set of possible actions and ϕ is a function defining the feedback generation mechanism. The setting proceeds in a sequence of iterations or rounds. In each iteration i, the algorithm chooses a pair of actions $\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2} \in \mathcal{A}$ to duel or compare, and observes the outcome of that comparison ($\boldsymbol{x}_{i1}$ and $\boldsymbol{x}_{i2}$ can be identical).[3] The outcome of each duel between $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{A}$ represents a preference between them, and is an independent sample of a Bernoulli random variable. We denote a preference for action $\boldsymbol{x}$ over action $\boldsymbol{x}'$ as $\boldsymbol{x} \succ \boldsymbol{x}'$, and define the probability that $\boldsymbol{x}$ is preferred to $\boldsymbol{x}'$ as:

$$P(\boldsymbol{x} \succ \boldsymbol{x}') := \phi(\boldsymbol{x}, \boldsymbol{x}') + \frac{1}{2},$$

where $\phi(\boldsymbol{x}, \boldsymbol{x}') \in [-1/2, 1/2]$ denotes the stochastic preference between $\boldsymbol{x}$ and $\boldsymbol{x}'$, so that $P(\boldsymbol{x} \succ \boldsymbol{x}') > \frac{1}{2}$ if and only if $\phi(\boldsymbol{x}, \boldsymbol{x}') > 0$.

In each learning iteration i, the preference for the i^{th} query is denoted by $y_i := \mathbb{I}_{[\boldsymbol{x}_{i2} \succ \boldsymbol{x}_{i1}]} - \frac{1}{2} \in \left\{-\frac{1}{2}, \frac{1}{2}\right\}$, where $\mathbb{I}_{[\cdot]}$ denotes the indicator function, so that $P\left(y_i = \frac{1}{2}\right) =$

[3] The actions are bolded because they could be represented by d-dimensional vectors (i.e., $\mathcal{A} \subset \mathbb{R}^d$); however, such a representation is not required.

$1 - P\left(y_i = -\frac{1}{2}\right) = P(\boldsymbol{x}_{i2} \succ \boldsymbol{x}_{i1}) = \phi(\boldsymbol{x}_{i2}, \boldsymbol{x}_{i1}) + \frac{1}{2}$. The outcome y_i is always a binary preference (not a tie), but preference feedback need not be received for every preference query.

Bounding the regret of a bandit algorithm requires some assumptions on the tuple $(\mathcal{A}, \phi)$. For instance, if $\mathcal{A}$ is an infinite set and the outcomes associated with different actions are independent—that is, observing $\boldsymbol{x} \succ \boldsymbol{x}'$ yields no information about any action $\boldsymbol{x}'' \notin \{\boldsymbol{x}, \boldsymbol{x}'\}$—then, it would be impossible to achieve sublinear regret. Thus, either $\mathcal{A}$ must be finite, or there must be structural dependencies between outcomes associated with different actions.[4]

In the dueling bandits setting, the agent's decision-making quality can be quantified using a notion of cumulative regret in terms of the preference probabilities:

$$\mathrm{Reg}_{\phi}^{\mathrm{DB}}(T) = \sum_{i=1}^{T} \left[\phi(\boldsymbol{x}^*, \boldsymbol{x}_{i1}) + \phi(\boldsymbol{x}^*, \boldsymbol{x}_{i2})\right], \tag{2.15}$$

where $\boldsymbol{x}^*$ is the optimal action. Importantly, this regret definition assumes the existence of a Condorcet winner, that is, a unique action $\boldsymbol{x}^*$ superior to all other actions (Sui, Zoghi, et al., 2018); formally, $\boldsymbol{x}^* \in \mathcal{A}$ is a Condorcet winner if $P(\boldsymbol{x}^* \succ \boldsymbol{x}) > \frac{1}{2}$ for any $\boldsymbol{x}^* \neq \boldsymbol{x} \in \mathcal{A}$. When the learning algorithm has converged to the best action $\boldsymbol{x}^*$, then it can simply duel $\boldsymbol{x}^*$ against itself, incurring no additional regret. Intuitively, one can interpret Eq. (2.15) as a measure of how much the user(s) would have preferred the best action over the ones presented by the algorithm.

The Multi-Dueling Bandits Setting

The multi-dueling bandits setting (Brost et al., 2016; Sui, Zhuang, et al., 2017) generalizes the dueling bandits setting such that in each iteration, the algorithm can select more than 2 actions. More specifically, in the i^{th} iteration, the algorithm selects a set $S_i \subset \mathcal{A}$ of actions, and observes outcomes of duels between some pairs of the actions in S_i. These observed preferences could correspond to all action pairs in S_i or to any subset. The number of actions selected in each iteration is assumed to be a fixed constant, $m := |S_i|$. When $m = 2$, the problem reduces to the original dueling bandits setting. For any $m \geq 2$, the dueling bandits regret formulation in Eq. (2.15) extends to:

$$\mathrm{Reg}_{\phi}^{\mathrm{MDB}}(T) = \sum_{i=1}^{T} \sum_{\boldsymbol{x} \in S_i} \phi(\boldsymbol{x}^*, \boldsymbol{x}). \tag{2.16}$$

[4] Both of these conditions can also be true simultaneously.

The learning algorithm seeks action subsets S_i that minimize the cumulative regret in Eq. (2.16). Intuitively, the algorithm must explore the action space while aiming to minimize the number of times that it selects suboptimal actions. As the algorithm converges to the best action $\boldsymbol{x}^*$, then it increasingly chooses S_i to only contain $\boldsymbol{x}^*$, thus incurring no additional regret.

The multi-dueling setting could correspond to a number of different feedback collection mechanisms. For example, in some applications it may be viable to collect pairwise feedback between all pairs of actions in S_i. In other applications, it is more realistic to observe a subset of these preferences, for example the "winner" of S_i, that is, a preference set of the form, $\{\boldsymbol{x} \succ \boldsymbol{x}' \mid \boldsymbol{x}' \in S_i \setminus \boldsymbol{x}\}$ for some $\boldsymbol{x} \in S_i$.

Algorithms for the Dueling (and Multi-Dueling) Bandits Setting

A number of algorithms have been proposed for the preference-based bandit setting, many of which are equipped with regret bounds. Earlier work on dueling bandits considered an "explore-then-exploit" algorithmic structure; examples include the Interleaved Filter (Yue, Broder, et al., 2012), Beat-the-Mean (Yue and Joachims, 2011), and SAVAGE (Urvoy et al., 2013) algorithms. Confidence-based approaches include RUCB (Zoghi, Whiteson, Munos, et al., 2014), MergeRUCB (Zoghi, Whiteson, and Rijke, 2015), and CCB (Zoghi, Karnin, et al., 2015). The RMED algorithm (Komiyama et al., 2015), meanwhile, defines Bernoulli distributions whose parameters are actions' probabilities of being preferred to one another. The approach calculates Kullback-Leibler divergences between these distributions, and uses them to estimate each action's probability of being the Condorcet winner and to decide whether actions have been sufficiently explored. RMED is optimal in the sense that when a Condorcet winner exists, the constant factor in its regret matches the regret lower bound for the dueling bandits problem.

Ailon, Karnin, and Joachims (2014) propose several strategies for reducing the dueling bandits problem to a standard multi-armed bandit problem with numerical feedback. In particular, this work introduces the Sparring method, in which two multi-armed bandit algorithms are trained in parallel. These two algorithms could be instances of any standard multi-armed bandit algorithm, for instance RUCB or RMED. In every iteration, each of the two algorithms selects an action; let $\boldsymbol{x}_{i1}$ and $\boldsymbol{x}_{i2}$ be the actions selected by algorithms 1 and 2 in iteration i, respectively. These two actions are dueled against each other. If $\boldsymbol{x}_1$ wins, then algorithm 1 receives a reward of 1 and algorithm 2 receives a reward of 0; conversely, if $\boldsymbol{x}_2$ wins, then

Algorithm 3 IndependentSelfSparring (Sui, Zhuang, et al., 2017)

Input: m = the number of actions taken in each iteration, η = the learning rate
For each action $a = 1, 2, \cdots, A$, set $W_a = 0$, $F_a = 0$.
for $i = 1, 2, \ldots$ **do**
    for $j = 1, \ldots, m$ **do**
        For each action $a = 1, 2, \cdots, A$, sample $\theta_a \sim \text{Beta}(W_a + 1, F_a + 1)$
        Select $a_j(i) := \text{argmax}_a \, \theta_a$
    end for
    Execute m actions: $\{a_1(i), \ldots, a_m(i)\}$
    Observe pairwise feedback matrix $R \in \{0, 1, \emptyset\}^{m \times m}$
    for $j, l = 1, \ldots, m$ **do**
        if $r_{jl} \neq \emptyset$ **then**
            $W_{a_j(i)} \leftarrow W_{a_j(i)} + \eta \cdot r_{jl}$, $F_{a_j(i)} \leftarrow F_{a_j(i)} + \eta(1 - r_{jl})$
        end if
    end for
end for

algorithm 1 receives a reward of 0 and algorithm 2 receives a reward of 1.

There are also a number of posterior sampling-based strategies for the dueling bandit problem. The Relative Confidence Sampling algorithm, for instance, on each iteration selects one action via posterior sampling and another via an upper confidence bound strategy (Zoghi, Whiteson, De Rijke, et al., 2014). Meanwhile, the Double Thompson Sampling algorithm (Wu and Liu, 2016) selects both actions in each iteration via posterior sampling; however, each action is selected from a restricted set. The first action is selected among those thought to be relatively well performing, while the second is chosen among actions with less certain priors.

In particular, SelfSparring (Sui, Zhuang, et al., 2017) is a state-of-the-art posterior sampling framework for preference-based learning in the multi-dueling bandits setting, inspired by Sparring (Ailon, Karnin, and Joachims, 2014). SelfSparring maintains a distribution over the probability of selecting each action, and updates it during each iteration based on the observed preferences. Actions are selected by drawing independent samples from this probability distribution, and for each sample, selecting the action with the highest sampled reward. Intuitively, a flatter sampling distribution results in more exploration, while more peaked sampling distributions reflect certainty about the optimal action and yield more exploitation.

Algorithm 3 describes SelfSparring for the case of a finite action set ($|\mathcal{A}| = A$) with independent actions, that is, observing a preference $\boldsymbol{x} \succ \boldsymbol{x}'$ does not yield information about the performance of any action $\boldsymbol{x}'' \notin \{\boldsymbol{x}, \boldsymbol{x}'\}$. SelfSparring maintains a Beta

posterior for each action, and updates it similarly to the Bernoulli bandit posterior sampling algorithm (Algorithm 1). For each action a, the variables W_a and F_a, respectively, record the number of times that the action wins and loses against any other action (possibly scaled by a learning rate, η). Actions are then selected by drawing a sample for each action $a \in \mathcal{A}$, $\theta_a \sim \text{Beta}(W_a + 1, F_a + 1)$, and choosing the action for which θ_a is maximized. Sui, Zhuang, et al. (2017) derive a no-regret guarantee for Algorithm 3 when the actions have a total ordering with respect to the preferences.

Additionally, Sui, Zhuang, et al. (2017) propose a kernel-based variant of Self-Sparring called KernelSelfSparring, which leverages Gaussian process regression to model dependencies among rewards corresponding to different actions. Other dueling bandit approaches which handle structured action spaces include the Dueling Bandit Gradient Descent algorithm for convex utility functions over continuous action spaces (Yue and Joachims, 2009); the Correlational Dueling Bandits algorithm, in which a large number of actions have a low-dimensional correlative structure based on a given similarity function (Sui, Yue, and Burdick, 2017); and the StageOpt algorithm for safe learning with Gaussian process dueling bandits (Sui, Zhuang, et al., 2018).

The multi-dueling bandits setting is also considered in Brost et al. (2016), which proposes a confidence interval strategy for estimating preference probabilities.

Related Problem Settings

Aside from pairwise preferences, a number of other forms of qualitative reward feedback have also been studied. For instance, in the coactive feedback problem (Shivaswamy and Joachims, 2015; Shivaswamy and Joachims, 2012; Raman et al., 2013), the agent selects an action $\boldsymbol{a}_i$ in the i^{th} iteration, and the environment reveals an action $\boldsymbol{a}'_i$ which is *preferred* to $\boldsymbol{a}_i$. With ordinal feedback, meanwhile, the user assigns each trial to one of a discrete set of ordered categories (Chu and Ghahramani, 2005a; Herbrich, Graepel, and Obermayer, 1999). In the "learning to rank" problem, an agent learns a ranking function for information retrieval using feedback such as users' clickthrough data (Burges, Shaked, et al., 2005; Radlinski and Joachims, 2005; Burges, Ragno, and Le, 2007; Liu, 2009; Schuth, Sietsma, et al., 2014; Schuth, Oosterhuis, et al., 2016; Yue, Finley, et al., 2007).

2.6 Episodic Reinforcement Learning

In the episodic reinforcement learning (RL) problem setting (Sutton and Barto, 2018; Dann and Brunskill, 2015; Osband, Russo, and Van Roy, 2013), an agent interacts with an environment in a series of episodes, with the goal of maximizing its cumulative reward over time. The RL setting differs from the bandit setting in that the reward distributions depend upon the environment's state, which changes stochastically in each time step as a function of the previous state and the selected action.

The tabular RL problem is formalized as a Markov decision process (MDP), defined via the tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, r, p, p_0, h)$, where the state space $\mathcal{S}$ and action space $\mathcal{A}$ are finite sets with cardinalities $S = |\mathcal{S}|$ and $A = |\mathcal{A}|$. At each time step t, the agent observes the environment's current state s_t, takes an action a_t, and then observes a reward r_t and the subsequent state, s_{t+1}. The episode's initial state is sampled from the distribution p_0 such that $s_1 \sim p_0$, while p defines the transition dynamics: $s_{t+1} \sim p(\cdot|s_t, a_t)$. Because s_{t+1} does not depend on any states or actions prior to time t, the environment is Markovian. Finally, r defines the expected reward given a state and action: $\mathbb{E}[r_t] = r(s_t, a_t)$.[5]

The agent interacts with the environment through a sequence of episodes of length h, where h is known as the episode horizon time. Each episode is a trajectory of states, actions, and rewards of the form:

$$\tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \ldots, s_h, a_h, s_{h+1}\}.$$

Many episodic RL algorithms select a single action selection policy per episode. In this context, a *policy* π is a possibly-stochastic mapping from states and time indices (within an episode) to actions: $\pi : \mathcal{S} \times \{1, \ldots, h\} \longrightarrow \mathcal{A}$. In each iteration i, the agent selects a policy π_i and executes it to observe a trajectory, τ_i.

Given a policy π, the *value function* is defined as the expected total reward accrued when starting in state s at step j, and following π for the rest of the episode:

$$V_{\pi,j}(s) = \mathbb{E}\left[\sum_{t=j}^{h} r(s_t, \pi(s_t, t)) \,\middle|\, s_j = s\right]. \tag{2.17}$$

[5] Similarly, r could be defined as a function of just the current state, $\mathbb{E}[r_t] = r(s_t)$, or of the state, action, and next state, $\mathbb{E}[r_t] = r(s_t, a_t, s_{t+1})$.

Similarly, the action-value function is the expected total reward when starting in state s at step j, taking action a, and then following π until the episode's completion:

$$Q_{\pi,j}(s,a) = r(s,a) + \mathbb{E}\left[\sum_{t=j+1}^{h} r(s_t, \pi(s_t,t)) \,\middle|\, s_j = s, a_j = a\right].$$

An optimal policy π^* is one which maximizes the expected total reward over the episode:

$$\pi^* = \sum_{s\in\mathcal{S}} p_0(s) \sup_{\pi} V_{\pi,1}(s),$$

where π is maximized over the set of all possible policies.

The agent's learning goal is to minimize the regret, which is defined as the gap between the total reward obtained when repeatedly executing π^* and the total reward accrued by the learning algorithm:

$$\mathbb{E}[\mathrm{Reg}(T)] = \mathbb{E}\left\{\sum_{i=1}^{\lceil T/h\rceil} \sum_{s\in\mathcal{S}} p_0(s)\left[V_{\pi^*,1}(s) - V_{\pi_i,1}(s)\right]\right\},$$

where T is the total number of time-steps of agent-environment interaction (i.e., the number of episodes multiplied by h), and where the expected value is taken with respect to randomness in the policy selection algorithm and randomness in the rewards (which affect future policy selection). The Bayesian regret $\mathrm{BayesReg}(T)$ is defined similarly, but the expected value is additionally taken with respect to randomness in the MDP parameters p, p_0, and r, which are assumed to be sampled from a prior over MDP environments.

Remark 1. *Note that the episodic RL setting stands in contrast to the infinite-horizon RL problem. The latter is also defined as a Markov decision process, except without the time horizon h (since the interaction consists of a single, infinite-length trajectory). The MDP is defined as* $\mathcal{M} = (\mathcal{S}, \mathcal{A}, r, p, p_0, \gamma)$*, where* $\gamma \in (0,1)$ *is the discount factor used to quantify the total reward. Unlike in the episodic setting, the value does not depend on the initial state, the policy is time-independent, and the discounted value function (Sutton and Barto, 2018) is also time-step-independent and is defined as:*

$$V_\pi(s) = \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^{t-1} r(s_t, \pi(s_t))\right].$$

While a number of infinite-horizon RL studies derive performance guarantees for the discounted reward setting (Kearns and Singh, 2002; Lattimore and Hutter, 2012),

others consider the average reward setting (Jaksch, Ortner, and Auer, 2010; Bartlett and Tewari, 2012; Agrawal and Jia, 2017), which removes the factor γ and defines $V_\pi(s)$ as follows:

$$V_\pi(s) = \lim_{T\to\infty} \frac{1}{T}\mathbb{E}\left[\sum_{t=1}^{\infty} r(s_t, \pi(s_t))\right].$$

Using Value Iteration to Compute Optimal Policies

When the transition dynamics p and rewards r are known, then it is possible to exactly calculate the optimal, reward-maximizing policy via a dynamic programming method called finite-horizon value iteration (Sutton and Barto, 2018). In practice, the rewards and dynamics are typically unknown and must be learned, so that value iteration is insufficient to obtain the best policy. As will be discussed in the next subsection, however, value iteration can still be a useful component within algorithms that learn the optimal policy.

The value iteration procedure operates as follows. Firstly, the value function $V_{\pi,t}(s)$ is defined as in Eq. (2.17) for each state and time-step in the episode. Because the episode stops accruing reward at its completion, one can set $V_{\pi,h+1}(s) := 0$ for each $s \in \mathcal{S}$. Then, successively for each $t \in \{h, h-1, \ldots, 1\}$, $\pi(s,t)$ is set deterministically to maximize the total reward starting from time t, and the Bellman equation is used to calculate $V_{\pi,t}(s)$ given p and r. Formally, for each $t \in \{h, h-1, \ldots, 1\}$:

$$\pi(s,t) = \operatorname*{argmax}_{a\in\mathcal{A}} \left[r(s,a) + \sum_{s'\in\mathcal{S}} P(s_{t+1} = s' \mid s_t = s, a_t = a) V_{\pi,t+1}(s')\right] \text{ for all } s \in \mathcal{S},$$

$$V_{\pi,t}(s) = \sum_{a\in\mathcal{A}} \mathbb{I}_{[\pi(s,t)=a]} \left[r(s,a) + \sum_{s'\in\mathcal{S}} P(s_{t+1} = s' \mid s_t = s, a_t = a) V_{\pi,t+1}(s')\right] \text{ for all } s \in \mathcal{S}.$$

Note that value iteration results in only deterministic policies, of which there are finitely-many (more precisely, there are A^{Sh}).

Reinforcement Learning Algorithms

RL algorithms can be broadly divided into two categories: model-based and model-free approaches. Model-based algorithms explicitly construct a model of the environment as they interact with it, for instance storing information about state transitions and rewards. Model-free algorithms, in contrast, do not attempt to directly learn the MDP's parameters, but rather only model value function information.

Both model-free and model-based RL algorithms have leveraged deep learning to successfully execute tasks in such domains as Atari games (Mnih, Kavukcuoglu, Silver, Graves, et al., 2013; Mnih, Kavukcuoglu, Silver, Rusu, et al., 2015; Kaiser et al., 2019), simulated physics-based tasks in the Mujoco environment (Lillicrap et al., 2015; Chua et al., 2018), and robotics (Ebert et al., 2018; Ruan et al., 2019). Model-based RL algorithms are often more sample-efficient than model-free algorithms (Kaiser et al., 2019; Ebert et al., 2018; Plaat, Kosters, and Preuss, 2020), as they explicitly build up a dynamics model of the environment.

There is also significant work on analyzing algorithmic performance in the tabular RL setting, in which there are finite numbers of states and actions. In addition to regret bounds, common performance metrics include sample complexity guarantees, which upper-bound the number of steps for which performance is not near-optimal. In particular, a PAC sample complexity guarantee states that with probability at least $1 - \delta$, in all but an upper-bounded number of time-steps, the values of the selected policies are within ϵ of the optimal policy's value.

There has been extensive work on confidence and posterior sampling approaches to RL (among other types of algorithms). These are both model-based methods, as they require learning estimates of the state transition probabilities and rewards.

Confidence-based algorithms for the episodic RL setting include the Bayesian Exploration Bonus algorithm (Kolter and Ng, 2009), which learns transition and reward information to estimate the action-value function $Q(s, a)$ at each state-action pair. Actions are chosen greedily with respect to the sum of the Q-function and an exploration bonus given to less-explored state-action pairs. The authors prove a PAC sample complexity guarantee, specifically that with probability at least $1 - \delta$, the policy is ϵ-close to optimal for all but (at most) $O\left(\frac{SAh^6}{\epsilon^2} \log\left(\frac{SA}{\delta}\right)\right)$ time-steps. Meanwhile, Dann and Brunskill (2015) dramatically improve upon this h^6 dependence in the analysis of their upper-confidence fixed-horizon RL algorithm (UCFH). UCFH selects policies by optimistically estimating each policy's value, or total reward, and maximizing the expected total reward over an MDP confidence set containing the true MDP with high probability. This confidence region is constructed by defining a confidence interval around the empirical mean of each reward and transition dynamics parameter in the MDP. The authors prove that with probability at least $1 - \delta$, UCFH selects policies that are ϵ-close to optimal for all but (at most) $\tilde{O}\left(\frac{SAh^2C}{\epsilon^2} \log\left(\frac{1}{\delta}\right)\right)$ steps, where the $\tilde{O}$-notation hides logarithmic factors and $C \leq S$ upper-bounds the number of possible successor states from any state in the MDP.

Algorithm 4 Posterior Sampling for Reinforcement Learning (PSRL) (Osband, Russo, and Van Roy, 2013)

Initialize prior for f_p ▷ Initialize transition dynamics model
Initialize prior for f_r ▷ Initialize reward model
for $i = 1, 2, \ldots$ **do**
    Sample $\tilde{\boldsymbol{p}} \sim f_p(\cdot)$ ▷ Sample transition dynamics from posterior
    Sample $\tilde{\boldsymbol{r}} \sim f_r(\cdot)$ ▷ Sample rewards from posterior
    $\pi_i \longleftarrow$ optimal policy given $\tilde{\boldsymbol{p}}$ and $\tilde{\boldsymbol{r}}$ ▷ Value iteration
    for $t = 1, \ldots, h$ **do** ▷ Execute policy π_i to roll out a trajectory
        Observe state s_t and take action $a_t = \pi_i(s_t, t)$
        Observe subsequent state s_{t+1} and reward r_t
    end for
    Update posteriors f_p, f_r given observed transitions and rewards
end for

Posterior sampling approaches to finite-horizon RL include the posterior sampling for reinforcement learing (PSRL) algorithm (Osband, Russo, and Van Roy, 2013), outlined in Algorithm 4. PSRL learns Bayesian posteriors over both the MDP's transition dynamics and rewards. In each learning iteration, the algorithm samples both dynamics and rewards from their posteriors, and computes the optimal policy for this sampled MDP via value iteration. The resultant policy is executed to get a roll-out trajectory, which is used to update the dynamics and reward posteriors. Osband, Russo, and Van Roy (2013) derive a Bayesian regret bound of $O(hS\sqrt{AT\log(SAT)})$ for PSRL after T steps of agent-environment interaction. Notably, computational results in Osband, Russo, and Van Roy (2013) demonstrate that posterior sampling outperforms confidence-based approaches, and Osband and Van Roy (2017) theorize further about the reasons behind this phenomenon.

Beyond finite-horizon RL, upper confidence bound methods (Jaksch, Ortner, and Auer, 2010; Bartlett and Tewari, 2012) and posterior sampling techniques (Agrawal and Jia, 2017; Ouyang et al., 2017; Gopalan and Mannor, 2015) have also been demonstrated to perform well in the infinite-horizon RL domain.

Meanwhile, randomized least-squares value iteration (RLSVI), proposed in Osband, Van Roy, and Wen (2016), is a model-free approach inspired by posterior sampling. RLSVI maintains a distribution over plausible linearly-parameterized value functions, and explores by randomly sampling from these distributions and selecting the actions which are optimal under these value functions. The authors prove an expected regret bound of $\tilde{O}(\sqrt{h^3SAT})$.

Finally, information-directed sampling methods select actions by estimating the in-

formation ratio proposed in Russo and Van Roy (2016) and given by Eq. (2.12), which can be adapted from the bandit setting to RL. As in the bandit setting, the information ratio can be intuitively understood as the squared expected instantaneous regret divided by the current iteration's information gain. Information-directed sampling, first developed in Russo and Van Roy (2014a) for the bandit problem, selects the actions that minimize the information ratio.

Zanette and Sarkar (2017) propose both exact and approximate methods for information-directed RL. Their exact method identifies the true information-ratio-minimizing policy, and has a Bayesian regret bound of $S\sqrt{\frac{1}{2}AT \log A}$. This result is obtained by assuming known transition dynamics and applying a result from Russo and Van Roy (2016) for linear bandits. The method, however, is computationally intractable for all but the smallest MDPs; thus, Zanette and Sarkar (2017) also propose several approximations to the algorithm, for instance only optimizing the information ratio over a reduced subset of policies in each episode, where the policies considered are obtained using posterior sampling. In experiments, the approximate information-directed sampling algorithm significantly outperforms PSRL.

Lastly, Nikolov et al. (2018) extend frequentist information-directed exploration to the deep RL setting, and estimate the information ratio via a Bootstrapped DQN[6] (Osband, Blundell, et al., 2016) that learns the action-value function. The resulting algorithm uses information-theoretic methods to account for heteroscedastic noise—i.e., that the distribution of the reward noise varies depending on the state and action—and outperforms state-of-the-art algorithms on Atari games.

Preference-Based Reinforcement Learning

Existing work on preference-based RL has shown successful performance in a number of applications, including Atari games and the Mujoco environment (Christiano et al., 2017; Ibarz et al., 2018), autonomous driving (Sadigh et al., 2017; Bıyık, Huynh, et al., 2020), and robot control (Kupcsik, Hsu, and Lee, 2018; Akrour, Schoenauer, Sebag, and Souplet, 2014). Yet, the preference-based RL literature still lacks theoretical guarantees.

Much of the existing work in preference-based RL handles a distinct setting from that considered in this thesis. While the work found in this thesis is concerned with online regret minimization, several existing algorithms instead minimize the number of preference queries (Christiano et al., 2017; Wirth, Fürnkranz, and Neumann, 2016).

[6]Deep Q-Network.

Such algorithms, for instance those which apply deep learning, typically assume that many simulations can be cheaply run between preference queries. In contrast, the present setting assumes that experimentation is as expensive as preference elicitation; this includes any situation in which a simulator is unavailable.

Existing approaches for trajectory-level preference-based RL may be broadly divided into three categories (Wirth, 2017): a) directly optimizing policy parameters (Wilson, Fern, and Tadepalli, 2012; Busa-Fekete et al., 2014; Kupcsik, Hsu, and Lee, 2018); b) modeling preferences between actions in each state (Fürnkranz, Hüllermeier, et al., 2012); and c) learning a utility function to characterize the rewards, returns, or values of state-action pairs (Wirth and Fürnkranz, 2013b; Wirth and Fürnkranz, 2013a; Akrour, Schoenauer, and Sebag, 2012; Wirth, Fürnkranz, and Neumann, 2016; Christiano et al., 2017). In c), the utility is often modeled as linear in features of the trajectories. If those features are defined in terms of visitations to each state-action pair, then utility directly corresponds to the total (undiscounted) reward. Notably, under a), Wilson, Fern, and Tadepalli (2012) learn a Bayesian model over policy parameters, and sample from its posterior to inform action selection. This is a model-free posterior sampling-inspired method; however, compared to utility-based approaches, policy search methods typically require either more samples or expert knowledge to craft the policy parameters (Wirth, Akrour, et al., 2017; Kupcsik, Hsu, and Lee, 2018).

The work presented in this thesis adopts the third of these paradigms: preference-based RL with underlying utility functions. By inferring state-action rewards from preference feedback, one can derive relatively-interpretable reward models and employ such methods as value iteration. In addition, utility-based approaches may be more sample efficient compared to policy search and preference relation methods (Wirth, 2017), as they extract more information from each observation.

Chapter 3

THE PREFERENCE-BASED GENERALIZED LINEAR BANDIT AND REINFORCEMENT LEARNING PROBLEM SETTINGS

This chapter describes the problem formulations and key assumptions that are considered in Chapter 4, which introduces the Dueling Posterior Sampling framework for preference-based learning. Below, Section 3.1 describes the generalized linear dueling bandit setting, while Section 3.2 defines the tabular preference-based RL setting. These interrelated problem settings formalize the challenge of learning from preference feedback when the preferences are governed by underlying utilities of the bandit actions and RL trajectories, and when these utilities are linear in features of the actions and trajectories. Subsequently, Chapter 5 further extends the problem settings presented in this chapter by allowing mixed-initiative user feedback and considering kernelized utility functions.

3.1 The Generalized Linear Dueling Bandit Problem Setting

The generalized linear dueling bandit setting defines a low-dimensional structure relating actions to preferences. This framework allows for large or infinite action spaces, which would be infeasible for a learning algorithm that models each action's reward independently.

The setting is defined by a tuple, $(\mathcal{A}, \phi)$, in which $\mathcal{A} \subset \mathbb{R}^d$ is a compact set of possible actions and ϕ is a function defining the preference feedback generation mechanism. In each learning iteration i, the algorithm chooses a pair of actions $\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2} \in \mathcal{A}$ to duel or compare, and receives a preference between them. For two actions $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{A}$, the function ϕ defines the probability that $\boldsymbol{x}$ is preferred to $\boldsymbol{x}'$:

$$P(\boldsymbol{x} \succ \boldsymbol{x}') := \phi(\boldsymbol{x}, \boldsymbol{x}') + \frac{1}{2},$$

where $\phi(\boldsymbol{x}, \boldsymbol{x}') \in [-1/2, 1/2]$ denotes the stochastic preference between $\boldsymbol{x}$ and $\boldsymbol{x}'$. The outcome of the i^{th} preference query is denoted by $y_i := \mathbb{I}_{[\boldsymbol{x}_{i2} \succ \boldsymbol{x}_{i1}]} - \frac{1}{2} \in \left\{-\frac{1}{2}, \frac{1}{2}\right\}$, so that $P\left(y_i = \frac{1}{2}\right) = 1 - P\left(y_i = -\frac{1}{2}\right) = \phi(\boldsymbol{x}_{i2}, \boldsymbol{x}_{i1}) + \frac{1}{2}$.

Preferences are assumed to be governed by a low-dimensional structure, which depends upon a model parameter $\overline{\boldsymbol{r}} \in \Theta \subset \mathbb{R}^d$ for a compact set Θ. This structure is defined by three main assumptions. Firstly, to guarantee that the rewards are

learnable, one must assume that they belong to a compact set. This is satisfied by assuming that the reward vector $\overline{\boldsymbol{r}}$ has bounded magnitude:

Assumption 1. *For some known $S_r < \infty$, $||\overline{\boldsymbol{r}}||_2 \leq S_r$.*

Secondly, each action has an associated utility:

Assumption 2. *Each action $\boldsymbol{x} \in \mathcal{A}$ has a utility to the user, which reflects the value that the user places upon $\boldsymbol{x}$. This utility, denoted by $u(\boldsymbol{x})$, is bounded and given by $u(\boldsymbol{x}) := \overline{\boldsymbol{r}}^T \boldsymbol{x}$.*

The third assumption defines the preference relation function ϕ in terms of the utilities u.

Assumption 3. *The user's preference between any two actions $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{A}$ is given by $P(\boldsymbol{x} \succ \boldsymbol{x}') = \phi(\boldsymbol{x}, \boldsymbol{x}') + \frac{1}{2}$, where $\phi(\boldsymbol{x}, \boldsymbol{x}')$ is defined in terms of a* link function *$g : \mathbb{R} \longrightarrow \left[-\frac{1}{2}, \frac{1}{2}\right]$: $\phi(\boldsymbol{x}, \boldsymbol{x}') := g(u(\boldsymbol{x}) - u(\boldsymbol{x}')) = g(\overline{\boldsymbol{r}}^T(\boldsymbol{x} - \boldsymbol{x}'))$. The link function g must a) be non-decreasing, and b) satisfy $g(x) = -g(-x)$ to ensure that $P(\boldsymbol{x} \succ \boldsymbol{x}') = 1 - P(\boldsymbol{x}' \succ \boldsymbol{x})$. Note that if $u(\boldsymbol{x}) = u(\boldsymbol{x}')$, then $P(\boldsymbol{x} \succ \boldsymbol{x}') = \frac{1}{2}$, and that $P(\boldsymbol{x} \succ \boldsymbol{x}') > \frac{1}{2} \iff g(u(\boldsymbol{x}) - u(\boldsymbol{x}')) > 0 \iff u(\boldsymbol{x}) > u(\boldsymbol{x}')$.*

Assumptions 2 and 3 are likely to hold in many real-world applications. The utility could reflect a human user's satisfaction with the algorithm's performance, and it is plausible that users would give more accurate preferences between actions that have more disparate utilities. Furthermore, linear utility models have been applied in a number of domains, including web search (Shivaswamy and Joachims, 2015), movie recommendation (Shivaswamy and Joachims, 2015), autonomous driving (Sadigh et al., 2017; Bıyık, Palan, et al., 2020), and robotics (Bıyık, Palan, et al., 2020).

There are many alternatives for the link function g. For noiseless preferences,

$$g_{\text{ideal}}(x) := \mathbb{I}_{[x>0]} - \frac{1}{2}, \tag{3.1}$$

while the linear link function (Ailon, Karnin, and Joachims, 2014) is given by,

$$g_{\text{lin}}(x) := \frac{x}{c}, \text{ for } c > 0 \text{ and } x \in \left[-\frac{c}{2}, \frac{c}{2}\right]. \tag{3.2}$$

Under g_{lin}, $P(\boldsymbol{x}_{i2} \succ \boldsymbol{x}_{i1}) - \frac{1}{2} = \frac{\overline{\boldsymbol{r}}^T(\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1})}{c}$. Without loss of generality, c can be set to 1 by subsuming it into $\overline{\boldsymbol{r}}$. Alternatively, the logistic or Bradley-Terry link function is defined as,

$$g_{\log}(x) := [1 + \exp(-x/c)]^{-1} - \frac{1}{2}, \text{ with "temperature" } c \in (0, \infty). \tag{3.3}$$

The noise in the i^{th} preference is defined as η_i, so that $y_i = \mathbb{E}[y_i] + \eta_i = P(\boldsymbol{x}_{i2} \succ \boldsymbol{x}_{i1}) - \frac{1}{2} + \eta_i$. Under the linear link function, for instance, $\eta_i = y_i - \overline{\boldsymbol{r}}^T(\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1})$. Lastly, define $\boldsymbol{x}_i := \boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}$ as the observed feature difference vector in iteration i.

In the utility-based setting, regret can be defined in two ways. The first approach, which is typically used in the dueling bandits setting (Yue, Broder, et al., 2012; Sui, Zhuang, et al., 2017), was previously presented in Eq. (2.15):

$$\text{Reg}_{\phi}^{\text{DB}}(T) = \sum_{i=1}^{T} \left[\phi(\boldsymbol{x}^*, \boldsymbol{x}_{i1}) + \phi(\boldsymbol{x}^*, \boldsymbol{x}_{i2})\right],$$

where $\boldsymbol{x}^* \in \mathcal{A}$ is the optimal action given $\overline{\boldsymbol{r}}$, that is, $\boldsymbol{x}^* := \text{argmax}_{\boldsymbol{x}\in\mathcal{A}} u(\boldsymbol{x}) = \text{argmax}_{\boldsymbol{x}\in\mathcal{A}} \boldsymbol{x}^T\overline{\boldsymbol{r}}$. In the second approach, regret can also be defined with respect to the underlying utilities, u, as the total loss in utility during the learning process:

$$\text{Reg}_{u}^{\text{DB}}(T) = \sum_{i=1}^{T} \left[2u(\boldsymbol{x}^*) - u(\boldsymbol{x}_{i1}) - u(\boldsymbol{x}_{i2})\right] = \sum_{i=1}^{T} \left[2\overline{\boldsymbol{r}}^T(\boldsymbol{x}^* - \boldsymbol{x}_{i1} - \boldsymbol{x}_{i2})\right]. \quad (3.4)$$

Remark 2. *Note that unlike* $\text{Reg}_{u}^{DB}(T)$*, the non-utility-based regret* $\text{Reg}_{\phi}^{\text{DB}}(T)$ *is well-defined even when utilities do not exist. When the utilities exist and are defined as above, however,* $\text{Reg}_{\phi}^{\text{DB}}(T)$ *and* $\text{Reg}_{u}^{\text{DB}}(T)$ *can be bounded in terms of each other via the constants* $g'_{\min}, g'_{\max} \in (0, \infty)$*, which are, respectively, the minimum and maximum derivatives of the link function* g *over its possible inputs:*

$$g'_{\max} := \sup_{x\in[-u',u']} g'(x) \quad \textit{and} \quad g'_{\min} := \inf_{x\in[-u',u']} g'(x),$$

where $u' := \sup_{\boldsymbol{x},\boldsymbol{x}'\in\mathcal{A}}[u(\boldsymbol{x}) - u(\boldsymbol{x}')]$*. These facts can be derived as follows:*

$$\begin{aligned}
\text{Reg}_{\phi}^{\text{DB}}(T) &= \sum_{i=1}^{T} \left[\phi(\boldsymbol{x}^*, \boldsymbol{x}_{i1}) + \phi(\boldsymbol{x}^*, \boldsymbol{x}_{i2})\right] = \sum_{i=1}^{T} \left[g(u(\boldsymbol{x}^*) - u(\boldsymbol{x}_{i1})) + g(u(\boldsymbol{x}^*) - u(\boldsymbol{x}_{i2}))\right] \\
&\overset{(a)}{\leq} g'_{\max} \sum_{i=1}^{T} [u(\boldsymbol{x}^*) - u(\boldsymbol{x}_{i1}) + u(\boldsymbol{x}^*) - u(\boldsymbol{x}_{i2})] = g'_{\max}\text{Reg}_{u}^{\text{DB}}(T), \textit{ and similarly,} \\
\text{Reg}_{\phi}^{\text{DB}}(T) &= \sum_{i=1}^{T} \left[\phi(\boldsymbol{x}^*, \boldsymbol{x}_{i1}) + \phi(\boldsymbol{x}^*, \boldsymbol{x}_{i2})\right] = \sum_{i=1}^{T} \left[g(u(\boldsymbol{x}^*) - u(\boldsymbol{x}_{i1})) + g(u(\boldsymbol{x}^*) - u(\boldsymbol{x}_{i2}))\right] \\
&\overset{(b)}{\geq} g'_{\min} \sum_{i=1}^{T} [u(\boldsymbol{x}^*) - u(\boldsymbol{x}_{i1}) + u(\boldsymbol{x}^*) - u(\boldsymbol{x}_{i2})] = g'_{\min}\text{Reg}_{u}^{\text{DB}}(T),
\end{aligned}$$

where (a) and (b) hold because $u(\boldsymbol{x}^*) - u(\boldsymbol{x}) \geq 0$ *for all* $\boldsymbol{x} \in \mathcal{A}$*.*

Remark 3. *The generalized linear dueling bandits setting straightforwardly extends to the multi-dueling bandits paradigm discussed in Section 2.5, in which a set S_i of $m \geq 2$ actions are selected in each learning iteration. Recall that the regret is defined as in Eq.* (2.15)*:*

$$\mathrm{Reg}_{\phi}^{MDB}(T) = \sum_{i=1}^{T} \sum_{\boldsymbol{x} \in S_i} \phi(\boldsymbol{x}^*, \boldsymbol{x}).$$

When the actions have associated utilities, the regret can also be defined with respect to the loss in accumulated utility:

$$\mathrm{Reg}_{u}^{MDB}(T) = \sum_{i=1}^{T} \sum_{\boldsymbol{x} \in S_i} [u(\boldsymbol{x}^*) - u(\boldsymbol{x})] = \sum_{i=1}^{T} \left[mu(\boldsymbol{x}^*) - \sum_{\boldsymbol{x} \in S_i} u(\boldsymbol{x}) \right]. \tag{3.5}$$

Just as in the dueling bandit setting, if utilities exist, then $g'_{\min}\mathrm{Reg}_{u}^{MDB}(T) \leq \mathrm{Reg}_{\phi}^{MDB}(T) \leq g'_{\max}\mathrm{Reg}_{u}^{MDB}(T)$ *(this can be shown as in Remark 2).*

Finally, the Bayesian regret is defined as the expected regret, where the expectation is taken over the algorithm's action selection strategy, randomness in the outcomes, and randomness due to the prior over $\overline{\boldsymbol{r}}$:

$$\mathrm{BayesReg}_{\phi}^{\mathrm{DB}}(T) = \mathbb{E}\left[\sum_{i=1}^{T} [\phi(\boldsymbol{x}^*, \boldsymbol{x}_{i1}) + \phi(\boldsymbol{x}^*, \boldsymbol{x}_{i2})] \right],$$

and similarly,

$$\mathrm{BayesReg}_{u}^{\mathrm{DB}}(T) = \mathbb{E}\left[\sum_{i=1}^{T} \left[2\overline{\boldsymbol{r}}^T (\boldsymbol{x}^* - \boldsymbol{x}_{i1} - \boldsymbol{x}_{i2}) \right] \right].$$

3.2 The Preference-Based Reinforcement Learning Problem Setting

In this setting, the agent interacts with fixed-horizon Markov Decision Processes (MDPs), in which rewards are replaced by preferences over trajectories. This class of MDPs can be represented as a tuple, $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \phi, p, p_0, h)$, where the state space $\mathcal{S}$ and action space $\mathcal{A}$ are finite sets with cardinalities S and A, respectively. The agent episodically interacts with the environment in length-h roll-out trajectories of the form $\tau = \{s_1, a_1, s_2, a_2, \ldots, s_h, a_h, s_{h+1}\}$. In the i^{th} iteration, the agent executes two trajectory roll-outs τ_{i1} and τ_{i2} and observes a preference between them; similarly to the preference-based bandit setting, the notation $\tau \succ \tau'$ indicates a preference for trajectory τ over τ'. The initial state is sampled from p_0, while p defines the

transition dynamics: $s_{t+1} \sim p(\cdot|s_t, a_t)$. Finally, the function ϕ captures the preference feedback generation mechanism: $\phi(\tau, \tau') := P(\tau \succ \tau') - \frac{1}{2} \in \left[-\frac{1}{2}, \frac{1}{2}\right]$.

A *policy*, $\pi : \mathcal{S} \times \{1, \ldots, h\} \longrightarrow \mathcal{A}$, is a (possibly-stochastic) mapping from states and time indices to actions. In each iteration i, the agent selects two policies, π_{i1} and π_{i2}, which are rolled out to obtain trajectories τ_{i1} and τ_{i2} and preference label y_i. Each trajectory is represented as a feature vector, where each feature records the number of times that a particular state-action pair is visited. In iteration i, rolled-out trajectories τ_{i1} and τ_{i2} correspond, respectively, to feature vectors $\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2} \in \mathbb{R}^d$, where $d := SA$ is the total number of state-action pairs, and the k^{th} element of $\boldsymbol{x}_{ij}$, $j \in \{1, 2\}$, is the number of times that τ_{ij} visits state-action pair k. The preference for iteration i is denoted by $y_i := \mathbb{I}_{[\tau_{i2} \succ \tau_{i1}]} - \frac{1}{2} \in \left\{-\frac{1}{2}, \frac{1}{2}\right\}$, where $\mathbb{I}_{[\cdot]}$ is the indicator function, so that $P\left(y_i = \frac{1}{2}\right) = 1 - P\left(y_i = -\frac{1}{2}\right) = \phi(\tau_{i2}, \tau_{i1}) + \frac{1}{2}$; there are no ties in any comparisons. Lastly, the i^{th} feature difference vector is written as $\boldsymbol{x}_i := \boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}$.

Analogously to the generalized linear dueling bandit setting discussed in Section 3.1, preferences are assumed to be generated based upon underlying trajectory utilities that are linear in the state-action visitation features, as formalized in the following two assumptions. Firstly, the analysis assumes the existence of underlying utilities, which quantify the user's satisfaction with each trajectory:

Assumption 4. *Each trajectory τ has utility $u(\tau)$, which decomposes additively: $u(\tau) \equiv \sum_{t=1}^{h} \overline{r}(s_t, a_t)$ for the state-action pairs in τ. Defining $\overline{\boldsymbol{r}} \in \mathbb{R}^d$ as the vector of all state-action rewards, $u(\tau)$ can also be expressed in terms of τ's state-action visit counts $\boldsymbol{x}$: $u(\tau) = \overline{\boldsymbol{r}}^T \boldsymbol{x}$.*

Secondly, the utilities $u(\tau)$ are stochastically translated to preferences via the noise model ϕ, such that the probability of observing $\tau_{i2} \succ \tau_{i1}$ is a function of the *difference* in the trajectories' utilities. Intuitively, the greater the disparity in two trajectories' utilities, the more accurate the user's preference between them:

Assumption 5. *For two trajectories τ_{i1} and τ_{i2}, $P(\tau_{i2} \succ \tau_{i1}) = \phi(\tau_{i2}, \tau_{i1}) + \frac{1}{2} = g(u(\tau_{i2}) - u(\tau_{i1})) + \frac{1}{2} = g(\overline{\boldsymbol{r}}^T \boldsymbol{x}_{i2} - \overline{\boldsymbol{r}}^T \boldsymbol{x}_{i1}) + \frac{1}{2}$, where $g : \mathbb{R} \longrightarrow \left[-\frac{1}{2}, \frac{1}{2}\right]$ is a* link function *such that:*

a) g is non-decreasing, and

b) $g(x) = -g(-x)$ to ensure that $P(\tau \succ \tau') = 1 - P(\tau' \succ \tau)$.

Note that if $u(\tau) = u(\tau')$*, then* $P(\tau \succ \tau') = \frac{1}{2}$*, and* $P(\tau_{i2} \succ \tau_{i1}) > \frac{1}{2} \Leftrightarrow g(\overline{\boldsymbol{r}}^T \boldsymbol{x}_i) > 0 \Leftrightarrow \overline{\boldsymbol{r}}^T \boldsymbol{x}_{i2} > \overline{\boldsymbol{r}}^T \boldsymbol{x}_{i1}$.

Section 3.1 gives several examples of link functions, including the linear and logistic link functions g_{lin} and g_{log}, which are respectively defined in Eq.s (3.2) and (3.3). Similarly, the observation noise in iteration i is $\eta_i := y_i - g(\overline{\boldsymbol{r}}^T(\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}))$. For the linear link function, for instance, $\eta_i := y_i - \overline{\boldsymbol{r}}^T(\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1})$.

Lastly, as in the bandit setting, the rewards must belong to a compact set. This is satisfied by assuming the reward vector $\overline{\boldsymbol{r}}$ has bounded magnitude:

Assumption 6. *For some known* $S_r < \infty$, $||\overline{\boldsymbol{r}}||_2 \leq S_r$.

Given a policy π, the standard RL value function is defined as the expected total utility when starting in state s at step j, and following π:

$$V_{\pi,j}(s) = \mathbb{E}\left[\sum_{t=j}^{h} \overline{r}(s_t, \pi(s_t, t)) \,\middle|\, s_j = s\right]. \tag{3.6}$$

The optimal policy π^* is then defined as one that maximizes the expected value over all input states:

$$\pi^* = \sup_{\pi} \sum_{s \in \mathcal{S}} p_0(s) V_{\pi,1}(s).$$

Note that $\mathbb{E}_{s_1 \sim p_0}\left[V_{\pi,1}(s_1)\right] \equiv \mathbb{E}_{\tau \sim \pi}\left[u(\tau)\right]$. Given fully-specified dynamics and rewards, p and $\overline{r}$, it is straightforward to apply standard dynamic programming approaches such as value iteration (Sutton and Barto, 2018) to arrive at the optimal policy under p and $\overline{r}$. The learning goal, then, is to infer p and $\overline{r}$ to the extent necessary for good decision-making.

The learning agent's performance is quantified via its cumulative T-step Bayesian regret relative to the optimal policy:

$$\text{BayesReg}(T) = \mathbb{E}\left\{\sum_{i=1}^{\lceil T/(2h) \rceil} \sum_{s \in \mathcal{S}} p_0(s)\left[2V_{\pi^*,1}(s) - V_{\pi_{i1},1}(s) - V_{\pi_{i2},1}(s)\right]\right\}. \tag{3.7}$$

To minimize regret, the agent must balance exploration (collecting new data) with exploitation (behaving optimally given current knowledge). Over-exploration of bad trajectories will incur large regret, and under-exploration can prevent convergence to optimality. In contrast to the standard regret formulation in RL, the regret of both selected policies is measured in each iteration.

3.3 Comparing the Preference-Based Generalized Linear Bandit and RL Settings

In both the preference-based bandit and RL settings discussed in Sections 3.1 and 3.2, preferences are assumed to be governed by linear utilities of the form $u(\boldsymbol{x}) = \overline{\boldsymbol{r}}^T \boldsymbol{x}$, where in the bandit case, $\boldsymbol{x} \in \mathbb{R}^d$ represents the features of an action, while in the RL setting, $\boldsymbol{x}$ contains the features of a trajectory, which are the trajectory's state-action visitation counts. While in the bandit case, the algorithm directly selects actions $\boldsymbol{x}_{i1}$ and $\boldsymbol{x}_{i2}$ in each iteration i, in the RL setting, the agent selects policies π_{i1} and π_{i2}, which stochastically map to the trajectory feature representations $\boldsymbol{x}_{i1}$ and $\boldsymbol{x}_{i2}$.

In both cases, the linear and logistic link functions' analysis assumes a regularity condition upon the observation noise, $\eta_i = y_i - g(\overline{\boldsymbol{r}}^T \boldsymbol{x}_i)$. Namely, η_i must be conditionally-R-sub-Gaussian, that is, there exists $R \geq 0$ such that $\forall\, \lambda \in \mathbb{R}$:

$$\mathbb{E}\left[e^{\lambda \eta_i} \,\middle|\, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_i, \eta_1, \ldots, \eta_{i-1}\right] \leq \exp\left(\frac{\lambda^2 R^2}{2}\right).$$

One can see that this requirement is satisfied for $R = 1$ as follows. Firstly, note that bounded, zero-mean noise lying in an interval of length at most $2R$ is R-sub-Gaussian (Abbasi-Yadkori, Pál, and Szepesvári, 2011). The noise η_i is by definition conditionally-zero-mean, since given any $\overline{\boldsymbol{r}}$:

$$\begin{aligned}\mathbb{E}[\eta_i \mid \boldsymbol{x}_1, \ldots, \boldsymbol{x}_i, \eta_1, \ldots, \eta_{i-1}] &= \mathbb{E}[y_i - g(\overline{\boldsymbol{r}}^T \boldsymbol{x}_i) \mid \boldsymbol{x}_1, \ldots, \boldsymbol{x}_i, \eta_1, \ldots, \eta_{i-1}] \\ &= g(\overline{\boldsymbol{r}}^T \boldsymbol{x}_i) - g(\overline{\boldsymbol{r}}^T \boldsymbol{x}_i) = 0.\end{aligned}$$

Secondly, η_i is bounded:

$$|\eta_i| = \left|y_i - g(\overline{\boldsymbol{r}}^T \boldsymbol{x}_i)\right| \leq |y_i| + \left|g(\overline{\boldsymbol{r}}^T \boldsymbol{x}_i)\right| = \frac{1}{2} + \left|\mathbb{E}[y_i \mid \boldsymbol{x}_i]\right| \leq \frac{1}{2} + \frac{1}{2} = 1.$$

Because η_i is zero-mean and $\eta_i \in [-1, 1]$, it must be sub-Gaussian with $R = 1$.

Chapter 4

DUELING POSTERIOR SAMPLING FOR PREFERENCE-BASED BANDITS AND REINFORCEMENT LEARNING

This chapter describes, analyzes, and evaluates the Dueling Posterior Sampling (DPS) framework for preference-based bandits and reinforcement learning (RL). The framework is applied to both the preference-based RL and generalized linear bandit problem settings, which are defined in Chapter 3. Part of these results are published in Novoseller et al. (2020b).

Below, Section 4.1 introduces the DPS algorithm, which extends the SelfSparring framework from Sui, Zhuang, et al. (2017) to the preference-based generalized linear bandit and RL settings. Section 4.3 then discusses several posterior inference strategies that can be coupled with DPS. Subsequently, Section 4.4 presents a theoretical analysis of DPS in two parts: 1) an asymptotic consistency result which holds when preferences are modeled via both the linear and logistic link functions, and 2) a regret analysis for the linear link function, which is based on empirically bounding the information ratio introduced in Russo and Van Roy (2016) (defined in Section 2.3). Finally, Section 4.5 demonstrates the performance of DPS in simulation, and Section 4.6 discusses the results and proposes some future directions.

4.1 The Dueling Posterior Sampling Algorithm

The Dueling Posterior Sampling (DPS) algorithm adapts the SelfSparring framework from Sui, Zhuang, et al. (2017) to the preference-based RL setting, and as a byproduct, to the generalized linear dueling bandits problem. SelfSparring applies posterior sampling to preference-based bandit learning by maintaining a Bayesian model posterior over the environment, and by drawing multiple samples from this posterior to duel against each other for preference elicitation. This section first introduces DPS in the preference-based RL setting, and then modifies it for the generalized linear dueling bandit setting.

Dueling Posterior Sampling for Preference-Based RL

In the RL setting, DPS maintains Bayesian posteriors f_p and f_r over both the transition dynamics and utilities $\overline{\boldsymbol{r}}$, respectively. As outlined in Algorithm 5, DPS iterates among three steps: (a) sampling two policies π_{i1}, π_{i2} from the Bayesian

Algorithm 5 Dueling Posterior Sampling (DPS) for Preference-Based RL

$\mathcal{H}_0 = \emptyset$ ▷ Initialize history
Initialize prior for f_p ▷ Initialize state transition model
Initialize prior for f_r ▷ Initialize utility model
for $i = 1, 2, \ldots$ **do**
    $\pi_{i1} \leftarrow$ Advance (f_p, f_r)
    $\pi_{i2} \leftarrow$ Advance (f_p, f_r)
    Sample trajectories τ_{i1} and τ_{i2} from π_{i1} and π_{i2}
    Observe feedback $y_i = \mathbb{I}_{[\tau_{i2} \succ \tau_{i1}]} - \frac{1}{2}$
    $\mathcal{H}_i = \mathcal{H}_{i-1} \cup (\tau_{i1}, \tau_{i2}, y_i)$
    $f_p, f_r =$ Feedback $(\mathcal{H}_i, f_p, f_r)$
end for

Algorithm 6 Advance (RL): Sample policy from dynamics and utility models

Input: transition model f_p, utility model f_r
Sample $\tilde{\boldsymbol{p}} \sim f_p(\cdot)$ ▷ Sample MDP transition dynamics parameters from posterior
Sample $\tilde{\boldsymbol{r}} \sim f_r(\cdot)$ ▷ Sample utilities from posterior
Compute $\pi = \text{argmax}_\pi V(\tilde{\boldsymbol{p}}, \tilde{\boldsymbol{r}})$ ▷ Value iteration yields sampled MDP's optimal policy
Return π

Algorithm 7 Feedback (RL): Update dynamics and utility models based on new user feedback

Input: history $\mathcal{H}$, transition model f_p, utility model f_r
Apply Bayesian update to f_p, given $\mathcal{H}$ ▷ Update dynamics model given history
Apply Bayesian update to f_r, given $\mathcal{H}$ ▷ Update utility model given preferences
Return f_p, f_r

posteriors of the dynamics and utility models (Advance – Algorithm 6); (b) rolling out π_{i1} and π_{i2} to obtain trajectories τ_{i1} and τ_{i2}, and receiving a preference y_i between them; and (c) updating the posterior (Feedback – Algorithm 7). In contrast to conventional posterior sampling with absolute feedback, DPS samples two policies rather than one during each iteration and solves a credit assignment problem to learn from feedback.

Advance (Algorithm 6) selects a policy to execute by drawing samples from the Bayesian posteriors of both the dynamics and utilities. The sampled dynamics and utilities form an MDP, for which finite-horizon value iteration (Sutton and Barto, 2018) derives the optimal policy π under the sample. One can also view π as a random function whose randomness depends on the sampling of the dynamics and utility models. In the Bayesian setting, it can be shown that π is sampled according to its posterior probability of being the optimal policy π^* (Osband, Russo, and Van Roy, 2013). Intuitively, peaked (i.e., certain) posteriors lead to less variability when

Algorithm 8 Dueling Posterior Sampling (DPS) for Generalized Linear Dueling Bandits

$\mathcal{H}_0 = \emptyset$ ▷ Initialize history
Initialize prior for f_r ▷ Initialize utility model
for $i = 1, 2, \ldots$ **do**
    $\boldsymbol{x}_{i1} \leftarrow$ Advance (f_r)
    $\boldsymbol{x}_{i2} \leftarrow$ Advance (f_r)
    Observe feedback $y_i = \mathbb{I}_{[\boldsymbol{x}_{i2} \succ \boldsymbol{x}_{i1}]} - \frac{1}{2}$
    $\mathcal{H}_i = \mathcal{H}_{i-1} \cup (\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}, y_i)$
    $f_r =$ Feedback $(\mathcal{H}_i, f_r)$
end for

Algorithm 9 Advance (bandit): Sample policy from utility model

Input: utility model f_r
Sample $\tilde{\boldsymbol{r}} \sim f_r(\cdot)$ ▷ Sample utilities from posterior
$\boldsymbol{x} = \text{argmax}_{\boldsymbol{x}' \in \mathcal{A}} \tilde{\boldsymbol{r}}^T \boldsymbol{x}'$ ▷ Identify action with maximum sampled reward
Return $\boldsymbol{x}$

Algorithm 10 Feedback (bandit): Update utility model based on new user feedback

Input: history $\mathcal{H}$, utility model f_r
Apply Bayesian update to f_r, given $\mathcal{H}$ ▷ Update utility model given preferences
Return f_r

sampling π, which implies less exploration, while diffuse (i.e., uncertain) posteriors lead to greater variability when sampling π, implying more exploration.

Feedback (Algorithm 7) updates the Bayesian posteriors of the dynamics and utility models based on new data. Updating the dynamics posterior is relatively straightforward, as the dynamics are assumed to be fully-observed. This work models the dynamics prior via a Dirichlet distribution for each state-action pair; the observed state transitions have a multinomial likelihood, leading to a conjugate Dirichlet posterior (see Section 2.1). In contrast, performing Bayesian inference over state-action utilities from trajectory-level feedback is much more challenging. A range of approaches are therefore considered, as discussed in Section 4.3. In particular, Bayesian linear regression was found to both perform well and to admit tractable analysis within the theoretical framework presented.

Dueling Posterior Sampling for Generalized Linear Dueling Bandits

The bandit setting can be viewed as a special case of the RL problem in which there is only one state. Thus, modeling state transition information is unnecessary, and the environment posterior consists only of the utility model. Secondly, while the RL setting stochastically maps policies to trajectories, in the bandit setting, the

concepts of a policy and a trajectory both reduce to the selected action. Thirdly, each action is associated with a d-dimensional vector, rather than with just an index in $\{1, \ldots, A\}$, as in tabular RL. While in the RL case, the utility vector $\overline{\boldsymbol{r}}$ denotes the utility of each state-action pair, in the generalized linear bandit case, it specifies a linear weight on each action space dimension. With these modifications, DPS is outlined in Algorithms 8-10.

4.2 Additional Notation

This thesis uses the following notation. In the RL setting, let $\overline{\boldsymbol{p}} \in \mathbb{R}^{S^2A}$ be the vector containing all true state transition dynamics parameters, $\{P(s_{t+1} = s' \mid s_t = s, a_t = a) \mid s, s' \in \mathcal{S}, a \in \mathcal{A}\}$. Let $\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{p}}_{i2} \in \mathbb{R}^{S^2A}$ be the two posterior samples of the transition dynamics $\overline{\boldsymbol{p}}$ in iteration i. Similarly, $\overline{\boldsymbol{r}} \in \mathbb{R}^{SA}$ is the vector of true utility parameters, while $\tilde{\boldsymbol{r}}_{i1}, \tilde{\boldsymbol{r}}_{i2} \in \mathbb{R}^{SA}$ are the posterior samples of $\overline{\boldsymbol{r}}$ in iteration i.

For a random variable X and a sequence of random variables (X_n), $n \in \mathbb{N}$, $X_n \xrightarrow{D} X$ denotes that X_n converges to X in distribution. Similarly, $X_n \xrightarrow{P} X$ denotes that X_n converges to X in probability.

For $\boldsymbol{x} \in \mathbb{R}^d$ and positive definite matrix $B \in \mathbb{R}^{d \times d}$, the norm $||\boldsymbol{x}||_B$ is defined as $||\boldsymbol{x}||_B := \sqrt{\boldsymbol{x}^T B \boldsymbol{x}}$.

4.3 Posterior Modeling for Utility Inference and Credit Assignment

At the start of iteration n, the algorithm has observed $n - 1$ preferences. Recall that in each learning iteration $i \in \{1, \ldots, n-1\}$, the algorithm observes a preference $y_i \in \left\{-\frac{1}{2}, \frac{1}{2}\right\}$ between two vectors $\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2} \in \mathbb{R}^d$. Importantly, the quantities $\boldsymbol{x}_{i1}$ and $\boldsymbol{x}_{i2}$ represent actions in the bandit problem, while they are trajectory feature vectors (more specifically, state-action pair visitation counts) in the RL problem. Yet, in both settings, the preference labels y_i are assumed to be generated according to the same model:

$$P\left(y_i = \frac{1}{2}\right) = P(\boldsymbol{x}_{i2} \succ \boldsymbol{x}_{i1}) = g(\overline{\boldsymbol{r}}^T \boldsymbol{x}_i),$$

where g is the link function and $\boldsymbol{x}_i := \boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}$. Although $\boldsymbol{x}_i$ has a different interpretation in each of the two settings, the utility inference task takes the same form in both: to estimate $\overline{\boldsymbol{r}}$ given the dataset $\mathcal{D}_n := \{(\boldsymbol{x}_i, y_i) \mid i = 1, \ldots, n-1\}$.

In the RL setting, utility inference is known as the *credit assignment problem* because it involves identifying which state-action pairs are responsible for the observed trajectory-level preferences.

Let $X_n \in \mathbb{R}^{(n-1)\times d}$ be the observation matrix after $n-1$ preferences, in which the i^{th} row contains observation $\boldsymbol{x}_i = \boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}$. Furthermore, define $\boldsymbol{y}_n \in \mathbb{R}^{n-1}$ as the vector of corresponding preference labels, with its i^{th} element equal to $y_i \in \left\{-\frac{1}{2}, \frac{1}{2}\right\}$.

The following two subsections discuss Bayesian linear regression and Bayesian logistic regression approaches to utility inference. In addition, Appendix A discusses Gaussian process methods for posterior inference of the utilities.

Remark 4. *The posterior inference methods discussed in this section also apply to the multi-dueling setting (see Section 2.5), where in each iteration, more than 2 actions or policies can be sampled to yield multiple pairwise preferences. Posterior inference simply translates a preference dataset to a posterior distribution; this procedure is agnostic to the number of preference queries per iteration.*

Utility Inference via Bayesian Linear Regression

Several different variants of Bayesian linear regression could be performed to infer the utilities, $\overline{\boldsymbol{r}}$. Firstly, in standard Bayesian linear regression, one defines a Gaussian prior over the reward vector $\overline{\boldsymbol{r}} \in \mathbb{R}^d$: $\boldsymbol{r} \sim \mathcal{N}(0, (\lambda')^{-1} I)$. The likelihood of the data conditioned upon $\boldsymbol{r}$ is also Gaussian:

$$p(\mathbf{y}_n | X_n, \boldsymbol{r}; \sigma^2) = \frac{1}{(2\pi\sigma^2)^{\frac{n-1}{2}}} \exp\left(-\frac{1}{2\sigma^2} ||\mathbf{y}_n - X_n \boldsymbol{r}||^2\right).$$

This conjugate Gaussian prior and likelihood pair result in the following closed-form Gaussian posterior:

$$\boldsymbol{r} | X_n, \boldsymbol{y}_n, \sigma^2, \lambda' \sim \mathcal{N}\left((X_n^T X_n + \sigma^2 \lambda' I)^{-1} X_n^T \boldsymbol{y}_n,\ \sigma^2 (X_n^T X_n + \sigma^2 \lambda' I)^{-1}\right).$$

Defining $\lambda := \sigma^2 \lambda'$, this posterior can equivalently be written as:

$$\boldsymbol{r} | \mathcal{D}_n, \sigma^2, \lambda \sim \mathcal{N}\left(\hat{\boldsymbol{r}}_n,\ \sigma^2 M_n^{-1}\right), \text{ where} \tag{4.1}$$

$$M_n := \lambda I + \sum_{i=1}^{n-1} \boldsymbol{x}_i \boldsymbol{x}_i^T \text{ for } \lambda \geq 1, \text{ and } \hat{\boldsymbol{r}}_n := M_n^{-1} \sum_{i=1}^{n-1} y_i \boldsymbol{x}_i. \tag{4.2}$$

Note that $\hat{\boldsymbol{r}}_n$ is the MAP estimate of $\overline{\boldsymbol{r}}$, as well as a ridge regression estimate of $\overline{\boldsymbol{r}}$.

In the linear bandit setting, several posterior sampling regret analyses (Agrawal and Goyal, 2013; Abeille and Lazaric, 2017) use a modification of this posterior, which scales the covariance matrix by a factor that increases logarithmically in n:

$$\boldsymbol{r} | \mathcal{D}_n, \lambda \sim \mathcal{N}\left(\hat{\boldsymbol{r}}_n,\ \beta_n(\delta)^2 M_n^{-1}\right), \tag{4.3}$$

where $\hat{\boldsymbol{r}}_n$ and M_n^{-1} are defined in Eq. (4.2), and $\beta_n(\delta)$ is given by:

$$\beta_n(\delta) := R\sqrt{2\log\left(\frac{\det(M_n)^{1/2}\lambda^{-d/2}}{\delta}\right)} + \sqrt{\lambda}S_r \leq R\sqrt{d\log\left(\frac{1+\frac{L^2 n}{d\lambda}}{\delta}\right)} + \sqrt{\lambda}S_r, \quad (4.4)$$

where $\delta \in (0,1)$ is a failure probability to be discussed shortly, and L is an upper bound such that $||\boldsymbol{x}_n||_2 \leq L$ for all n. Note that in the RL setting, $L \leq 2h$, since $||\boldsymbol{x}_n||_2 = ||\boldsymbol{x}_{n2} - \boldsymbol{x}_{n1}||_2 \leq ||\boldsymbol{x}_{n2} - \boldsymbol{x}_{n1}||_1 \leq ||\boldsymbol{x}_{n2}||_1 + ||\boldsymbol{x}_{n1}||_1 = 2h$. Recall that $R \leq 1$ (see Section 3.3) due to the sub-Gaussianity of preference feedback, and that $||\overline{\boldsymbol{r}}||_2 \leq S_r$ (see Assumptions 1 and 6).

In Agrawal and Goyal (2013) and Abeille and Lazaric (2017), the factor of $\beta_n(\delta)^2$ is leveraged to prove regret bounds for linear bandits with sub-Gaussian noise in the feedback. In the current preference-based problem, meanwhile, scaling by $\beta_n(\delta)^2$ will lead to an asymptotic consistency guarantee. This is shown in Section 4.4 by applying the following result from Abbasi-Yadkori, Pál, and Szepesvári (2011), which arises from a martingale concentration inequality. This proposition demonstrates that with high probability, $\overline{\boldsymbol{r}}$ lies within a confidence ellipsoid centered at $\hat{\boldsymbol{r}}_n$:

Proposition 1 (Theorem 2 from Abbasi-Yadkori, Pál, and Szepesvári, 2011)**.** *Let $\{F_i\}_{i=0}^{\infty}$ be a filtration, and let $\{\eta_i\}_{i=1}^{\infty}$ be a real-valued stochastic process such that η_i is both F_i-measurable and conditionally R-sub-Gaussian for some $R \geq 0$. Let $\{\boldsymbol{x}_i\}$ be an $\mathbb{R}^d$-valued stochastic process such that $\boldsymbol{x}_i$ is F_{i-1}-measurable. Define $y_i := \boldsymbol{x}_i^T\overline{\boldsymbol{r}} + \eta_i$, and assume that $||\overline{\boldsymbol{r}}||_2 \leq S_r$ and $||\boldsymbol{x}_i||_2 \leq L$. Then, for any $\delta > 0$, with probability at least $1-\delta$ and for all $i > 0$, $||\hat{\boldsymbol{r}}_i - \overline{\boldsymbol{r}}||_{M_i} \leq \beta_i(\delta)$, where $\hat{\boldsymbol{r}}_i$, M_i and $\beta_i(\delta)$ are defined in Eq.s* (4.2) *and* (4.4).

Importantly, this result does not require the noise in the labels y_i to be Gaussian, but instead only assumes that the noise is sub-Gaussian. Furthermore, preference noise is sub-Gaussian as discussed in Section 3.3.

Note that while the distribution in Eq. (4.1) is an exact, conjugate posterior given a Gaussian prior and likelihood, the scaled distribution in Eq. (4.3) is no longer an exact posterior, as it is not proportional to the product of a prior and likelihood. In fact, as discussed in Agrawal and Goyal (2013) and Abeille and Lazaric (2017), posterior sampling does not need to sample from an actual Bayesian posterior distribution; rather, any distribution satisfying appropriate concentration and anti-concentration inequalities will achieve low regret.

In fact, neither of the probability distributions in Eq.s (4.1) or (4.3) represents a true, exact posterior for modeling preference data, as binary preference feedback does not exhibit a Gaussian likelihood. Recall that $y_i \in \left\{-\frac{1}{2}, \frac{1}{2}\right\}$, such that $y_i = \frac{1}{2}$ if $\boldsymbol{x}_{i2} \succ \boldsymbol{x}_{i1}$ and $y_i = -\frac{1}{2}$ if $\boldsymbol{x}_{i1} \succ \boldsymbol{x}_{i2}$. The exact posterior takes the form:

$$p(\boldsymbol{r} \mid \mathcal{D}_n) \propto p(\boldsymbol{r}) \prod_{i=1}^{n-1} P(y_i \mid \boldsymbol{x}_i, \boldsymbol{r}) = p(\boldsymbol{r}) \prod_{i=1}^{n-1} \left[2y_i \boldsymbol{r}^T \boldsymbol{x}_i + \frac{1}{2}\right], \tag{4.5}$$

where $p(\boldsymbol{r})$ is an appropriate prior for $\overline{\boldsymbol{r}}$ and $2y_i \boldsymbol{r}^T \boldsymbol{x}_i \in (-0.5, 0.5)$ ensures that $P(y_i \mid \boldsymbol{x}_i, \boldsymbol{r}) \in [0, 1]$. To guarantee that $2y_i \boldsymbol{r}^T \boldsymbol{x}_i \in (-0.5, 0.5)$, the prior $p(\boldsymbol{r})$ must have bounded support; possibilities include uniform and truncated Gaussian distributions.

While the posterior in Eq. (4.5) lacks a closed-form representation and is therefore intractable, one can sample from it via approximate posterior inference techniques such as the Laplace approximation (Section 2.1) or Markov chain Monte Carlo sampling. The Laplace approximation is only valid when the Hessian matrix of the negative log posterior is positive definite. In this case, the negative log posterior is:

$$-\log p(\boldsymbol{r} \mid \mathcal{D}_n) = -\log p(\boldsymbol{r}) - \sum_{i=1}^{n-1} \log\left(2y_i \boldsymbol{r}^T \boldsymbol{x}_i + \frac{1}{2}\right) + C,$$

where C is a constant. Its Hessian matrix is:

$$H := \nabla_{\boldsymbol{r}}^2[-\log p(\boldsymbol{r} \mid \mathcal{D}_n)] = \nabla_{\boldsymbol{r}}^2[-\log p(\boldsymbol{r})] + \sum_{i=1}^{n-1} \frac{\boldsymbol{x}_i \boldsymbol{x}_i^T}{(2y_i \boldsymbol{r}^T \boldsymbol{x}_i + 0.5)^2}.$$

Clearly, the terms containing $\boldsymbol{x}_i \boldsymbol{x}_i^T$ are positive semidefinite. Thus, $H \succ 0$ holds, provided that the selected prior satisfies $\nabla_{\boldsymbol{r}}^2[-\log p(\boldsymbol{r})] \succ 0$. Under such a prior, the Laplace approximation is given by:

$$p(\boldsymbol{r} \mid \mathcal{D}_n) \approx \mathcal{N}\left(\hat{\boldsymbol{r}}_n,\, H^{-1}\big|_{\boldsymbol{r}=\hat{\boldsymbol{r}}_n}\right),$$

where $\hat{\boldsymbol{r}}_n = \operatorname{argmin}_{\boldsymbol{r}}[-\log p(\boldsymbol{r} \mid \mathcal{D}_n)]$ and $H = \nabla_{\boldsymbol{r}}^2[-\log p(\boldsymbol{r} \mid \mathcal{D}_n)]$.

Utility Inference via Bayesian Logistic Regression

One can also infer the utilities $\overline{\boldsymbol{r}} \in \mathbb{R}^d$ via Bayesian logistic regression, especially if a logistic link function is thought to govern user preferences. Typically, one sets a Gaussian prior over the utilities $\overline{\boldsymbol{r}}$: $\boldsymbol{r} \sim \mathcal{N}(\boldsymbol{0}, \lambda I)$, where $\lambda > 0$. As in the linear case, the outcomes are $y_i \in \left\{-\frac{1}{2}, \frac{1}{2}\right\}$. Then, the logistic regression likelihood is:

$$p(\mathcal{D}_n|\boldsymbol{r}) = \prod_{i=1}^{n-1} p(y_i|\boldsymbol{r}, \boldsymbol{x}_i) = \prod_{i=1}^{n-1} \frac{1}{1 + \exp(-2y_i \boldsymbol{x}_i^T \boldsymbol{r})}.$$

The posterior, $p(\boldsymbol{r} \mid \mathcal{D}_n) \propto p(\mathcal{D}_n \mid \boldsymbol{r})p(\boldsymbol{r})$, can be approximated as Gaussian via the Laplace approximation:

$$p(\boldsymbol{r} \mid \mathcal{D}_n) \approx \mathcal{N}(\hat{\boldsymbol{r}}_n^{\text{MAP}}, \Sigma_n^{\text{MAP}}), \text{ where:} \tag{4.6}$$

$$\hat{\boldsymbol{r}}_n^{\text{MAP}} = \underset{\boldsymbol{r}}{\operatorname{argmin}}\, f(\boldsymbol{r}), \quad f(\boldsymbol{r}) := -\log p(\mathcal{D}_n, \boldsymbol{r}) = -\log p(\boldsymbol{r}) - \log p(\mathcal{D}_n|\boldsymbol{r}), \tag{4.7}$$

$$\Sigma_n^{\text{MAP}} = \left(\nabla_{\boldsymbol{r}}^2 f(\boldsymbol{r}) \Big|_{\hat{\boldsymbol{r}}_n^{\text{MAP}}} \right)^{-1}, \tag{4.8}$$

and where the optimization problem in Eq. (4.7) is convex: similarly to the linear case, one can show that the Hessian matrix of $f(\boldsymbol{r})$, $\nabla_{\boldsymbol{r}}^2[f(\boldsymbol{r})]$, is positive definite.

Note that the Laplace approximation's inverse posterior covariance, $(\Sigma_n^{\text{MAP}})^{-1}$, is given by:

$$(\Sigma_n^{\text{MAP}})^{-1} = \lambda I + \sum_{i=1}^{n-1} \tilde{g}(2y_i \boldsymbol{x}_i^T \hat{\boldsymbol{r}}_n^{\text{MAP}}) \boldsymbol{x}_i \boldsymbol{x}_i^T, \tag{4.9}$$

where $\tilde{g}(x) := \left(\frac{g'_{\log}(x)}{g_{\log}(x)} \right)^2 - \frac{g''_{\log}(x)}{g_{\log}(x)}$, and $g_{\log}$ is the sigmoid function.

The posterior $p(\boldsymbol{r} \mid \mathcal{D}_n)$ can also be estimated using other posterior approximation methods, for instance Markov chain Monte Carlo.

Filippi et al. (2010) prove that for logistic regression, the true utilities $\overline{\boldsymbol{r}}$ lie within a confidence ellipsoid centered at the estimate $\hat{\boldsymbol{r}}_n$ with high probability, where $\hat{\boldsymbol{r}}_n$ is the maximum quasi-likelihood estimator of $\overline{\boldsymbol{r}}$, projected onto the domain Θ:

$$\hat{\boldsymbol{r}}_n = \underset{\boldsymbol{r} \in \Theta}{\operatorname{argmin}} \left\| \sum_{i=1}^{n-1} \left(g_{\log}(\boldsymbol{x}_i^T \boldsymbol{r}) \boldsymbol{x}_i - \left(y_i + \frac{1}{2} \right) \boldsymbol{x}_i \right) \right\|_{M_n^{-1}}, \tag{4.10}$$

where M_n is as defined in Eq. (4.2) and $g_{\log}$ is the sigmoidal function, $g_{\log}(x) = (1 + e^{-x})^{-1}$.[1]

Proposition 2 (Proposition 1 from Filippi et al., 2010)**.** *Let $\{F_i\}_{i=0}^{\infty}$ be a filtration, and let $\{\eta_i\}_{i=1}^{\infty}$ be a real-valued stochastic process such that η_i is F_i-measurable and conditionally R-sub-Gaussian for some $R \geq 0$. Let $\{\boldsymbol{x}_i\}$ be an $\mathbb{R}^d$-valued stochastic process such that $\boldsymbol{x}_i$ is F_{i-1}-measurable. The observations $y_i \in \left\{-\frac{1}{2}, \frac{1}{2}\right\}$ are given*

[1] The equation $\sum_{i=1}^{n-1} \left(g_{\log}(\boldsymbol{x}_i^T \boldsymbol{r}) \boldsymbol{x}_i - \left(y_i + \frac{1}{2} \right) \boldsymbol{x}_i \right) = 0$ is known as the *likelihood equation*, and its unique solution is the maximum likelihood estimate of the logistic regression problem. This fact is derived, for instance, on page 2 of Gourieroux and Monfort (1981). Because the typical form of the likelihood equation assumes that the labels belong to $\{0, 1\}$, Eq. (4.10) adds 0.5 to the labels in order to use this standard form, mirroring Filippi et al. (2010).

by $y_i = g_{\log}(\boldsymbol{x}_i^T \overline{\boldsymbol{r}}) + \eta_i$*, where* $g_{\log}$ *is a sigmoidal function. Assume that* $||\boldsymbol{x}_i||_2 \leq L$*,* $g'_{\min}$ *is a positive lower bound on the slope of* $g_{\log}$*, and* $\lambda > 0$ *lower-bounds the minimum eigenvalue of* M_i*. Then, for any* $\delta > 0$*, with probability at least* $1 - \delta$ *and for all* $i > 0$*,* $||\hat{\boldsymbol{r}}_i - \overline{\boldsymbol{r}}||_{M_i} \leq \beta_i(\delta)$*, where:*

$$\beta_i(\delta) = \frac{2R}{g'_{\min}} \sqrt{3 + 2\log\left(1 + \frac{2L^2}{\lambda}\right)} \sqrt{2d \log i} \sqrt{\log\left(\frac{d}{\delta}\right)}. \tag{4.11}$$

Recall from Eq. (4.2) that $\lambda > 0$ indeed lower-bounds M_i's minimum eigenvalue.

Remark 5. *In fact, the result in Filippi et al. (2010) applies not only to logistic regression on binary labels, but also to any generalized linear model. Generalized linear models are a well-known statistical framework in which outcomes are generated from distributions belonging to the exponential family, and are further described in McCullagh and Nelder (1989).*

Proof. The stated result is in fact a slightly-adapted version of Proposition 1 from Filippi et al. (2010), which actually derives that $|g(\boldsymbol{x}_i^T \overline{\boldsymbol{r}}) - g(\boldsymbol{x}_i^T \hat{\boldsymbol{r}}_i)| \leq \beta'_i(\delta) ||\boldsymbol{x}_i||_{M_i^{-1}}$ for all $i \geq 1$ and vectors $\boldsymbol{x}_i$ with probability $1 - \delta$. The constant $\beta'_i(\delta)$ is related to $\beta_i(\delta)$ (defined above) by $\beta'_i(\delta) = \left(\frac{R_{\max}}{R}\right) L_g \beta_i(\delta)$, where L_g is the Lipschitz constant of g (denoted by k_μ in Filippi et al., 2010) and where each reward lies in $[0, R_{\max}]$ in the setting described in Filippi et al. (2010). To prove their result, Filippi et al. (2010) use that $|g(\boldsymbol{x}_i^T \overline{\boldsymbol{r}}) - g(\boldsymbol{x}_i^T \hat{\boldsymbol{r}}_i)| \leq L_g |\boldsymbol{x}_i^T \overline{\boldsymbol{r}} - \boldsymbol{x}_i^T \hat{\boldsymbol{r}}_i|$ by definition of Lipschitz continuity, and then show that $|\boldsymbol{x}_i^T \overline{\boldsymbol{r}} - \boldsymbol{x}_i^T \hat{\boldsymbol{r}}_i| \leq \frac{1}{L_g} \beta'_i(\delta) ||\boldsymbol{x}_i||_{M_i^{-1}}$ to obtain the result. Therefore, as a byproduct, their analysis also proves that:

$$|\boldsymbol{x}_i^T \overline{\boldsymbol{r}} - \boldsymbol{x}_i^T \hat{\boldsymbol{r}}_i| \leq \frac{1}{L_g} \beta'_i(\delta) ||\boldsymbol{x}_i||_{M_i^{-1}} \overset{(a)}{=} \frac{R_{\max}}{R} \beta_i(\delta) ||\boldsymbol{x}_i||_{M_i^{-1}},$$

where (a) comes from substituting the relationship between $\beta_i(\delta)$ and $\beta'_i(\delta)$.

Also, the analysis in Filippi et al. (2010) begins with quantities in terms of R instead of $R_{\max}$, and then replaces R by $R_{\max}$ by using that $R \leq R_{\max}$ (which holds in their setting). Therefore, their result is still valid when reverting from $R_{\max}$ to R, so that:

$$|\boldsymbol{x}_i^T \overline{\boldsymbol{r}} - \boldsymbol{x}_i^T \hat{\boldsymbol{r}}_i| \leq \beta_i(\delta) ||\boldsymbol{x}_i||_{M_i^{-1}}.$$

To adapt this result to the desired form, it suffices to show that the following two statements are equivalent for any $\boldsymbol{r}_1, \boldsymbol{r}_2 \in \mathbb{R}^d$, positive definite $M \in \mathbb{R}^{d \times d}$, and $\beta > 0$: 1) $|\boldsymbol{x}^T (\boldsymbol{r}_1 - \boldsymbol{r}_2)| \leq \beta ||\boldsymbol{x}||_{M^{-1}}$ for all $\boldsymbol{x} \in \mathbb{R}^d$, and 2) $||\boldsymbol{r}_1 - \boldsymbol{r}_2||_M \leq \beta$.

Firstly, if 1) holds, then in particular, it holds when setting $\boldsymbol{x} = M(\boldsymbol{r}_1 - \boldsymbol{r}_2)$. Substituting this definition of $\boldsymbol{x}$ into 1) yields: $|(\boldsymbol{r}_1 - \boldsymbol{r}_2)^T M(\boldsymbol{r}_1 - \boldsymbol{r}_2)| \leq \beta ||M(\boldsymbol{r}_1 - \boldsymbol{r}_2)||_{M^{-1}}$. Now, the left-hand side is $|(\boldsymbol{r}_1 - \boldsymbol{r}_2)^T M(\boldsymbol{r}_1 - \boldsymbol{r}_2)| = ||\boldsymbol{r}_1 - \boldsymbol{r}_2||_M^2$, while on the right-hand side, $||M(\boldsymbol{r}_1 - \boldsymbol{r}_2)||_{M^{-1}} = \sqrt{(\boldsymbol{r}_1 - \boldsymbol{r}_2)^T M M^{-1} M(\boldsymbol{r}_1 - \boldsymbol{r}_2)} = ||\boldsymbol{r}_1 - \boldsymbol{r}_2||_M$. Thus, the statement is equivalent to $||\boldsymbol{r}_1 - \boldsymbol{r}_2||_M^2 \leq \beta ||\boldsymbol{r}_1 - \boldsymbol{r}_2||_M$, and therefore $||\boldsymbol{r}_1 - \boldsymbol{r}_2||_M \leq \beta$ as desired.

Secondly, if 2) holds, then for any $\boldsymbol{x} \in \mathbb{R}^d$,

$$\begin{aligned} |\boldsymbol{x}^T(\boldsymbol{r}_1 - \boldsymbol{r}_2)| &= |\boldsymbol{x}^T M^{-1} M(\boldsymbol{r}_1 - \boldsymbol{r}_2)| = \left| < \boldsymbol{x}, M(\boldsymbol{r}_1 - \boldsymbol{r}_2) >_{M^{-1}} \right| \\ &\overset{(a)}{\leq} ||\boldsymbol{x}||_{M^{-1}} ||M(\boldsymbol{r}_1 - \boldsymbol{r}_2)||_{M^{-1}} = ||\boldsymbol{x}||_{M^{-1}} \sqrt{(\boldsymbol{r}_1 - \boldsymbol{r}_2)^T M M^{-1} M(\boldsymbol{r}_1 - \boldsymbol{r}_2)} \\ &= ||\boldsymbol{x}||_{M^{-1}} ||\boldsymbol{r}_1 - \boldsymbol{r}_2||_M \overset{(b)}{\leq} \beta ||\boldsymbol{x}||_{M^{-1}}, \end{aligned}$$

where (a) applies the Cauchy-Schwarz inequality, while (b) uses 2). □

This theoretical guarantee motivates the following modification to the Laplace-approximation posterior, which will be useful for guaranteeing asymptotic consistency:

$$p(\boldsymbol{r} \mid \mathcal{D}_n) \approx \mathcal{N}(\hat{\boldsymbol{r}}_n, \beta_n(\delta)^2 (M_n')^{-1}), \tag{4.12}$$

where $\hat{\boldsymbol{r}}_n$, and $\beta_n(\delta)$ are respectively defined in Eq.s (4.10) and (4.11), and M_n' is given analogously to Σ_n^{MAP}, defined in Eq.s (4.6) and (4.9), by simply replacing $\hat{\boldsymbol{r}}_n^{\text{MAP}}$ with $\hat{\boldsymbol{r}}_n$ in the definition in Eq. (4.9):

$$(M_n')^{-1} = \lambda I + \sum_{i=1}^{n-1} \tilde{g}(2 y_i \boldsymbol{x}_i^T \hat{\boldsymbol{r}}_n) \boldsymbol{x}_i \boldsymbol{x}_i^T,$$

where $\tilde{g}(x) := \left(\frac{g_{\log}'(x)}{g_{\log}(x)} \right)^2 - \frac{g_{\log}''(x)}{g_{\log}(x)}$, and $g_{\log}$ is the sigmoid function.

4.4 Theoretical Analysis

This section begins by deriving several asymptotic consistency results for DPS, and then motivates the use of an information-theoretic analysis inspired by Russo and Van Roy (2016) to analyze the regret of DPS. The regret analysis is subsequently discussed.

Asymptotic Consistency Results

Propositions 1 and 2 (derived in Abbasi-Yadkori, Pál, and Szepesvári, 2011 and Filippi et al., 2010, respectively) prove that with high probability, the utilities $\overline{\boldsymbol{r}}$

lie within confidence ellipsoids centered at the estimators $\hat{\boldsymbol{r}}_n$, defined in Eq.s (4.2) and (4.10) for the linear and logistic link functions. These results can be leveraged to prove asymptotic consistency of DPS when the algorithm draws utility samples from the distributions in Eq.s (4.3) and (4.12), rewritten here:

$$p(\boldsymbol{r} \mid \mathcal{D}_n) \approx \mathcal{N}(\hat{\boldsymbol{r}}_n, \beta_n(\delta)^2 \tilde{M}_n^{-1}), \tag{4.13}$$

where $\hat{\boldsymbol{r}}_n$ and $\beta_n(\delta)$ have different definitions corresponding to the linear and logistic link functions, and $\tilde{M}_n = M_n$ for the linear link function, while $\tilde{M}_n = M'_n$ for the logistic link function. This overloaded notation is useful because the asymptotic consistency proofs are nearly-identical in the two cases.

The asymptotic consistency results are stated below, with detailed proofs given in Appendix B.

Firstly, in the RL setting, the MDP transition dynamics model is asymptotically-consistent, that is, the sampled transition dynamics parameters converge in distribution to the true dynamics. Recall from Section 4.2 that $\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{p}}_{i2} \in \mathbb{R}^{S^2A}$ are the two posterior samples of the transition dynamics $\overline{\boldsymbol{p}}$ in iteration i, while $\overline{\boldsymbol{p}}$ represents the true transition dynamics parameters and $\xrightarrow{D}$ denotes convergence in distribution.

Proposition 3. *Assume that DPS is executed in the preference-based RL setting, with transition dynamics modeled via a Dirichlet model, utilities modeled via either the linear or logistic link function, and utility posterior sampling distributions given in Eq.* (4.13)*. Then, the sampled transition dynamics* $\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{p}}_{i2}$ *converge in distribution to the true dynamics,* $\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{p}}_{i2} \xrightarrow{D} \overline{\boldsymbol{p}}$*. This consistency result also holds when removing the* $\beta_n(\delta)$ *factors from the distributions in Eq.* (4.13)*.*

Proof sketch. The full proof is located in Appendix B.2. Applying standard concentration inequalities (i.e., Propositions 1 and 2) to the Dirichlet dynamics posterior, one can show that the sampled dynamics converge in distribution to their true values if every state-action pair is visited infinitely often. The latter condition can be proven via contradiction: assuming that certain state-action pairs are visited only finitely often, DPS does not receive new information about their rewards. Examining their reward posteriors, one can show that DPS is guaranteed to eventually sample high enough rewards in the unvisited state-actions that its policies will attempt to reach them.

It can similarly be proven that the reward samples $\tilde{\boldsymbol{r}}_{i1}, \tilde{\boldsymbol{r}}_{i2}$ converge in distribution to the true utilities, $\overline{\boldsymbol{r}}$. This phenomenon is first considered in the bandit setting, and then extended to the RL case.

Proposition 4. *When running DPS in the generalized linear dueling bandit setting, with utilities given via either the linear or logistic link functions and with posterior sampling distributions given in Eq.* (4.13)*, then with probability* $1 - \delta$*, the sampled utilities* $\tilde{\boldsymbol{r}}_{i1}, \tilde{\boldsymbol{r}}_{i2}$ *converge in distribution to their true values,* $\tilde{\boldsymbol{r}}_{i1}, \tilde{\boldsymbol{r}}_{i2} \xrightarrow{D} \overline{\boldsymbol{r}}$*, as* $i \longrightarrow \infty$.

Proof sketch. The full proof is located in Appendix B.3. By leveraging Propositions 1 and 2 (proven in Abbasi-Yadkori, Pál, and Szepesvári, 2011 and Filippi et al., 2010), one obtains that under stated conditions, for any $\delta > 0$ and for all $i > 0$, $||\hat{\boldsymbol{r}}_i - \overline{\boldsymbol{r}}||_{M_i} \leq \beta_i(\delta)$ with probability $1 - \delta$. This result defines a high-confidence ellipsoid, which can be linked to the posterior sampling distribution. The analysis demonstrates that it suffices to show that all eigenvalues of the posterior covariance matrix, $\beta_i(\delta)^2 \tilde{M}_i^{-1}$, converge in distribution to zero. This statement is proven via contradiction, by analyzing the behavior of posterior sampling if it does not hold. The probability of failure, δ, comes entirely from Propositions 1 and 2.

To extend these results to the preference-based RL setting (the proof can be found in Appendix B.3), one can leverage asymptotic consistency of the transition dynamics, as given by Proposition 3, to obtain:

Proposition 5. *When running DPS in the preference-based RL setting, with utilities given via either the linear or logistic link functions and with posterior sampling distributions given in Eq.* (4.13)*, then with probability* $1 - \delta$*, the sampled utilities* $\tilde{\boldsymbol{r}}_{i1}, \tilde{\boldsymbol{r}}_{i2}$ *converge in distribution to their true values,* $\tilde{\boldsymbol{r}}_{i1}, \tilde{\boldsymbol{r}}_{i2} \xrightarrow{D} \overline{\boldsymbol{r}}$*, as* $i \longrightarrow \infty$.

Finally, in the preference-based RL setting, one can utilize asymptotic consistency of the transition dynamics and utility model posteriors to guarantee that the selected policies are asymptotically consistent, that is, the probability of selecting a non-optimal policy approaches zero. An analogous result holds in the generalized linear dueling bandit setting if the action space $\mathcal{A}$ is finite.

Theorem 1. *Assume that DPS is executed in the preference-based RL setting, with utilities given via either the linear or logistic link function and with posterior sampling distributions given in Eq.* (4.13)*.*

With probability $1 - \delta$*, the sampled policies* π_{i1}, π_{i2} *converge in distribution to the optimal policy,* π^**, as* $i \longrightarrow \infty$*. That is,* $P(\pi_{i1} = \pi^*) \longrightarrow 1$ *and* $P(\pi_{i2} = \pi^*) \longrightarrow 1$ *as* $i \longrightarrow \infty$*.*

Under the same conditions, with probability $1 - \delta$*, in the generalized linear dueling bandit setting with a finite action space* $\mathcal{A}$*, the sampled actions converge in distribution to the optimal actions:* $P(\boldsymbol{x}_{i1} = \boldsymbol{x}^*) \longrightarrow 1$ *and* $P(\boldsymbol{x}_{i2} = \boldsymbol{x}^*) \longrightarrow 1$ *as* $i \longrightarrow \infty$*.*

The proof of Theorem 1 is located in Appendix B.4.

Why Use an Information-Theoretic Approach to Analyze Regret?

Several existing regret analyses in the linear bandit domain (e.g., Abbasi-Yadkori, Pál, and Szepesvári, 2011; Agrawal and Goyal, 2013; Abeille and Lazaric, 2017) utilize martingale concentration properties introduced by Abbasi-Yadkori, Pál, and Szepesvári (2011). In these analyses, a key step requires sublinearly upper-bounding an expression of the form (e.g. Lemma 11 in Abbasi-Yadkori, Pál, and Szepesvári, 2011, Prop. 2 in Abeille and Lazaric, 2017):

$$\sum_{i=1}^{n} \boldsymbol{x}_i^T \left(\lambda I + \sum_{s=1}^{i-1} \boldsymbol{x}_s \boldsymbol{x}_s^T \right)^{-1} \boldsymbol{x}_i, \tag{4.14}$$

where $\lambda \geq 1$ and $\boldsymbol{x}_i \in \mathbb{R}^d$ is the observation in iteration i. In the preference-based feedback setting, however, the analogous quantity cannot always be sublinearly upper-bounded. Consider the settings defined in Chapter 3, with Bayesian linear regression credit assignment. Under preference feedback, the probability that one trajectory is preferred to another is assumed to be fully determined by the *difference* between the trajectories' total rewards: in iteration i, the algorithm receives a pair of observations $\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}$, with $\boldsymbol{x}_i := \boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}$, and a preference generated according to $P(\boldsymbol{x}_{i2} \succ \boldsymbol{x}_{i1}) = \overline{\boldsymbol{r}}^T (\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}) + \frac{1}{2}$. Thus, only *differences* between compared trajectory feature vectors yield information about the rewards. Under this assumption, one can show that applying the analogous martingale techniques yields the following variant of Eq. (4.14):

$$\sum_{i=1}^{n} \sum_{j=1}^{2} \boldsymbol{x}_{ij}^T \left(\lambda I + \sum_{s=1}^{i-1} \boldsymbol{x}_s \boldsymbol{x}_s^T \right)^{-1} \boldsymbol{x}_{ij}. \tag{4.15}$$

This difference occurs because the expression within the matrix inverse comes from the posterior, and learning occurs with respect to the observations $\boldsymbol{x}_i$, while regret is incurred with respect to $\boldsymbol{x}_{i1}$ and $\boldsymbol{x}_{i2}$. In contrast, in the non-preference-based case

considered in Agrawal and Goyal (2013), Abeille and Lazaric (2017), and Abbasi-Yadkori, Pál, and Szepesvári (2011), learning and regret both occur with respect to the same vectors $\boldsymbol{x}_i$.

To see that Eq. (4.15) does not necessarily have a sublinear upper bound, consider a deterministic MDP as a counterexample. For the regret to have a sublinear upper-bound, the probability of choosing the optimal policy must approach 1. In a fully deterministic MDP, this means that $P(\boldsymbol{x}_{i1} = \boldsymbol{x}_{i2}) \longrightarrow 1$ as $i \longrightarrow \infty$, and thus $P(\boldsymbol{x}_i = 0) \longrightarrow 1$ as $i \longrightarrow \infty$. Clearly, in this case, the inverted quantity in Eq. (4.15) acquires nonzero terms at a rate that decays in n, and so the expression in Eq. (4.15) does not have a sublinear upper bound.

Intuitively, to upper-bound Eq. (4.15), one would need the observations $\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}$ to contribute fully toward learning the rewards, rather than the contribution coming only from their difference. Notice that in the counterexample, even when the optimal policy is selected increasingly-often, corresponding to a low regret, Eq. (4.15) cannot be sublinearly bounded. In contrast, Russo and Van Roy (2016) introduce a more direct approach for quantifying the trade-off between instantaneous regret and the information gained from reward feedback, as encapsulated by the *information ratio* defined therein; the theoretical analysis of DPS is thus based upon this latter framework.

Relationship between Information Ratio and Regret

This section extends the concept of the information ratio introduced in Russo and Van Roy (2016) to analyze regret in the preference-based setting. In particular, a relationship is derived between the information ratio and Bayesian regret; this result is analogous to Proposition 1 in Russo and Van Roy (2016), which applies to the bandit setting with absolute, numerical rewards. This section considers both the preference-based generalized linear bandit setting and the preference-based RL setting with known transition dynamics. In the former case, the action space $\mathcal{A}$ is assumed to be finite.

Firstly, regarding notation, let $\boldsymbol{a}_{i1}, \boldsymbol{a}_{i2}$ denote DPS's two selections in iteration i. More specifically, in the bandit setting, $\boldsymbol{a}_{i1}, \boldsymbol{a}_{i2}$ are actions in $\mathcal{A}$, i.e., $\boldsymbol{a}_{i1} = \boldsymbol{x}_{i1}$ and $\boldsymbol{a}_{i2} = \boldsymbol{x}_{i2}$; meanwhile, in RL they are policies, such that $\boldsymbol{a}_{i1} = \pi_{i1}$ and $\boldsymbol{a}_{i2} = \pi_{i2}$. These variables are introduced to obtain a common notation between the bandit and RL settings.

Secondly, for an action or RL policy $\boldsymbol{a}$, let $u(\boldsymbol{a})$ denote the total expected utility

of executing it. In the bandit setting, $u(\boldsymbol{a}) = \overline{\boldsymbol{r}}^T \boldsymbol{a}$, while in RL, $u(\boldsymbol{a}) = \overline{\boldsymbol{r}}^T \mathbb{E}_{\boldsymbol{x} \sim \boldsymbol{a}}[\boldsymbol{x}]$, where the trajectory features $\boldsymbol{x}$ are sampled by rolling out the policy $\boldsymbol{a}$. The optimal choice $\boldsymbol{a}^*$ is the one which maximizes the utility (i.e., $\boldsymbol{a}^* = \boldsymbol{x}^*$ in the bandit case, and $\boldsymbol{a}^* = \pi^*$ in RL).

Define the algorithm's history after i iterations as $\mathcal{H}_i = \{Z_1, Z_2, \ldots, Z_i\}$, where $Z_i = (\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}, y_i)$. Note that the algorithm does not need to keep track of $\boldsymbol{a}_{i1}$ and $\boldsymbol{a}_{i2}$ in its history, as the utility posterior is updated only with respect to $(\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}, y_i)$, and in the RL setting, the transition dynamics are assumed to be known.

Analogously to Russo and Van Roy (2016), notation is established for probabilities and information-theoretic quantities when conditioning on the history $\mathcal{H}_{i-1}$. In particular, $P_i(\cdot) := P(\cdot \,|\, \mathcal{H}_{i-1})$ and $\mathbb{E}_i[\cdot] := \mathbb{E}[\cdot \,|\, \mathcal{H}_{i-1}]$. With respect to the history, the entropy of a random variable X is $H_i(X) := -\sum_x P_i(X = x) \log P_i(X = x)$. Finally, conditioned on $\mathcal{H}_{i-1}$, the mutual information of two random variables X and Y is given by $I_i(X;Y) := H_i(X) - H_i(X|Y)$.

The information ratio Γ_i in iteration i is then defined as follows:

$$\Gamma_i := \frac{\mathbb{E}_i[2u(\boldsymbol{a}^*) - u(\boldsymbol{a}_{i1}) - u(\boldsymbol{a}_{i2})]^2}{I_i(\boldsymbol{a}^*; (\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}, y_i))}. \tag{4.16}$$

Analogously to Russo and Van Roy (2016), the numerator of the information ratio is the squared expected instantaneous regret, conditioned on all prior knowledge given the history. Meanwhile, the denominator is the expected information gain with respect to the optimal action or policy $\boldsymbol{a}^*$ upon receiving data $Z_i = (\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}, y_i)$ in the current iteration. Intuitively, if the ratio Γ_i is small, then in iteration i, DPS either incurs little instantaneous regret or gains significant information about $\boldsymbol{a}^*$.

The following result, analogous to Proposition 1 in Russo and Van Roy (2016), assumes that the information ratio Γ_i has a uniform upper bound $\overline{\Gamma}$, and shows that the cumulative Bayesian regret can be upper-bounded in terms of $\overline{\Gamma}$:

Theorem 2. *In either the preference-based generalized linear bandit problem or preference-based RL with known transition dynamics, if $\Gamma_i \leq \overline{\Gamma}$ almost surely for each $i \in \{1, \ldots, N\}$, then:*

$$\mathrm{BayesReg}(N) \leq \sqrt{\overline{\Gamma} N H(\boldsymbol{a}^*)},$$

where N is the number of learning iterations, and $H(\boldsymbol{a}^*)$ is the entropy of $\boldsymbol{a}^*$.

Proof. The Bayesian regret can be upper-bounded as follows:

$$\begin{aligned}\text{BayesReg}(N) &= \mathbb{E}\left[\sum_{i=1}^{N}(2u(\boldsymbol{a}^*) - u(\boldsymbol{a}_{i1}) - u(\boldsymbol{a}_{i2}))\right] \\ &\overset{(a)}{=} \mathbb{E}\left[\sum_{i=1}^{N}\mathbb{E}_i\left[(2u(\boldsymbol{a}^*) - u(\boldsymbol{a}_{i1}) - u(\boldsymbol{a}_{i2}))\right]\right],\end{aligned}$$

where (a) follows from the Tower property of expectation and the outer expectation is taken with respect to the history. Substituting the definition of the information ratio into the latter expression yields:

$$\begin{aligned}\text{BayesReg}(N) &= \mathbb{E}\left[\sum_{i=1}^{N}\sqrt{\Gamma_i I_i(\boldsymbol{a}^*; (\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}, y_i))}\right] \\ &\leq \sqrt{\overline{\Gamma}}\sum_{i=1}^{N}\mathbb{E}\left[\sqrt{I_i(\boldsymbol{a}^*; (\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}, y_i)}\right] \\ &\overset{(a)}{\leq} \sqrt{\overline{\Gamma}}\sum_{i=1}^{N}\sqrt{\mathbb{E}[I_i(\boldsymbol{a}^*; (\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}, y_i)]} \\ &\overset{(b)}{\leq} \sqrt{\overline{\Gamma}N\sum_{i=1}^{N}\mathbb{E}[I_i(\boldsymbol{a}^*; (\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}, y_i)]},\end{aligned}$$

where (a) applies Jensen's inequality, and (b) holds via the Cauchy-Schwarz inequality. To upper-bound the sum $\sum_{i=1}^{N}\mathbb{E}[I_i(\boldsymbol{a}^*; (\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}, y_i)]$, first recall that $Z_i = (\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}, y_i)$ denotes the information acquired during iteration i. Then, each term in the summation can be written:

$$\mathbb{E}[I_i(\boldsymbol{a}^*; (\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}, y_i)] = \mathbb{E}[I_i(\boldsymbol{a}^*; Z_i)] = I(\boldsymbol{a}^*; Z_i \mid (Z_1, \ldots, Z_{i-1})),$$

where the last step follows from the definition of conditional mutual information. Therefore:

$$\begin{aligned}\sum_{i=1}^{N}\mathbb{E}[I_i(\boldsymbol{a}^*; (\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}, y_i)] = \sum_{i=1}^{N} I(\boldsymbol{a}^*; Z_i \mid (Z_1, \ldots, Z_{i-1})) &\overset{(a)}{=} I(\boldsymbol{a}^*; (Z_1, \ldots, Z_i)) \\ &\overset{(b)}{=} H(\boldsymbol{a}^*) - H(\boldsymbol{a}^* \mid Z_1, \ldots, Z_i) \overset{(c)}{\leq} H(\boldsymbol{a}^*),\end{aligned}$$

where (a) applies the chain rule for mutual information (Fact 5, Section 2.3), (b) follows from the entropy reduction form of mutual information (Fact 4), and (c) holds by non-negativity of entropy (Fact 1).

The desired result follows. □

Note that the total number of actions selected is $T = 2N$ in the dueling bandit case and $T = 2Nh$ for preference-based RL. Furthermore, the entropy $H(\boldsymbol{a}^*)$ can be upper-bounded in terms of the number of actions or number of policies (for bandits and RL, respectively) using Fact 1. For the bandit case, $H(\boldsymbol{a}^*) = H(\boldsymbol{x}^*) \leq \log |\mathcal{A}|$; under an informative prior, $H(\boldsymbol{a}^*)$ could potentially be much smaller than the upper bound $\log |\mathcal{A}|$. In the RL scenario, meanwhile, the number of deterministic policies is equal to A^{Sh}, so that $H(\boldsymbol{a}^*) = H(\pi^*) \leq \log(A^{Sh}) = Sh \log A$. This leads to the following corollary to Theorem 2, specific to preference-based RL:

Corollary 1. *In the preference-based RL setting with known transition dynamics, if* $\Gamma_i \leq \overline{\Gamma}$ *almost surely for each* $i \in \{1, \ldots, N\}$*, then:*

$$\text{BayesReg}(N) \leq \sqrt{\overline{\Gamma} N S h \log A}.$$

Moreover, since $T = 2Nh$*, the regret in terms of the total number of actions (or time-steps) is:*

$$\text{BayesReg}_{RL}(T) \leq \sqrt{\frac{\overline{\Gamma} T S \log A}{2}},$$

where $\text{BayesReg}_{RL}(T)$ *is in terms of the total number of time-steps, while* $\text{BayesReg}(N)$ *is in terms of the number of learning iterations.*

Thus, Bayesian regret can be upper-bounded by showing that the information ratio is upper-bounded, and then applying either Theorem 2 or Corollary 1.

Remark 6. *Importantly, the relationship in Theorem 2 between the Bayesian regret and information ratio only applies under an exact posterior (i.e., the posterior is an exact product of a prior and likelihood); however, not all of the posterior sampling distributions considered in Section 4.3 satisfy this criterion. For instance, while the* $\beta_n(\delta)$ *factors in Eq.* (4.13) *were useful for deriving asymptotic consistency results, they yield a sampling distribution that is no longer an exact posterior. To see that a true posterior is indeed required, consider the following inequality, used in proving Theorem 2:*

$$\sum_{i=1}^{N} \mathbb{E}[I_i(\boldsymbol{a}^*; (\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}, y_i)] \leq H(\boldsymbol{a}^*).$$

This inequality uniformly upper-bounds the sum of the information gains across all iterations: intuitively, the total amount of information gained cannot exceed the initial uncertainty about the optimal policy or action. Yet, the factor $\beta_n(\delta)$ *increases*

in n, so that multiplying the posterior covariance by $\beta_n(\delta)^2$ injects uncertainty into the model posterior indefinitely. In such a case, the sum of the information gains could not have a finite upper bound, thus the above inequality could not hold.

Estimating the Information Ratio Empirically

This thesis conjectures an upper bound to the information ratio Γ_i for the linear link function by empirically estimating Γ_i's values. This subsection discusses the procedure used to estimate the information ratio in simulation. The process is first discussed for the preference-based bandit setting, and then extended to RL with known transition dynamics.

Information ratio estimation for generalized linear dueling bandits. In the preference-based generalized linear bandit setting, the information ratio can equivalently be written as:

$$\Gamma_i := \frac{\mathbb{E}_i[2u(\boldsymbol{x}^*) - u(\boldsymbol{x}_{i1}) - u(\boldsymbol{x}_{i2})]^2}{I_i(\boldsymbol{x}^*; (\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}, y_i))}.$$

The following lemma, inspired by similar analyses in Russo and Van Roy (2016) and Russo and Van Roy (2014a) for settings with numerical rewards, expresses both the numerator and denominator of Γ_i in forms that are more straightforward to calculate empirically.

Lemma 1. *When running DPS in the preference-based generalized linear bandit setting, the following two statements hold:*

$$\mathbb{E}_i[2u(\boldsymbol{x}^*) - u(\boldsymbol{x}_{i1}) - u(\boldsymbol{x}_{i2})] = 2\sum_{\boldsymbol{x}\in\mathcal{A}} P_i(\boldsymbol{x}^* = \boldsymbol{x})\boldsymbol{x}^T\left[\mathbb{E}_i[\overline{\boldsymbol{r}} \mid \boldsymbol{x}^* = \boldsymbol{x}] - \mathbb{E}_i[\overline{\boldsymbol{r}}]\right], \text{ and}$$

$$I_i(\boldsymbol{x}^*; (\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}, y_i)) \geq 2\sum_{\boldsymbol{x},\boldsymbol{x}'\in\mathcal{A}} P_i(\boldsymbol{x}^* = \boldsymbol{x})P_i(\boldsymbol{x}^* = \boldsymbol{x}')(\boldsymbol{x} - \boldsymbol{x}')^T$$
$$* \left\{\sum_{\boldsymbol{x}''\in\mathcal{A}} P_i(\boldsymbol{x}^* = \boldsymbol{x}'')(\mathbb{E}_i[\overline{\boldsymbol{r}} \mid \boldsymbol{x}^* = \boldsymbol{x}''] - \mathbb{E}_i[\overline{\boldsymbol{r}}])(\mathbb{E}_i[\overline{\boldsymbol{r}} \mid \boldsymbol{x}^* = \boldsymbol{x}''] - \mathbb{E}_i[\overline{\boldsymbol{r}}])^T\right\}(\boldsymbol{x} - \boldsymbol{x}').$$

Proof. To prove the first statement:

$$\begin{aligned}
\mathbb{E}_i[2u(\boldsymbol{x}^*) - u(\boldsymbol{x}_{i1}) - u(\boldsymbol{x}_{i2})] &\overset{(a)}{=} 2\mathbb{E}_i[u(\boldsymbol{x}^*) - u(\boldsymbol{x}_{i1})] \\
&= 2\sum_{\boldsymbol{x}\in\mathcal{A}} \Big[P_i(\boldsymbol{x}^* = \boldsymbol{x})\mathbb{E}_i[u(\boldsymbol{x}^*) \mid \boldsymbol{x}^* = \boldsymbol{x}] - P_i(\boldsymbol{x}_{i1} = \boldsymbol{x})\mathbb{E}_i[u(\boldsymbol{x}_{i1}) \mid \boldsymbol{x}_{i1} = \boldsymbol{x}]\Big] \\
&\overset{(b)}{=} 2\sum_{\boldsymbol{x}\in\mathcal{A}} P_i(\boldsymbol{x}^* = \boldsymbol{x})\Big[\mathbb{E}_i[u(\boldsymbol{x}) \mid \boldsymbol{x}^* = \boldsymbol{x}] - \mathbb{E}_i[u(\boldsymbol{x})]\Big] \\
&\overset{(c)}{=} 2\sum_{\boldsymbol{x}\in\mathcal{A}} P_i(\boldsymbol{x}^* = \boldsymbol{x})\Big[\mathbb{E}_i[\boldsymbol{x}^T\overline{\boldsymbol{r}} \mid \boldsymbol{x}^* = \boldsymbol{x}] - \mathbb{E}_i[\boldsymbol{x}^T\overline{\boldsymbol{r}}]\Big] \\
&= 2\sum_{\boldsymbol{x}\in\mathcal{A}} P_i(\boldsymbol{x}^* = \boldsymbol{x})\boldsymbol{x}^T\Big[\mathbb{E}_i[\overline{\boldsymbol{r}} \mid \boldsymbol{x}^* = \boldsymbol{x}] - \mathbb{E}_i[\overline{\boldsymbol{r}}]\Big],
\end{aligned}$$

where (a) holds because $\boldsymbol{x}_{i1}$ and $\boldsymbol{x}_{i2}$ are independent samples from the model posterior, and thus are identically-distributed; (b) holds because posterior sampling selects each action according to its posterior probability of being optimal, so that $P_i(\boldsymbol{x}^* = \boldsymbol{x}) = P_i(\boldsymbol{x}_{i1} = \boldsymbol{x})$; and (c) applies the definition of utility for generalized linear dueling bandits.

Turning to the second statement, it is notationally-convenient to define a function $y : \mathcal{A} \times \mathcal{A} \to \left\{-\frac{1}{2}, \frac{1}{2}\right\}$, where for each pair of actions $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{A}$, $y(\boldsymbol{x}, \boldsymbol{x}')$ is the outcome of a preference query between $\boldsymbol{x}$ and $\boldsymbol{x}'$:

$$\begin{cases}
y(\boldsymbol{x}, \boldsymbol{x}') = \frac{1}{2} & \text{with probability } P(\boldsymbol{x} \succ \boldsymbol{x}') \\
y(\boldsymbol{x}, \boldsymbol{x}') = -\frac{1}{2} & \text{with probability } P(\boldsymbol{x} \prec \boldsymbol{x}').
\end{cases}$$

Equipped with this definition, we can see that:

$$\begin{aligned}
I_i(\boldsymbol{x}^*; (\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}, y_i)) &\overset{(a)}{=} I_i(\boldsymbol{x}^*; \boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}) + I_i(\boldsymbol{x}^*; y_i \mid \boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}) \overset{(b)}{=} I_i(\boldsymbol{x}^*; y_i \mid \boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}) \\
&\overset{(c)}{=} \sum_{\boldsymbol{x},\boldsymbol{x}'\in\mathcal{A}} P_i(\boldsymbol{x}_{i1} = \boldsymbol{x}')P_i(\boldsymbol{x}_{i2} = \boldsymbol{x})I_i(\boldsymbol{x}^*; y_i \mid \boldsymbol{x}_{i2} - \boldsymbol{x}_{i1} = \boldsymbol{x} - \boldsymbol{x}') \\
&\overset{(d)}{=} \sum_{\boldsymbol{x},\boldsymbol{x}'\in\mathcal{A}} P_i(\boldsymbol{x}^* = \boldsymbol{x}')P_i(\boldsymbol{x}^* = \boldsymbol{x})I_i(\boldsymbol{x}^*; y(\boldsymbol{x}, \boldsymbol{x}')) \\
&\overset{(e)}{=} \sum_{\boldsymbol{x},\boldsymbol{x}'\in\mathcal{A}} P_i(\boldsymbol{x}^* = \boldsymbol{x}')P_i(\boldsymbol{x}^* = \boldsymbol{x}) \\
&\quad * \sum_{\boldsymbol{x}''\in\mathcal{A}} P_i(\boldsymbol{x}^* = \boldsymbol{x}'')D(P_i(y(\boldsymbol{x}, \boldsymbol{x}') \mid \boldsymbol{x}^* = \boldsymbol{x}'') \mid\mid P_i(y(\boldsymbol{x}, \boldsymbol{x}'))),
\end{aligned}$$

where (a) applies the chain rule for mutual information (Fact 5), and (b) uses that $I_i(\boldsymbol{x}^*; \boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}) = 0$, which holds because conditioned on the history, $\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}$ are independent samples from the posterior of $\boldsymbol{x}^*$ that yield no new information about

$\boldsymbol{x}^*$. Then, (c) holds by definition of conditional mutual information, and (d) holds because posterior sampling selects each action according to its probability of being optimal under the posterior, so that $P_i(\boldsymbol{x}^* = \boldsymbol{x}) = P_i(\boldsymbol{x}_{i1} = \boldsymbol{x})$. Finally, (e) applies Fact 6 from Section 2.3 (which is also Fact 6 in Russo and Van Roy, 2016).

Next, to lower-bound the Kullback-Leibler divergence in terms of utilities, one can apply Fact 7 from Section 2.3 (Fact 9 in Russo and Van Roy, 2016):

$$\begin{aligned}
I_i(\boldsymbol{x}^*; (\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}, y_i)) &\overset{(a)}{\geq} 2 \sum_{\boldsymbol{x},\boldsymbol{x}',\boldsymbol{x}'' \in \mathcal{A}} P_i(\boldsymbol{x}^* = \boldsymbol{x}) P_i(\boldsymbol{x}^* = \boldsymbol{x}') P_i(\boldsymbol{x}^* = \boldsymbol{x}'') \\
&\quad * \left[\mathbb{E}\left[y(\boldsymbol{x}, \boldsymbol{x}') \mid \boldsymbol{x}^* = \boldsymbol{x}''\right] - \mathbb{E}\left[y(\boldsymbol{x}, \boldsymbol{x}')\right]\right]^2 \\
&\overset{(b)}{=} 2 \sum_{\boldsymbol{x},\boldsymbol{x}',\boldsymbol{x}'' \in \mathcal{A}} P_i(\boldsymbol{x}^* = \boldsymbol{x}) P_i(\boldsymbol{x}^* = \boldsymbol{x}') P_i(\boldsymbol{x}^* = \boldsymbol{x}'') \\
&\quad * \left[\mathbb{E}[(\boldsymbol{x} - \boldsymbol{x}')^T \overline{\boldsymbol{r}} \mid \boldsymbol{x}^* = \boldsymbol{x}''] - \mathbb{E}[(\boldsymbol{x} - \boldsymbol{x}')^T \overline{\boldsymbol{r}}]\right]^2 \\
&= 2 \sum_{\boldsymbol{x},\boldsymbol{x}',\boldsymbol{x}'' \in \mathcal{A}} P_i(\boldsymbol{x}^* = \boldsymbol{x}) P_i(\boldsymbol{x}^* = \boldsymbol{x}') P_i(\boldsymbol{x}^* = \boldsymbol{x}'') \\
&\quad * \left[(\boldsymbol{x} - \boldsymbol{x}')^T \left(\mathbb{E}[\overline{\boldsymbol{r}} \mid \boldsymbol{x}^* = \boldsymbol{x}''] - \mathbb{E}[\overline{\boldsymbol{r}}]\right)\right]^2,
\end{aligned}$$

where (a) applies Fact 7 and (b) applies the definition of the linear link function.

It is straightforward to rearrange the final expression into the form given in the statement of the lemma. □

Similarly to Russo and Van Roy (2014a), the information ratio simulations in fact estimate an upper bound to Γ_i, which is obtained by replacing its denominator in Eq. (4.16) by the lower bound obtained in Lemma 1.

To estimate the quantities in Lemma 1, it is helpful to define the following notation. Firstly, let $\boldsymbol{\mu}_i$ be the expected value of $\overline{\boldsymbol{r}}$ at iteration i:

$$\boldsymbol{\mu}_i := \mathbb{E}_i[\overline{\boldsymbol{r}}].$$

Similarly, $\boldsymbol{\mu}_i^{(\boldsymbol{x})}$ is the expected value of $\overline{\boldsymbol{r}}$ at iteration i when conditioned on $\boldsymbol{x}^* = \boldsymbol{x}$:

$$\boldsymbol{\mu}_i^{(\boldsymbol{x})} := \mathbb{E}_i[\overline{\boldsymbol{r}} \mid \boldsymbol{x}^* = \boldsymbol{x}].$$

The bracketed matrix in the statement of Lemma 1 is denoted by L_i:

$$L_i := \sum_{\boldsymbol{x} \in \mathcal{A}} P_i(\boldsymbol{x}^* = \boldsymbol{x}) (\mathbb{E}_i[\overline{\boldsymbol{r}} \mid \boldsymbol{x}^* = \boldsymbol{x}] - \mathbb{E}_i[\overline{\boldsymbol{r}}]) (\mathbb{E}_i[\overline{\boldsymbol{r}} \mid \boldsymbol{x}^* = \boldsymbol{x}] - \mathbb{E}_i[\overline{\boldsymbol{r}}])^T.$$

Algorithm 11 Empirically estimating the information ratio in the linear dueling bandit setting

1: **Input**: $\mathcal{A}$ = action space, ν = posterior distribution over $\overline{\boldsymbol{r}}$, M = number of samples to draw from the posterior
2: $\boldsymbol{r}_1, \ldots, \boldsymbol{r}_M \sim \nu$ ▷ Draw samples from utility posterior
3: $\hat{\boldsymbol{\mu}} \longleftarrow \frac{1}{M} \sum_m \boldsymbol{r}_m$ ▷ Estimate $\boldsymbol{\mu}_i = \mathbb{E}_i[\overline{\boldsymbol{r}}]$
4: $\hat{\Theta}_{\boldsymbol{x}} \longleftarrow \{m : \boldsymbol{x}^T \boldsymbol{r}_m = \max_{\boldsymbol{x}' \in \mathcal{A}} (\boldsymbol{x}'^T \boldsymbol{r}_m)\} \; \forall \boldsymbol{x} \in \mathcal{A}$ ▷ Samples $\boldsymbol{r}_i$ for which $\boldsymbol{x}$ is optimal
5: $p^*(\boldsymbol{x}) \longleftarrow \frac{|\hat{\Theta}_{\boldsymbol{x}}|}{M} \; \forall \boldsymbol{x} \in \mathcal{A}$ ▷ Empirical posterior probability of actions being optimal
6: $\hat{\boldsymbol{\mu}}^{(\boldsymbol{x})} \longleftarrow \frac{1}{|\hat{\Theta}_{\boldsymbol{x}}|} \sum_{\boldsymbol{r} \in \hat{\Theta}_{\boldsymbol{x}}} \boldsymbol{r} \; \forall \boldsymbol{x} \in \mathcal{A}$ ▷ Estimate $\boldsymbol{\mu}_i^{(\boldsymbol{x})} = \mathbb{E}_i[\overline{\boldsymbol{r}} \mid \boldsymbol{x}^* = \boldsymbol{x}]$
7: $\hat{L} \longleftarrow \sum_{\boldsymbol{x} \in \mathcal{A}} p^*(\boldsymbol{x})(\hat{\boldsymbol{\mu}}^{(\boldsymbol{x})} - \hat{\boldsymbol{\mu}})(\hat{\boldsymbol{\mu}}^{(\boldsymbol{x})} - \hat{\boldsymbol{\mu}})^T$ ▷ Estimate L_i
8: $r^* \longleftarrow \sum_{\boldsymbol{x} \in \mathcal{A}} p^*(\boldsymbol{x}) \boldsymbol{x}^T \hat{\boldsymbol{\mu}}^{(\boldsymbol{x})}$ ▷ Estimate posterior reward of optimal action
9: $\Delta_{\boldsymbol{x}_1, \boldsymbol{x}_2} \longleftarrow 2r^* - (\boldsymbol{x}_1 + \boldsymbol{x}_2)^T \hat{\boldsymbol{\mu}} \; \forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{A}$
10: $\Delta \longleftarrow \sum_{\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{A}} p^*(\boldsymbol{x}_1) p^*(\boldsymbol{x}_2) \Delta_{\boldsymbol{x}_1, \boldsymbol{x}_2}$ ▷ Expected instantaneous regret
11: $v_{\boldsymbol{x}_1, \boldsymbol{x}_2} \longleftarrow (\boldsymbol{x}_2 - \boldsymbol{x}_1)^T \hat{L} (\boldsymbol{x}_2 - \boldsymbol{x}_1) \; \forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{A}$
12: $v \longleftarrow \sum_{\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{A}} p^*(\boldsymbol{x}_1) p^*(\boldsymbol{x}_2) v_{\boldsymbol{x}_1, \boldsymbol{x}_2}$ ▷ Information gain lower bound
13: $\hat{\Gamma} \longleftarrow \frac{\Delta^2}{2v}$ ▷ Estimated information ratio

The matrix L_i can also be written in the following, more compact form:

$$L_i = \mathbb{E}_i\Big[(\mathbb{E}_i[\overline{\boldsymbol{r}} \mid \boldsymbol{x}^*] - \mathbb{E}_i[\overline{\boldsymbol{r}}])(\mathbb{E}_i[\overline{\boldsymbol{r}} \mid \boldsymbol{x}^*] - \mathbb{E}_i[\overline{\boldsymbol{r}}])^T\Big] = \mathbb{E}_i\Big[(\boldsymbol{\mu}_i^{\boldsymbol{x}^*} - \boldsymbol{\mu}_i)(\boldsymbol{\mu}_i^{\boldsymbol{x}^*} - \boldsymbol{\mu}_i)^T\Big],$$

where the outer expectation is taken with respect to the optimal action, $\boldsymbol{x}^*$.

Algorithm 11 outlines the procedure for estimating the information ratio in the linear dueling bandit setting, given a posterior distribution $P(\boldsymbol{r} \mid \mathcal{D})$ over $\overline{\boldsymbol{r}}$ from which samples can be drawn. This procedure extends the one given in Russo and Van Roy (2014a) in order to handle relative preference feedback. First, in Line 2, the algorithm draws M samples from the model posterior over $\overline{\boldsymbol{r}}$. These are then averaged to obtain an estimate $\hat{\boldsymbol{\mu}}$ of $\boldsymbol{\mu}_i$ (Line 3). By dividing the posterior samples of $\overline{\boldsymbol{r}}$ into groups $\Theta_{\boldsymbol{x}}$ depending on which optimal action they induce, one can then estimate each action's posterior probability $p^*(\boldsymbol{x})$ of being optimal (Lines 4-5). Note that the values of $p^*(\boldsymbol{x})$ also give the probability that DPS selects each action. The procedure then uses these quantities to estimate $\boldsymbol{\mu}_i^{(\boldsymbol{x})}$ (Line 6) and L_i (Line 7). From these, the instantaneous regret and information gain lower bound are both estimated, using the forms given in Lemma 1. Finally, the information ratio is estimated; in fact, the value calculated in Line 13 upper bounds the true information ratio Γ_i, due to the inequality in Lemma 1.

Information ratio estimation for preference-based RL with known dynamics. To extend Algorithm 11 to preference-based RL with known transition dynamics, one must replace the action space $\mathcal{A}$ with the set of deterministic policies Π, where $|\Pi| = A^{Sh}$

(recall that S is the number of states in the MDP and h is the episode length). While the bandit case associates each action in $\mathcal{A}$ with a d-dimensional vector, the present setting analogously associates each policy π with a known d-dimensional *occupancy* vector $\boldsymbol{x}_\pi$ (recall that $d = SA$): in $\boldsymbol{x}_\pi$, each element is the expected number of times that a particular state-action pair is visited under π. More formally, the occupancy vector associated with π is given by,

$$\boldsymbol{x}_\pi = \mathbb{E}_{\boldsymbol{x}\sim\pi}[\boldsymbol{x}],$$

where $\boldsymbol{x}$ is a trajectory obtained by executing the policy π, and the expectation is taken with respect to the known MDP transition dynamics and initial state distribution. This subsection first discusses computation of the occupancy vectors, and then shows that to extend Algorithm 11 to RL, one must simply replace the action space $\mathcal{A}$ with Π, and replace the actions in $\mathcal{A}$ with the occupancy vectors associated with the policies: $\{\boldsymbol{x}_\pi \mid \pi \in \Pi\}$. Each policy π can therefore be viewed as a meta-action leading to a known distribution over trajectory feature vectors, whose expectation is given by the occupancy vector $\boldsymbol{x}_\pi$.

The occupancy vector for a given deterministic policy π can be calculated as follows. First, define $\boldsymbol{s}_\pi(t) \in \mathbb{R}^S$ as a vector containing the probabilities of being in each state at time-step t in the episode, where $t \in \{1, \ldots, h\}$. Further, the state at time t is denoted by $s_t \in \{1, \ldots, S\}$. Then, for $t = 1$, $\boldsymbol{s}_\pi(1) = [p_0(1), \ldots, p_0(S)]^T$, where $p_0(i)$ is the probability of starting in state i, as given by the initial state probabilities (assumed to be known). Also, define $\boldsymbol{x}_\pi(t) \in \mathbb{R}^d$ as the probability of being in each state-action pair at time-step t. Then, for each $t = 1, 2, \ldots, h$:

1. Obtain $\boldsymbol{x}_\pi(t)$ from $\boldsymbol{s}_\pi(t)$: for each state-action pair (s, a), the corresponding element of $\boldsymbol{x}_\pi(t)$ is equal to $[\boldsymbol{s}_\pi(t)]_s \mathbb{I}_{[\pi(s,t)=a]}$, that is, the probability that the agent visits state s at time t is multiplied by 0 or 1 depending on whether the deterministic policy π takes action a in state s at time t.

2. Define the matrix $P^{(t)}$ such that: $[P^{(t)}]_{ij} = P(s_{t+1} = j \mid s = i, a = \pi(i, t))$.

3. Calculate the probability of being in each state at the next time-step: $\boldsymbol{s}_\pi(t+1) = P^{(t)} \boldsymbol{s}_\pi(t)$.

Finally, the expected number of visits to each state-action pair over the entire episode is given by summing the probabilities of encountering each state-action pair at each

time-step:

$$\boldsymbol{x}_\pi = \sum_{t=1}^{h} \boldsymbol{x}_\pi(t).$$

These occupancy vectors $\boldsymbol{x}_\pi$ can be pre-computed for each policy $\pi \in \Pi$. Below, Lemma 2 will show that these vectors can be used in place of action vectors in Algorithm 11. Regarding implementation, it is convenient to index the policies from 1 to $|\Pi| = A^{Sh}$, and to convert between these policy indices and the policies' parameters. To do so efficiently, one can represent each policy's parameters as a vector in $\{1, \ldots, A\}^{Sh}$, in which each element is the action taken in a specific state and time-step. Then, one can treat each policy vector in $\{1, \ldots, A\}^{Sh}$ as a base-A integer, and convert between bases A and 10 as needed to interchange policy parameters with policy indices.

Finally, the following lemma is analogous to Lemma 1 for the generalized linear dueling bandit setting, but is adapted to the preference-based RL setting. Though the proof of Lemma 2 is similar to that of Lemma 1, it requires several additional steps. Recall that the utility of a policy π, given the state-action utilities $\overline{\boldsymbol{r}}$ and known dynamics, is defined as: $u(\pi) := \overline{\boldsymbol{r}}^T \mathbb{E}_{\boldsymbol{x} \sim \pi}[\boldsymbol{x}] = \overline{\boldsymbol{r}}^T \boldsymbol{x}_\pi$.

Lemma 2. *When running DPS in preference-based RL with known transition dynamics and initial state probabilities, the following two statements hold:*

$$\mathbb{E}_i[2u(\pi^*) - u(\pi_{i1}) - u(\pi_{i2})] = 2 \sum_{\pi \in \Pi} P_i(\pi^* = \pi) \boldsymbol{x}_\pi^T \left[\mathbb{E}_i[\overline{\boldsymbol{r}} \mid \pi^* = \pi] - \mathbb{E}_i[\overline{\boldsymbol{r}}]\right], \text{ and}$$

$$I_i(\pi^*; (\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}, y_i)) \geq 2 \sum_{\pi, \pi' \in \Pi} P_i(\pi^* = \pi) P_i(\pi^* = \pi') (\boldsymbol{x}_\pi - \boldsymbol{x}_{\pi'})^T$$
$$* \left\{ \sum_{\pi'' \in \Pi} P_i(\pi^* = \pi'') (\mathbb{E}_i[\overline{\boldsymbol{r}} \mid \pi^* = \pi''] - \mathbb{E}_i[\overline{\boldsymbol{r}}]) (\mathbb{E}_i[\overline{\boldsymbol{r}} \mid \pi^* = \pi''] - \mathbb{E}_i[\overline{\boldsymbol{r}}])^T \right\} (\boldsymbol{x}_\pi - \boldsymbol{x}_{\pi'}).$$

Proof. The first statement is proven as follows:

$$\begin{aligned}
\mathbb{E}_i[2u(\pi^*) - u(\pi_{i1}) - u(\pi_{i2})] &\overset{(a)}{=} 2\mathbb{E}_i[u(\pi^*) - u(\pi_{i1})] \\
&= 2 \sum_{\pi \in \Pi} \left[P_i(\pi^* = \pi) \mathbb{E}_i[u(\pi^*) \mid \pi^* = \pi] - P_i(\pi_{i1} = \pi) \mathbb{E}_i[u(\pi_{i1}) \mid \pi_{i1} = \pi]\right] \\
&\overset{(b)}{=} 2 \sum_{\pi \in \Pi} P_i(\pi^* = \pi) \left[\mathbb{E}_i[u(\pi) \mid \pi^* = \pi] - \mathbb{E}_i[u(\pi)]\right] \\
&\overset{(c)}{=} 2 \sum_{\pi \in \Pi} P_i(\pi^* = \pi) \left[\mathbb{E}_i[\boldsymbol{x}_\pi^T \overline{\boldsymbol{r}} \mid \pi^* = \pi] - \mathbb{E}_i[\boldsymbol{x}_\pi^T \overline{\boldsymbol{r}}]\right] \\
&\overset{(d)}{=} 2 \sum_{\pi \in \Pi} P_i(\pi^* = \pi) \boldsymbol{x}_\pi^T \left[\mathbb{E}_i[\overline{\boldsymbol{r}} \mid \pi^* = \pi] - \mathbb{E}_i[\overline{\boldsymbol{r}}]\right],
\end{aligned}$$

where (a) holds because π_{i1} and π_{i2} are both sampled from the same posterior, and thus are identically-distributed; (b) holds because posterior sampling selects each policy according to its posterior probability of being optimal, so that $P_i(\pi^* = \pi) = P_i(\pi_{i1} = \pi)$; and (c) applies the definition of the utility under the linear link function. Finally, (d) holds because the transition dynamics are known.

To prove the second statement, denote by $\Upsilon \subset \mathbb{R}^d$ the set of all possible length-h trajectory feature vectors. Analogously to the bandit case, it is notationally-convenient to define a function $y : \Upsilon \times \Upsilon \to \left\{-\frac{1}{2}, \frac{1}{2}\right\}$, where for each pair of trajectory feature vectors $\boldsymbol{x}, \boldsymbol{x}' \in \Upsilon$, $y(\boldsymbol{x}, \boldsymbol{x}')$ is the outcome of a preference query between $\boldsymbol{x}$ and $\boldsymbol{x}'$:

$$\begin{cases} y(\boldsymbol{x}, \boldsymbol{x}') = \frac{1}{2} & \text{with probability } P(\boldsymbol{x} \succ \boldsymbol{x}') \\ y(\boldsymbol{x}, \boldsymbol{x}') = -\frac{1}{2} & \text{with probability } P(\boldsymbol{x} \prec \boldsymbol{x}'). \end{cases}$$

Using these definitions, one can see that:

$$\begin{aligned}
I_i(\pi^*; (\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}, y_i)) &\overset{(a)}{=} I_i(\pi^*; \boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}) + I_i(\pi^*; y_i \mid \boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}) \overset{(b)}{=} I_i(\pi^*; y_i \mid \boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}) \\
&\overset{(c)}{=} \sum_{\boldsymbol{x}, \boldsymbol{x}' \in \Upsilon} P_i(\boldsymbol{x}_{i1} = \boldsymbol{x}') P_i(\boldsymbol{x}_{i2} = \boldsymbol{x}) I_i(\pi^*; y_i \mid \boldsymbol{x}_{i2} - \boldsymbol{x}_{i1} = \boldsymbol{x} - \boldsymbol{x}') \\
&= \sum_{\boldsymbol{x}, \boldsymbol{x}' \in \Upsilon} P_i(\boldsymbol{x}_{i1} = \boldsymbol{x}') P_i(\boldsymbol{x}_{i2} = \boldsymbol{x}) I_i(\pi^*; y(\boldsymbol{x}, \boldsymbol{x}')) \\
&= \sum_{\pi, \pi' \in \Pi} \sum_{\boldsymbol{x}, \boldsymbol{x}' \in \Upsilon} P_i(\pi_{i1} = \pi') P_i(\pi_{i2} = \pi) \\
&\qquad * P_i(\boldsymbol{x}_{i1} = \boldsymbol{x}' \mid \pi_{i1} = \pi') P_i(\boldsymbol{x}_{i2} = \boldsymbol{x} \mid \pi_{i2} = \pi) I_i(\pi^*; y(\boldsymbol{x}, \boldsymbol{x}')) \\
&\overset{(d)}{=} \sum_{\pi, \pi' \in \Pi} P_i(\pi^* = \pi') P_i(\pi^* = \pi) \\
&\qquad * \sum_{\boldsymbol{x}, \boldsymbol{x}' \in \Upsilon} P(\boldsymbol{x}_{i1} = \boldsymbol{x}' \mid \pi_{i1} = \pi') P(\boldsymbol{x}_{i2} = \boldsymbol{x} \mid \pi_{i2} = \pi) I_i(\pi^*; y(\boldsymbol{x}, \boldsymbol{x}')) \\
&\overset{(e)}{=} \sum_{\pi, \pi' \in \Pi} P_i(\pi^* = \pi') P_i(\pi^* = \pi) \sum_{\boldsymbol{x}, \boldsymbol{x}' \in \Upsilon} P(\boldsymbol{x}_{i1} = \boldsymbol{x}' \mid \pi_{i1} = \pi') P(\boldsymbol{x}_{i2} = \boldsymbol{x} \mid \pi_{i2} = \pi) \\
&\qquad * \sum_{\pi'' \in \Pi} P_i(\pi^* = \pi'') D(P_i(y(\boldsymbol{x}, \boldsymbol{x}') \mid \pi^* = \pi'') \mid\mid P_i(y(\boldsymbol{x}, \boldsymbol{x}'))) \\
&\overset{(f)}{\geq} 2 \sum_{\pi, \pi', \pi'' \in \Pi} P_i(\pi^* = \pi') P_i(\pi^* = \pi) P_i(\pi^* = \pi'') \\
&\qquad * \sum_{\boldsymbol{x}, \boldsymbol{x}' \in \Upsilon} P(\boldsymbol{x}_{i1} = \boldsymbol{x}' \mid \pi_{i1} = \pi') P(\boldsymbol{x}_{i2} = \boldsymbol{x} \mid \pi_{i2} = \pi) \\
&\qquad * \left[\mathbb{E}_i\left[y(\boldsymbol{x}, \boldsymbol{x}') \mid \pi^* = \pi''\right] - \mathbb{E}_i\left[y(\boldsymbol{x}, \boldsymbol{x}')\right]\right]^2
\end{aligned}$$

$$
\begin{aligned}
&\overset{(g)}{=} 2 \sum_{\pi,\pi',\pi''\in\Pi} P_i(\pi^* = \pi')P_i(\pi^* = \pi)P_i(\pi^* = \pi'') \\
&\qquad * \sum_{\boldsymbol{x},\boldsymbol{x}'\in\Upsilon} P(\boldsymbol{x}_{i1} = \boldsymbol{x}' \mid \pi_{i1} = \pi')P(\boldsymbol{x}_{i2} = \boldsymbol{x} \mid \pi_{i2} = \pi) \\
&\qquad * \left[\mathbb{E}_i\left[(\boldsymbol{x} - \boldsymbol{x}')^T\overline{\boldsymbol{r}} \mid \pi^* = \pi''\right] - \mathbb{E}_i\left[(\boldsymbol{x} - \boldsymbol{x}')^T\overline{\boldsymbol{r}}\right]\right]^2 \\
&\overset{(h)}{=} 2 \sum_{\pi,\pi',\pi''\in\Pi} P_i(\pi^* = \pi')P_i(\pi^* = \pi)P_i(\pi^* = \pi'') \\
&\qquad * \sum_{\boldsymbol{x},\boldsymbol{x}'\in\Upsilon} P(\boldsymbol{x}_{i1} = \boldsymbol{x}' \mid \pi_{i1} = \pi')P(\boldsymbol{x}_{i2} = \boldsymbol{x} \mid \pi_{i2} = \pi) \\
&\qquad * \left[(\boldsymbol{x} - \boldsymbol{x}')^T\left(\mathbb{E}_i\left[\overline{\boldsymbol{r}} \mid \pi^* = \pi''\right] - \mathbb{E}_i\left[\overline{\boldsymbol{r}}\right]\right)\right]^2 \\
&= 2 \sum_{\pi,\pi',\pi''\in\Pi} P_i(\pi^* = \pi')P_i(\pi^* = \pi)P_i(\pi^* = \pi'') \\
&\qquad * \mathbb{E}_{\boldsymbol{x}\sim\pi,\boldsymbol{x}'\sim\pi'}\left[\left[(\boldsymbol{x} - \boldsymbol{x}')^T\left(\mathbb{E}_i\left[\overline{\boldsymbol{r}} \mid \pi^* = \pi''\right] - \mathbb{E}_i\left[\overline{\boldsymbol{r}}\right]\right)\right]^2\right] \\
&\overset{(i)}{\geq} 2 \sum_{\pi,\pi',\pi''\in\Pi} P_i(\pi^* = \pi')P_i(\pi^* = \pi)P_i(\pi^* = \pi'') \\
&\qquad * \left[\mathbb{E}_{\boldsymbol{x}\sim\pi,\boldsymbol{x}'\sim\pi'}\left[(\boldsymbol{x} - \boldsymbol{x}')^T\left(\mathbb{E}_i\left[\overline{\boldsymbol{r}} \mid \pi^* = \pi''\right] - \mathbb{E}_i\left[\overline{\boldsymbol{r}}\right]\right)\right]\right]^2 \\
&= 2 \sum_{\pi,\pi',\pi''\in\Pi} P_i(\pi^* = \pi')P_i(\pi^* = \pi)P_i(\pi^* = \pi'') \\
&\qquad * \left[(\boldsymbol{x}_\pi - \boldsymbol{x}_{\pi'})^T\left(\mathbb{E}_i\left[\overline{\boldsymbol{r}} \mid \pi^* = \pi''\right] - \mathbb{E}_i\left[\overline{\boldsymbol{r}}\right]\right)\right]^2,
\end{aligned}
$$

where (a) applies the chain rule for mutual information (Fact 5), and (b) uses that $I_i(\pi^*; \boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}) = 0$, which holds because $\boldsymbol{x}_{i1}$ and $\boldsymbol{x}_{i2}$ are determined by independently sampling two policies from the posterior of π^* and rolling them out (via known transition dynamics) to obtain two trajectories. Then, (c) holds by definition of conditional mutual information, and (d) holds because posterior sampling selects each policy according to its posterior probability of optimality, so that $P_i(\pi^* = \pi) = P_i(\pi_{i1} = \pi)$ for each π, and further uses that the probabilistic mapping from policies to trajectories is known. Next, (e) applies Fact 6 from Section 2.3 (which is Fact 6 in Russo and Van Roy, 2016), and step (f) applies Fact 7 from Section 2.3 (Fact 9 in Russo and Van Roy, 2016) to lower-bound the Kullback-Leibler divergence in terms of utilities. Finally, (g) applies the definition of the linear link function, (h) utilizes that the transition dynamics are known, and (i) holds via Jensen's inequality.

It is straightforward to rearrange the final expression into the form given in the statement of the lemma. □

Finally, Algorithm 12 details the procedure for estimating the information ratio for RL with known dynamics. It is analogous to the procedure for the linear dueling bandit setting, given by Algorithm 11.

Algorithm 12 Empirically estimating the information ratio in RL with known dynamics

Input: Π = policy space, ν = posterior distribution over $\overline{\boldsymbol{r}}$, M = number of samples to draw from the posterior, $\{\boldsymbol{x}_\pi \mid \pi \in \Pi\}$ = set of occupancy vectors for each policy in Π

$\boldsymbol{r}_1, \dots, \boldsymbol{r}_M \sim \nu$ ▷ Draw samples from utility posterior

$\hat{\boldsymbol{\mu}} \longleftarrow \frac{1}{M} \sum_m \boldsymbol{r}_m$ ▷ Estimate $\mathbb{E}_i[\overline{\boldsymbol{r}}]$

$\hat{\Theta}_\pi \longleftarrow \{m : \boldsymbol{x}_\pi^T \boldsymbol{r}_m = \max_{\pi' \in \Pi}(\boldsymbol{x}_{\pi'}^T \boldsymbol{r}_m)\} \ \forall \pi \in \Pi$ ▷ Samples $\boldsymbol{r}_i$ for which π is optimal

$p^*(\pi) \longleftarrow \frac{|\hat{\Theta}_\pi|}{M} \ \forall \pi \in \Pi$ ▷ Empirical posterior probability of policies being optimal

$\hat{\boldsymbol{\mu}}^{(\pi)} \longleftarrow \frac{1}{|\hat{\Theta}_\pi|} \sum_{\boldsymbol{r} \in \hat{\Theta}_\pi} \boldsymbol{r} \ \forall \pi \in \Pi$ ▷ Estimate $\mathbb{E}_i[\overline{\boldsymbol{r}} \mid \pi^* = \pi]$

$\hat{L} \longleftarrow \sum_{\pi \in \Pi} p^*(\pi)(\hat{\boldsymbol{\mu}}^{(\pi)} - \hat{\boldsymbol{\mu}})(\hat{\boldsymbol{\mu}}^{(\pi)} - \hat{\boldsymbol{\mu}})^T$

$r^* \longleftarrow \sum_{\pi \in \Pi} p^*(\pi) \boldsymbol{x}_\pi^T \hat{\boldsymbol{\mu}}^{(\pi)}$ ▷ Estimate posterior reward of optimal policy

$\Delta_{\pi_1, \pi_2} \longleftarrow 2r^* - (\boldsymbol{x}_{\pi_1} + \boldsymbol{x}_{\pi_2})^T \hat{\boldsymbol{\mu}} \ \forall \pi_1, \pi_2 \in \Pi$

$\Delta \longleftarrow \sum_{\pi_1, \pi_2 \in \Pi} p^*(\pi_1) p^*(\pi_2) \Delta_{\pi_1, \pi_2}$ ▷ Expected instantaneous regret

$v_{\pi_1, \pi_2} \longleftarrow (\boldsymbol{x}_{\pi_2} - \boldsymbol{x}_{\pi_1})^T \hat{L} (\boldsymbol{x}_{\pi_2} - \boldsymbol{x}_{\pi_1}) \ \forall \pi_1, \pi_2 \in \Pi$

$v \longleftarrow \sum_{\pi_1, \pi_2 \in \Pi} p^*(\pi_1) p^*(\pi_2) v_{\pi_1, \pi_2}$ ▷ Information gain lower bound

$\hat{\Gamma} \longleftarrow \frac{\Delta^2}{2v}$ ▷ Estimated information ratio

Empirically Upper-Bounding the Information Ratio

This subsection presents a set of simulations which empirically estimate the information ratio for DPS with a linear link function. Recall that under the linear link function, the utility of a bandit action or RL trajectory $\boldsymbol{x}$ is given by $u(\boldsymbol{x}) = \frac{\overline{\boldsymbol{r}}^T \boldsymbol{x}}{c}$. All of the simulation results support the following conjecture:

Conjecture 1. *Consider running DPS in the linear dueling bandit setting and in preference-based RL with known dynamics. With the linear link function, the information ratio in Eq.* (4.16) *is upper-bounded as follows during all iterations* $i \in \{1, \dots, N\}$*:*

$$\Gamma_i \leq d,$$

where in the bandit setting, d *is the dimensionality of the action space, whereas* $d = SA$ *in the preference-based RL setting.*

This conjecture leads to the following bounds on the Bayesian regret of DPS:

Theorem 3. *Combining Conjecture 1 with Theorem 2 yields the following upper bound on the Bayesian regret of DPS in the linear dueling bandit setting as a function*

of the number of learning iterations, N:

$$\text{BayesReg}(N) \leq \sqrt{dNH(\boldsymbol{a}^*)} \leq \sqrt{dN \log A}.$$

Furthermore, combining Conjecture 1 with Corollary 1 results in the following Bayesian regret bound for DPS in the preference-based RL setting with known transition dynamics, as a function of the number of learning iterations N:

$$\text{BayesReg}(N) \leq S\sqrt{ANh \log A}.$$

Using that $T = 2Nh$, the Bayesian regret in terms of the total number of actions (or time-steps) T is:

$$\text{BayesReg}_{RL}(T) \leq S\sqrt{\frac{AT \log A}{2}},$$

where $\text{BayesReg}_{RL}(T)$ *is expressed in terms of the total number of time-steps, while* $\text{BayesReg}(N)$ *is in terms of the number of learning iterations.*

The remainder of this subsection presents empirical evidence for Conjecture 1. The simulations span four cases, each of which will be further detailed below:

A) Linear bandit setting with relative feedback and a Gaussian prior, likelihood, and posterior;

B) Linear preference-based bandit setting with posterior approximation via MCMC;

C) RL with known transition dynamics and relative feedback, with a Gaussian prior, likelihood, and posterior; and,

D) Preference-based RL with known transition dynamics and posterior approximation via MCMC.

Before presenting the results corresponding to each of these four cases, this section describes the posterior inference techniques utilized in the information ratio simulations, and details the procedure for constructing the action space $\mathcal{A}$ in the bandit simulations. Furthermore, all of the simulations use 10,000 posterior samples in each iteration to estimate the information ratio (i.e., $M = 10,000$ in Algorithm 11).

Among the four cases, two classes of model posterior are considered: a conjugate Gaussian posterior in A) and C), and MCMC in B) and D). As explained in Remark

6, the information ratio-based regret analysis techniques assume that the posterior is the exact product of a prior and likelihood. As discussed next, both the conjugate Gaussian and MCMC posterior inference approaches indeed satisfy this requirement.

Conjugate Gaussian posterior inference. In the Gaussian posterior inference approach (A and C), the information ratio is estimated under relative feedback when the prior, likelihood, and posterior are all Gaussian. Firstly, the utility vector is assumed to be drawn from a Gaussian prior:

$$\boldsymbol{r} \sim \mathcal{N}(0, \lambda^{-1} I), \tag{4.17}$$

for some hyperparameter $\lambda > 0$. The likelihood of the outcome y_i given $\boldsymbol{x}_{i1}$ and $\boldsymbol{x}_{i2}$ is equal to its expectation with i.i.d, additive Gaussian noise:

$$P(y_i \mid \boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}, \boldsymbol{r}) = \boldsymbol{r}^T (\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}) + \varepsilon_i, \text{ where } \varepsilon_i \sim \mathcal{N}(0, \sigma^2), \tag{4.18}$$

where σ is a second hyperparameter, and c is subsumed into $\boldsymbol{r}$ without loss of generality.

This Gaussian prior and likelihood yield a conjugate Gaussian posterior which can be computed in closed-form, and is given in Eq.s (4.1)-(4.2).

Clearly, under the likelihood in Eq. (4.18), the feedback y_i is not binary. Similarly to pairwise preferences, however, the feedback is determined with respect to the *difference* of the two actions or trajectory feature vectors under comparison. Thus, the feedback is relative to the selected pairs of actions or policies.

MCMC posterior inference. Secondly, in cases B) and D), the algorithm receives binary preference feedback. Since with binary feedback, the exact posterior is no longer Gaussian, this set of simulations performs MCMC posterior approximation to both estimate the information ratio and select actions. Under the linear link function model, the likelihood of each preference outcome y_i is given by:

$$P(y_i \mid \boldsymbol{x}_i, \boldsymbol{r}) = \left[\frac{2 y_i \boldsymbol{r}^T \boldsymbol{x}_i}{c} + \frac{1}{2} \right], \tag{4.19}$$

where the parameter $c > 0$ must be large enough to guarantee that $2 y_i \boldsymbol{r}^T \boldsymbol{x}_i \in (-0.5, 0.5)$ for any $\boldsymbol{x}_i$ and $\boldsymbol{r}$ supported by the prior $p(\boldsymbol{r})$; this ensures that the likelihood probabilities lie within $(0, 1)$. Then, in the n^{th} learning iterations, the exact posterior takes the form :

$$p(\boldsymbol{r} \mid \mathcal{D}_n) \propto p(\boldsymbol{r}) \prod_{i=1}^{n-1} P(y_i \mid \boldsymbol{x}_i, \boldsymbol{r}) = p(\boldsymbol{r}) \prod_{i=1}^{n-1} \left[\frac{2 y_i \boldsymbol{r}^T \boldsymbol{x}_i}{c} + \frac{1}{2} \right]. \tag{4.20}$$

Importantly, the utility prior $p(\boldsymbol{r})$ cannot be Gaussian as in A) and C), since the linear link function's posterior in Eq. (4.20) requires that $2y_i\boldsymbol{r}^T\boldsymbol{x}_i \in (-0.5, 0.5)$. Equivalently, we must have $|\boldsymbol{r}^T\boldsymbol{x}| < \frac{1}{2}$ for any vector $\boldsymbol{r}$ that is supported by the prior over $\overline{\boldsymbol{r}}$. Thus, the prior for $\overline{\boldsymbol{r}}$ must have bounded support. In this set of simulations, a truncated Gaussian prior is placed over the utilities to enforce $||\overline{\boldsymbol{r}}||_2 \leq 1$:

$$p(\boldsymbol{r}) \propto \mathcal{N}(\mathbf{0}, \lambda^{-1}I)\mathbb{I}_{[||\overline{\boldsymbol{r}}||_2 \leq 1]}, \tag{4.21}$$

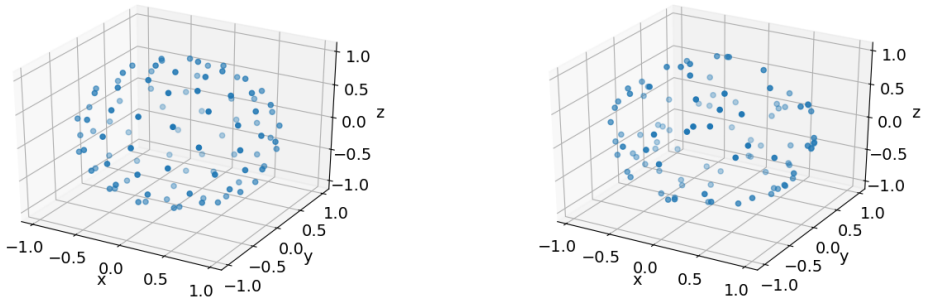
for some parameter $\lambda > 0$.

In practice, one can easily sample from the truncated Gaussian distribution by drawing samples from the Gaussian distribution $\mathcal{N}(\mathbf{0}, \lambda^{-1}I)$, and using rejection sampling to discard any sample $\boldsymbol{r}$ for which $||\boldsymbol{r}||_2 > 1$.

All MCMC simulations utilize the `emcee` Python package (Foreman-Mackey et al., 2013) to perform the MCMC sampling. Each instance of MCMC uses 32 walkers and an initial point sampled from $\mathcal{N}(\hat{\boldsymbol{r}}, 10^{-8})$ for each walker, where $\hat{\boldsymbol{r}}$ is equal to the prior mean ($\mathbf{0}$) in the first learning iteration, while in subsequent iterations, it is the mean of the previous iteration's MCMC samples.

Finally, as the simulations below will demonstrate, MCMC can estimate the model posterior with significantly-higher fidelity than the Laplace approximation.

Action space construction for information ratio simulations in the bandit setting. In the linear dueling bandit simulations, the action space consists of $A = |\mathcal{A}|$ vectors sampled from the surface of the d-dimensional unit sphere, given a specified number of actions A. Rather than sampling the actions uniformly from the sphere's surface, the actions are sampled via Poisson disk sampling (Bridson, 2007), which samples the actions sequentially and rejects candidate actions that are too close to existing samples; this ensures that the samples are spread evenly over the sphere's surface, rather than clustered together (see Figure 4.1). In particular, this work utilizes Mitchell's approximation to the Poisson disk sampling distribution (Dunbar and Humphreys, 2006), due to its simplicity of implementation. In this algorithm, points are sampled sequentially; at each step, some number of candidate points are uniformly sampled, and the one which is furthest away from any existing point is added to the set of sampled points. For large-enough numbers of candidate samples, this method is known to closely-approximate the Poisson disk sampling distribution (Dunbar and Humphreys, 2006). All of the simulations sample 1,000 candidate points at each step to generate the action space via Mitchell's approximation.



(a) Poisson disk samples

(b) Uniform samples

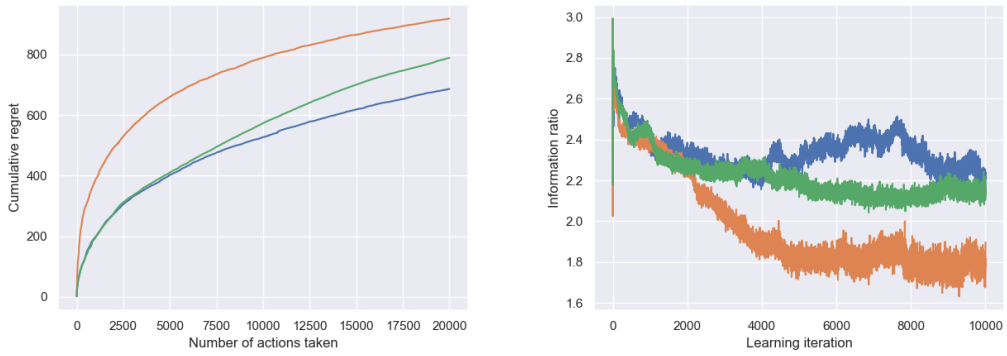
Figure 4.1: Comparison of Poisson disk sampling and uniform random sampling over the surface of the 3-dimensional unit sphere. Both plots show 100 samples. While the uniformly random samples often cluster together, the Poisson disk samples are more uniformly spaced over the sphere's surface.

A) Linear bandit setting with posterior inference via relative Gaussian feedback. In each simulation run, the action space is independently sampled using the Poisson disk method described above. Each simulation run also independently samples the utility vector $\overline{\boldsymbol{r}}$ from the prior in Eq. (4.17): $\overline{\boldsymbol{r}} \sim \mathcal{N}(0, \lambda^{-1} I)$. In each simulation, the information ratio is estimated via Algorithm 11 in every learning iteration. Samples are drawn from the Gaussian posterior (described above) to estimate Γ_i and to select actions in DPS.

This set of simulations includes 20 repetitions of each of the following parameter combinations: $\lambda \in \{0.1, 1\}$, $\sigma \in \{0.1, 1\}$, $d \in \{3, 5, 10\}$, and $A \in \{3, 10, 100, 1000\}$. These parameter combinations result in a total of 960 simulations with 10,000 learning iterations each. In each case, the same values of (λ, σ) are used to generate the feedback (i.e., the outcomes y_i) and as the hyperparameters that DPS uses to learn the utilities; thus, the algorithm knows the true environment prior and likelihood.

Figure 4.2 displays the cumulative regret and estimated information ratio values for three example simulation runs. One can see that values of Γ_i decrease as learning progresses.

Figure 4.3 plots the maximum information ratio estimate from each simulation; more specifically, for each of the 960 simulations, the information ratio value is calculated at each of the 10,000 learning iterations, and the maximum of these values is added to the plot. In all cases, the resulting information ratio estimates are strictly less than



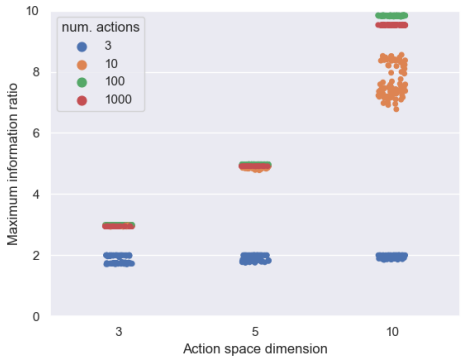
(a) Cumulative regret (b) Estimated information ratio

Figure 4.2: Cumulative regret and estimated information ratio values in the linear bandit setting with relative Gaussian feedback over pairs of actions. Values are plotted over the entire learning process for three representative experimental repetitions (colors are identical for corresponding experiment runs between the two plots). These three experiments use $d = 3$, $\lambda = 1$, $\sigma = 1$, and $A = 100$.

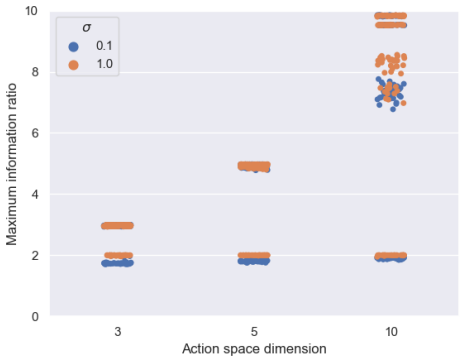
d, though they sometimes come extremely close. One can see from the figure that the Γ_i estimates tend to be larger for larger action space sizes A and noise parameters σ, but are equivalent for the two tested values of λ. Intuitively, the learning problem is harder for larger σ and A, and information ratios tend to be larger under higher outcome uncertainty, as seen in Figure 4.2.

Finally, when estimating the information ratio with a sufficiently-concentrated utility posterior $p(\boldsymbol{r} \mid \mathcal{D}_i)$, it is possible that all 10,000 samples from this posterior—which are used to estimate Γ_i—induce the same optimal action. As a result, the probabilities $p^*(\boldsymbol{x})$ (Line 5 of Algorithm 11), which empirically estimate each action's probability of being optimal, will equal one for the predicted optimal action and zero for all other actions. In this situation, Algorithm 11 is guaranteed to estimate that the information gain is zero, and therefore to return a Γ_i estimate of either ∞ or NaN ("not a number"). This phenomenon is guaranteed to occur after sufficiently many iterations, since the posterior becomes increasingly concentrated over time, but it only happens in a minority of steps for the simulations in this thesis. In these and subsequent information ratio simulations (cases A, B, C, and D), any such learning iterations are therefore excluded when identifying the maximum Γ_i estimate in the simulation run (e.g., the values shown in Figure 4.3) and when conjecturing an upper bound to Γ_i.

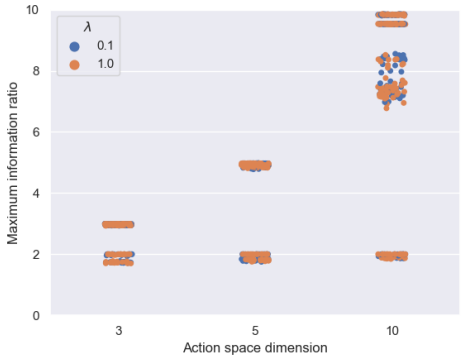
B) Linear dueling bandits with posterior inference via MCMC. Similarly to case A),



(a) Maximum information ratio values



(b) Maximum information ratio values



(c) Maximum information ratio values

Figure 4.3: Scatter plots of the maximum estimated information ratio value in each of the 960 relative Gaussian bandit simulations (each dot corresponds to one of these simulations). All 960 trials are shown in each of the three plots. Values are color-coded according to: a) the action space size A, b) σ, and c) λ.

each simulation run independently samples an action space and a utility vector $\overline{\boldsymbol{r}}$. In this case, however, $\overline{\boldsymbol{r}}$ is sampled from the truncated Gaussian prior given in Eq. (4.21). The hyperparameter c in the posterior, given by Eq. (4.20), is set to 4 to ensure that $\frac{2y_i\overline{\boldsymbol{r}}^T\boldsymbol{x}_i}{c} \in (-0.5, 0.5)$ as needed. In each case, the same values of (λ, c) are used to generate the preference feedback and as the hyperparameters that DPS uses for utility inference (i.e., the algorithm knows the true environment prior and likelihood).

MCMC sampling has a "burn-in" or "warm-up" phase, in which the sampler converges to accurately drawing samples from the model posterior. To ensure that sufficiently many warm-up steps are used to estimate Γ_i, the estimates are first compared across different numbers of warm-up steps, as shown in Figure 4.4. Each

color on the plot represents an experimental run, for five such runs. Within each of the five runs, DPS is executed with $\lambda = 1$, $d = 10$, and $A = 10$; since none of the information ratio simulations in either the bandit or RL settings exceed $d = 10$, these simulations utilize the highest dimensionality considered in this work. In each DPS learning iteration, the information ratio is estimated independently six times, each with a different number of warm-up samples: $\{0, 5, 10, 50, 100, 500\}$. As seen in the plot, the identically-colored lines from the same run mostly overlap. Thus, the number of warm-up steps does not appear to strongly influence Γ_i, perhaps because a large-enough number of posterior samples ($M = 10{,}000$) is used to estimate each information ratio.

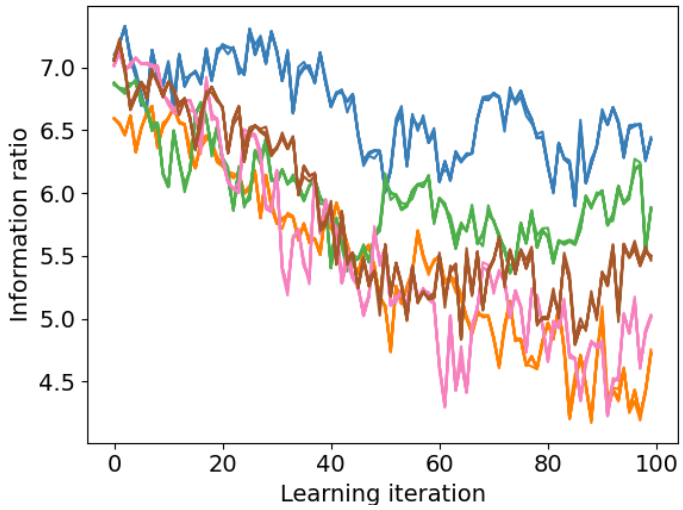


Figure 4.4: Estimating the information ratio with different numbers of MCMC warm-up steps. In each of the five DPS runs with 100 learning iterations each, the information ratio is independently estimated six times per learning iteration, once with each of the following numbers of warm-up samples: $\{0, 5, 10, 50, 100, 500\}$. On the plot, each color corresponds to one of the five simulation runs (i.e., for each run, the six different numbers of warm-up samples are plotted in the same color). Clearly, as the similarly-colored lines mostly overlap, the number of warm-up samples does not heavily impact the information ratio estimates.

In the remainder of this section (which also includes case D), all MCMC sampling uses 500 warm-up steps in the first 50 learning iterations, and 250 warm-up steps subsequently; after the first 50 steps, it is assumed that less warm-up is needed, as the MCMC sampler will by then use an improved initial point.

Notably, the Laplace approximation to the posterior is less computationally-intensive

than MCMC; however, this work uses MCMC for utility posterior approximation to estimate Γ_i values because it is more accurate than the Laplace approximation. To illustrate, Figures 4.5-4.6 depict several examples of the differences between these two posterior approximation methods.

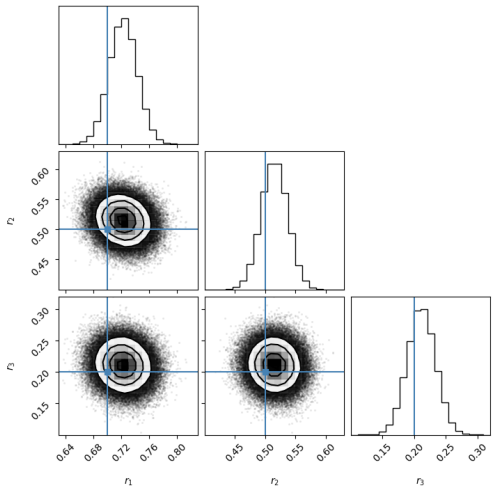
Firstly, Figure 4.5 compares posterior samples from the Laplace approximation and MCMC in two different cases. In these examples, the environment has dimensionality $d = 3$ and the learning algorithm uses $\lambda = 1$. Rather than sampling utility vectors from the prior, two different choices of $\overline{\boldsymbol{r}}$ are hard-coded: one that is close to the utility prior $p(\boldsymbol{r})$'s support boundary (recall that $p(\boldsymbol{r})$ is only supported on $||\boldsymbol{r}||_2 \leq 1$), and one that lies further from this boundary. In each of the two cases, a dataset of actions and preference outcomes is randomly-generated: actions are sampled uniformly-at-random from the surface of the unit sphere to obtain $10,000$ pairs of points, while pairwise preferences are generated via the linear link function likelihood in Eq. (4.19). Within each comparison between the Laplace approximation and MCMC, the two posterior approximations are trained over the same data. As seen in the figure, the MCMC and Laplace posteriors are similar when $\overline{\boldsymbol{r}}$ is far from the support boundary, since in this case, the true posterior is concentrated about $\overline{\boldsymbol{r}}$ in a Gaussian-like manner. When $\overline{\boldsymbol{r}}$ is closer to the support boundary, however, the MCMC posterior reflects the $||\overline{\boldsymbol{r}}||_2 \leq 1$ constraint, while the Gaussian Laplace approximation fails to capture this facet of the posterior.

Furthermore, with small numbers of preferences, the posterior is typically highly non-Gaussian. In these cases, the Laplace approximation is expected to be inaccurate. To illustrate, Figure 4.6 displays the difference between the Laplace and MCMC posteriors when running DPS after 10 and 100 preferences, respectively.

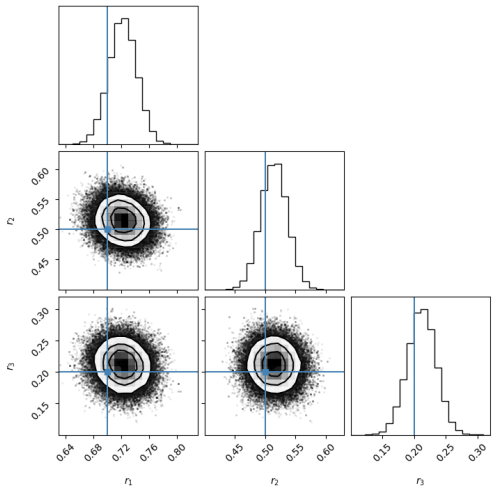
Now that the use of MCMC has been justified, a set of simulations are performed to estimate Γ_i values for DPS in the linear dueling bandit setting. For each of the following parameter combinations, 20 simulation runs of DPS are executed, each with 1,000 learning iterations: $\lambda = 1$, $d \in \{3, 5, 10\}$, and $A \in \{10, 100\}$. This results in a total of 120 simulation runs.

Figure 4.7 displays the cumulative regret and estimated information ratio values for three example simulation runs. As before, the values of Γ_i tend to be highest at the start of learning.

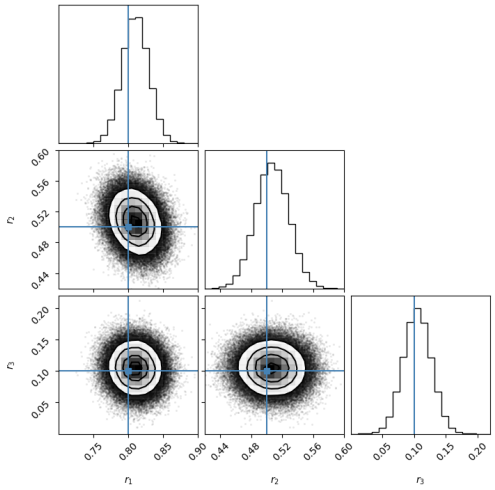
Lastly, for each of the 120 simulations, the maximum information ratio estimate is identified: in each case, this maximum is taken over the 1,000 learning iterations.



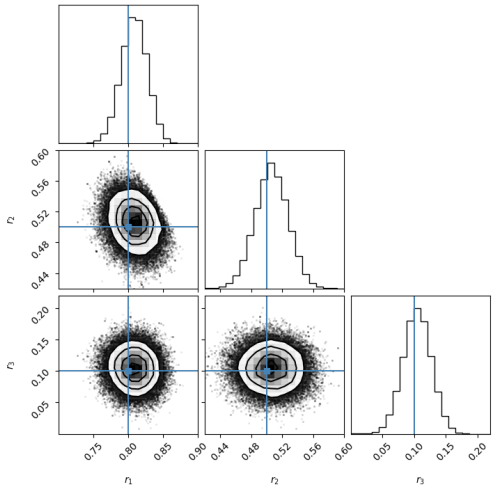
(a) Laplace approximation, case 1



(b) MCMC, case 1



(c) Laplace approximation, case 2

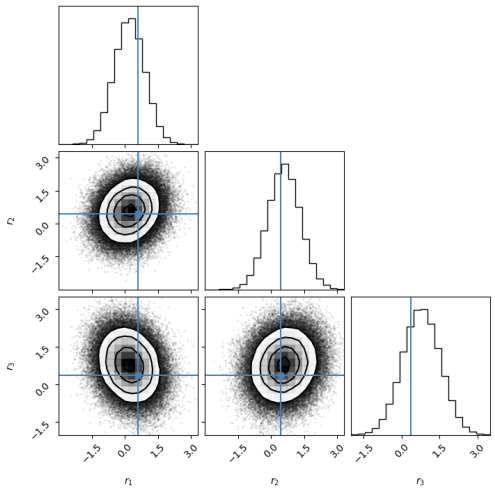


(d) MCMC, case 2

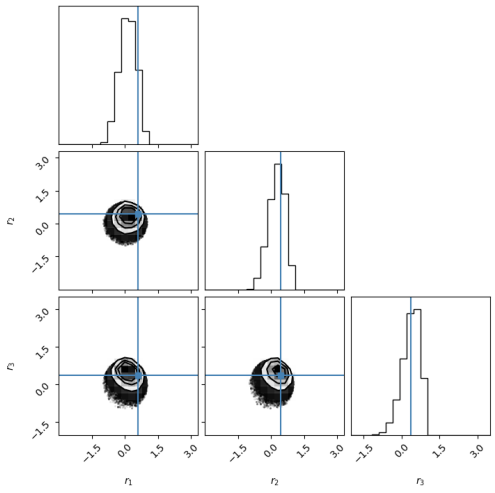
Figure 4.5: Corner plots comparing posterior samples from the Laplace approximation and MCMC. The prior over utility vectors $\overline{\boldsymbol{r}} = [r_1, r_2, r_3]^T$ is only supported on $||\overline{\boldsymbol{r}}||_2 \leq 1$. a)-b) Case 1: $\overline{\boldsymbol{r}} = [0.7, 0.5, 0.2]^T$. The true posterior is close to Gaussian, so the two approximations appear similar. c)-d) Case 2: $\overline{\boldsymbol{r}} = [0.8, 0.5, 0.1]^T$, which is closer to the boundary of allowable utility vectors, $||\overline{\boldsymbol{r}}||_2 = 1$. Because the true posterior is highly non-Gaussian, the two approximations are visibly different (compare r_1 versus r_2 in the plots). Importantly, the MCMC posterior respects the constraint $||\overline{\boldsymbol{r}}||_2 \leq 1$, while the Laplace approximation does not.

Figure 4.8 displays these numbers in a scatter plot. In every case, the maximum information ratio is strictly less than d (though in some cases, it comes close.)

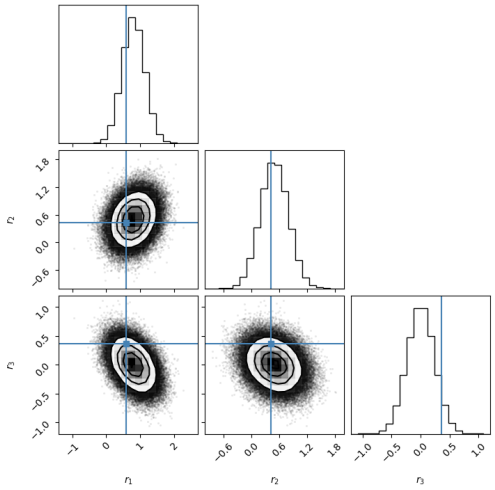
C) Preference-based RL with posterior inference via relative Gaussian feedback. In



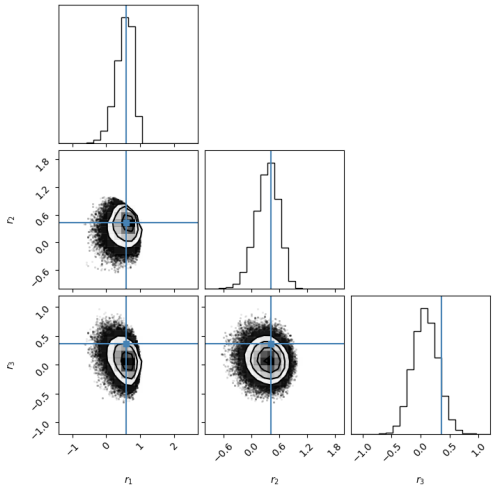
(a) Laplace approximation, 10 preferences



(b) MCMC, 10 preferences



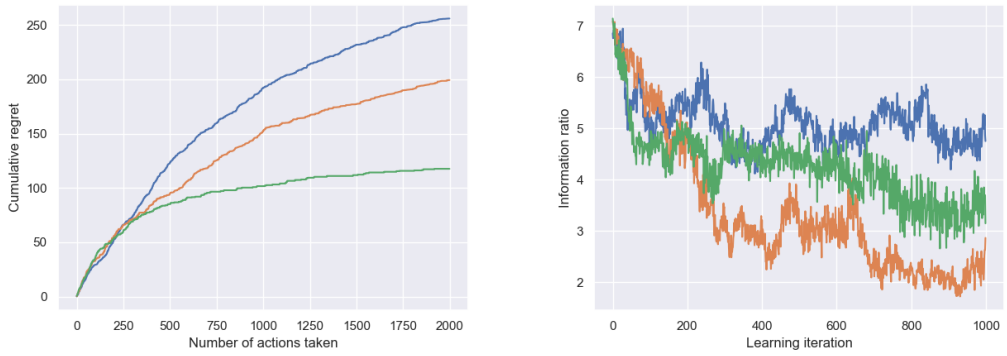
(c) Laplace approximation, 100 preferences



(d) MCMC, 100 preferences

Figure 4.6: Corner plots comparing posterior samples from the Laplace approximation and MCMC under small numbers of samples. The Laplace and MCMC posteriors are visibly different, as the utility posterior is highly non-Gaussian. The preference data was generated by running DPS with $d = 3, c = 4, \lambda = 1$, and $A = 10$.

the RL simulations, MDP parameters are generated randomly and independently for each simulation run. Firstly, the utilities $\overline{\boldsymbol{r}} \in \mathbb{R}^d$, where $d = SA$, are generated from a Gaussian distribution: $\overline{\boldsymbol{r}} \sim \mathcal{N}(\mathbf{0}, \lambda^{-1} I)$. Secondly, the MDP's state transition probabilities are sampled independently for each state-action pair from a Dirichlet distribution of order S, with all S parameters equal to 1. This Dirichlet prior is in fact a uniform distribution over all possible sets of S probabilities, $p_1, \ldots, p_S \in [0, 1]$,



(a) Cumulative regret (b) Estimated information ratio

Figure 4.7: Cumulative regret and estimated information ratio values in the preference-based linear bandit setting with posterior modeling via MCMC. Values are plotted over the learning process for three representative experimental repetitions (colors are identical for corresponding experimental runs between the two plots). The experiments in this plot use $d = 10$, $\lambda = 1$, and $A = 10$.

Figure 4.8: Scatter plot of the maximum information ratio estimates when running DPS in the linear dueling bandit setting with MCMC posterior inference. Each dot corresponds to one of 120 simulation runs, and values are color-coded according to the action space size, A.

such that $\sum_{j=1}^{S} p_S = 1$. Importantly, in this section, the MDP's transition probabilities are revealed to the algorithm, so that it must only infer the utilities. For each pair of trajectories, relative feedback is generated via the Gaussian likelihood in Eq. (4.18) just as in the bandit setting, for some parameter σ.

For each of the following parameter combinations, 20 DPS simulation runs were performed, with the information ratio estimated in each step: $\lambda \in \{0.1, 1\}, \sigma \in \{0.1, 1, 10\}$, and the five settings of the MDP parameters (S, A, h) listed in Table 4.1. This results in a total of 600 simulations with 1,000 learning iterations each. In each case, the same values of (λ, σ) are used to generate the feedback and as hyperparameters that DPS uses to learn the utilities (i.e., the algorithm knows the true environment prior and likelihood). Note that larger MDP sizes than those in Table 4.1 are not considered because calculating all the sums and differences of the policies' occupancy vectors (see Algorithm 11) requires considering $\binom{|\Pi|}{2}$ pairs of distinct policies. Storing these quantities in advance of online learning creates storage issues for larger MDPs (on a machine with 32 GB of RAM).

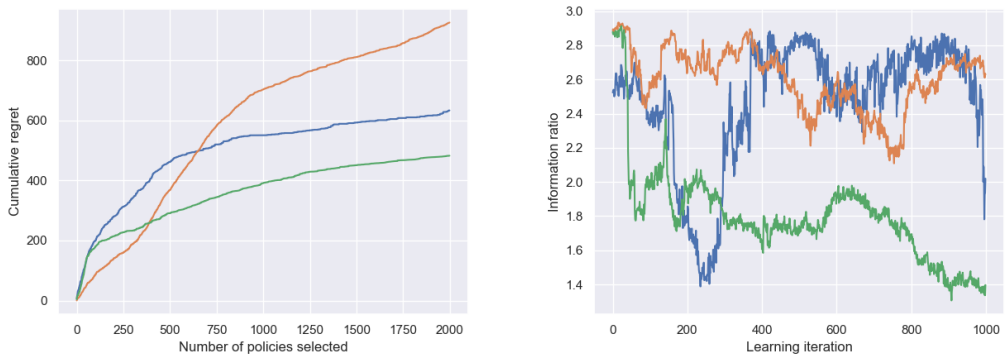
MDP label	S	A	h	$\lvert\Pi\rvert = A^{Sh}$	$\binom{\lvert\Pi\rvert}{2}$
0	2	3	4	6,561	21,520,080
1	3	2	4	4,096	8,386,560
2	4	2	3	4,096	8,386,560
3	2	4	3	4,096	8,386,560
4	3	3	2	729	265,356

Table 4.1: The labels 0-4 are assigned to the five MDP structures used in the RL information ratio simulations. For each of the five MDP structures, this table specifies the number of states S, the number of actions A, and the episode horizon h. For each (S, A, h) triple, the total numbers of deterministic policies and of distinct policy pairs (on which information ratio computations depend) are also displayed.

Figure 4.9 displays the cumulative regret and estimated information ratio values for three example simulation runs. Figure 4.10 depicts the maximum information ratio values for each simulation. One can see that higher values of σ tend to correspond to larger information ratios. Furthermore, as seen in the figure, all of the information ratio values are less than $d = SA$ by a significant margin.

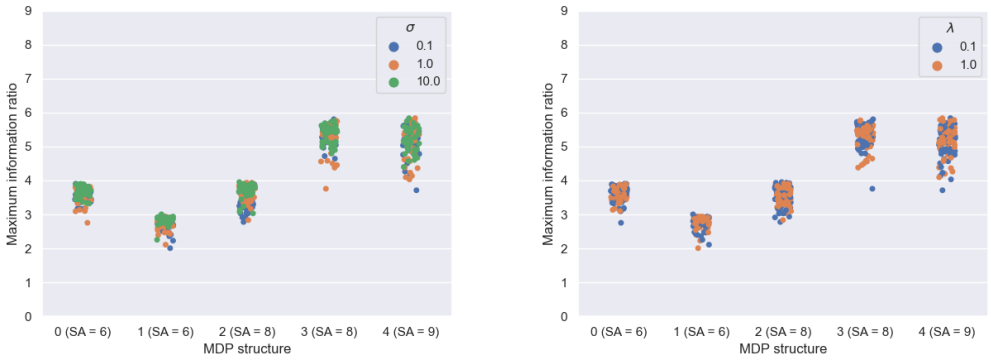
D) Preference-based RL with posterior inference via MCMC. For each simulation run, the MDP parameters are generated independently. The MDPs' transition dynamics are sampled from Dirichlet distributions as in the Gaussian RL case (C), and are similarly revealed to the algorithm. Meanwhile, as in the MCMC bandit case (B), the utilities $\overline{\boldsymbol{r}}$ are sampled from a truncated Gaussian prior to ensure that they have bounded support: $||\overline{\boldsymbol{r}}||_2 \leq 1$.

The following set of simulations are performed to estimate the information ratio, each with 1,000 learning iterations. For each of the MDP structures 1-4 in Table



(a) Cumulative regret (b) Estimated information ratio

Figure 4.9: Cumulative regret and information ratio estimates in the RL setting with relative feedback, known dynamics, and posterior inference via relative Gaussian feedback. Values are plotted over the learning process for three representative experimental repetitions (colors are identical for each experimental run between the two plots). These experiments utilize $\lambda = 1$, $\sigma = 10$, and MDP structure 1 (see Table 4.1), for which $S = 3$, $A = 2$, and $h = 4$.



(a) Maximum information ratio values (b) Maximum information ratio values

Figure 4.10: Scatter plots of the maximum information ratio estimates in each of the 600 relative Gaussian RL simulations (each dot corresponds to one of the 600 simulations). The x-axis labels indicate the MDP labels 0-4 (from Table 4.1) and the values of $d = SA$. All 600 trials are shown in both of the two plots. Values are color-coded according to: a) σ and b) λ. In all cases, the information ratios are significantly less than d.

4.1, 20 repetitions are executed with $\lambda = 1$ and $c = 2\sqrt{2}h$, to obtain 80 simulations in total. This particular setting of c is selected to ensure that $2y_i\frac{\bar{\boldsymbol{r}}^T\boldsymbol{x}_i}{c} \in (-0.5, 0.5)$,

as required by the likelihood:

$$\begin{aligned}\left|\frac{2y_i\overline{\boldsymbol{r}}^T\boldsymbol{x}_i}{c}\right| &= \left|\frac{\overline{\boldsymbol{r}}^T\boldsymbol{x}_i}{c}\right| \overset{(a)}{\leq} \frac{1}{c}||\overline{\boldsymbol{r}}||_2||\boldsymbol{x}_i||_2 \overset{(b)}{\leq} \frac{1}{c}||\boldsymbol{x}_i||_2 = \frac{1}{c}||\boldsymbol{x}_{i2}-\boldsymbol{x}_{i1}||_2 \\ &= \frac{1}{c}\sqrt{\sum_{j=1}^{d}\Big([\boldsymbol{x}_{i2}]_j - [\boldsymbol{x}_{i1}]_j\Big)^2} = \frac{1}{c}\sqrt{\sum_{j=1}^{d}\Big([\boldsymbol{x}_{i2}]_j^2 + [\boldsymbol{x}_{i1}]_j^2 - 2[\boldsymbol{x}_{i1}]_j[\boldsymbol{x}_{i2}]_j\Big)} \\ &\overset{(c)}{\leq} \frac{1}{c}\sqrt{\sum_{j=1}^{d}\Big([\boldsymbol{x}_{i2}]_j^2 + [\boldsymbol{x}_{i1}]_j^2\Big)} = \frac{1}{c}\sqrt{||\boldsymbol{x}_{i2}||_2^2 + ||\boldsymbol{x}_{i1}||_2^2} \overset{(d)}{\leq} \frac{1}{c}\sqrt{||\boldsymbol{x}_{i2}||_1^2 + ||\boldsymbol{x}_{i1}||_1^2} \\ &\overset{(e)}{=} \frac{1}{c}\sqrt{h^2+h^2} = \frac{1}{c}\sqrt{2}h,\end{aligned}$$

where (a) follows from the Cauchy-Schwarz inequality and (b) applies the prior constraint that $||\overline{\boldsymbol{r}}||_2 \leq 1$. Next, (c) holds because in the RL setting, $\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}$ contain state-action visitation counts, and so all of their elements are non-negative; thus, $[\boldsymbol{x}_{i1}]_j[\boldsymbol{x}_{i2}]_j > 0$. Fourthly, (d) follows because $||\boldsymbol{x}||_2 \leq ||\boldsymbol{x}||_1$ holds for any vector $\boldsymbol{x} \in \mathbb{R}^d$. Finally, (e) holds because the sum of a trajectory's state-action visitation counts must equal the episode horizon h. Thus, the required condition is satisfied by setting $\frac{1}{c}\sqrt{2}h = \frac{1}{2}$, resulting in $c = 2\sqrt{2}h$.

(a) Cumulative regret

(b) Estimated information ratio

Figure 4.11: Cumulative regret and estimated information ratio values in the preference-based RL setting with known dynamics and posterior inference via MCMC. Values are plotted over the learning process for three representative experimental repetitions (colors are identical for corresponding simulation runs between the two plots). These experiments use $\lambda = 1$ and MDP structure 4 (see Table 4.1), for which $S = 3, A = 3, h = 2$. The value of c is set to $2\sqrt{2}h = 4\sqrt{2}$.

In each case, the same values of (λ, c) are both used to generate the feedback and given to DPS as hyperparameters for learning the utilities; thus, the algorithm knows the true environment prior and likelihood.

Figure 4.11 displays the cumulative regret and estimated information ratio values for three example simulation runs. As before, information ratio values tend to decrease as learning progresses. Figure 4.12 depicts the maximum information ratio value observed in each of the 80 simulations, with the maximum taken over the 1,000 learning iterations. All of the information ratio values are significantly less than the dimensionality, $d = SA$.

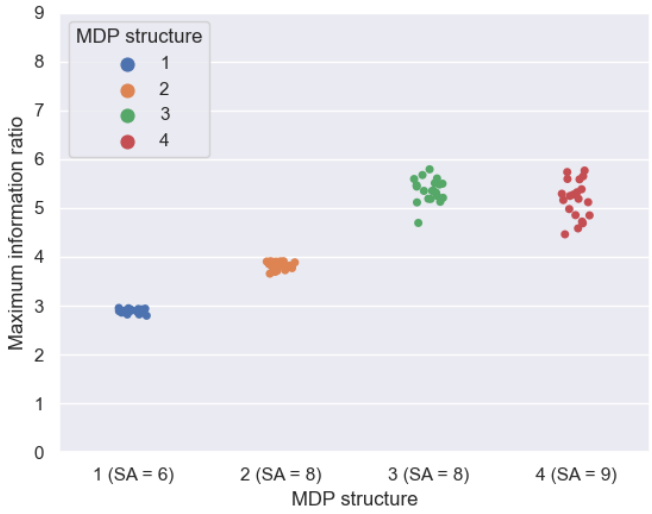


Figure 4.12: Scatter plot of the maximum information ratio estimates for the preference-based RL setting with known dynamics and MCMC posterior inference. Each dot corresponds to one of the 80 simulation runs, with the maximum taken over the 1,000 learning iterations. The x-axis labels indicate the MDP structure labels (from Table 4.1) and the values of $d = SA$. In all cases, the information ratios are significantly less than d.

4.5 Empirical Performance of DPS

This section presents empirical results of executing DPS in both the preference-based generalized linear bandit and RL settings.

Generalized Linear Dueling Bandit Simulations

The empirical performance of DPS is first evaluated in the linear and logistic dueling bandit settings, in which utility inference is respectively performed via Bayesian linear regression and Bayesian logistic regression. The results demonstrate that DPS generally performs well, and is competitive against baseline algorithms.

Experimental setup. Recall that in the linear and logistic dueling bandit settings,

DPS learns over an action space $\mathcal{A} \subset \mathbb{R}^d$ of cardinality $A = |\mathcal{A}|$. These experiments evaluate DPS with action spaces of size $A = 1,000$ and dimensionalities $d \in \{3, 5, 10\}$. For each dimension d considered, a set of 100 environments was randomly and independently sampled. Each sampled environment consists of an action space $\mathcal{A} \subset \mathbb{R}^d$ and a utility vector $\overline{\boldsymbol{r}} \in \mathbb{R}^d$. The action spaces were sampled from the surface of the d-dimensional unit sphere using the Poisson disk sampling method described in the previous section and illustrated in Figure 4.1. The utility vectors were sampled uniformly from the surface of the unit sphere.

Each simulation run consisted of 500 learning iterations, such that in each iteration, the algorithm selected two actions from $\mathcal{A}$ and received a pairwise preference between them. Preferences were generated in simulation by comparing the two actions' utilities, where the utility of an action $\boldsymbol{x} \in \mathcal{A}$ is given by $u(\boldsymbol{x}) = \overline{\boldsymbol{r}}^T \boldsymbol{x}$; note that this utility information was hidden from the learning algorithm, which only observed the pairwise preferences. For each action space dimensionality $d \in \{3, 5, 10\}$, preference feedback was generated via two different models: a) a logistic model, in which for $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{A}$, $P(\boldsymbol{x} \succ \boldsymbol{x}') = \{1 + \exp[-(u(\boldsymbol{x}) - u(\boldsymbol{x}'))/c]\}^{-1}$, and b) a linear model, $P(\boldsymbol{x} \succ \boldsymbol{x}') = (u(\boldsymbol{x}) - u(\boldsymbol{x}'))/c$. In both cases, the temperature $c > 0$ controls the degree of noisiness, and for the linear model, c is assumed to be large enough that $P(\boldsymbol{x} \succ \boldsymbol{x}') \in [0, 1]$. Note that in ties where $u(\boldsymbol{x}) = u(\boldsymbol{x}')$, preferences are generated uniformly-at-random.

Methods compared. With logistic preference feedback, the simulations consider three values of c for each dimension d: $c \in \{1, 0.1, 0.01\}$. With linear preference feedback, meanwhile, the simulations use $c = 4$. This latter value of c ensures that $P(\boldsymbol{x} \succ \boldsymbol{x}') \in [0, 1]$ for any $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{A}$, but yields significantly-noisier preference feedback than the logistic feedback models considered.

For each combination of the three dimensionalities ($d \in \{3, 5, 10\}$) and four types of preference feedback (logistic with $c \in \{1, 0.1, 0.01\}$ and linear with $c = 4$), the following learning algorithms are considered. Firstly, DPS is tested with utility inference via both Bayesian linear regression and Bayesian logistic regression. While Bayesian linear and logistic regression are both introduced in Section 4.3, implementation details of the exact Bayesian models considered in the experiments are given in Appendices A.1 and A.3, respectively.

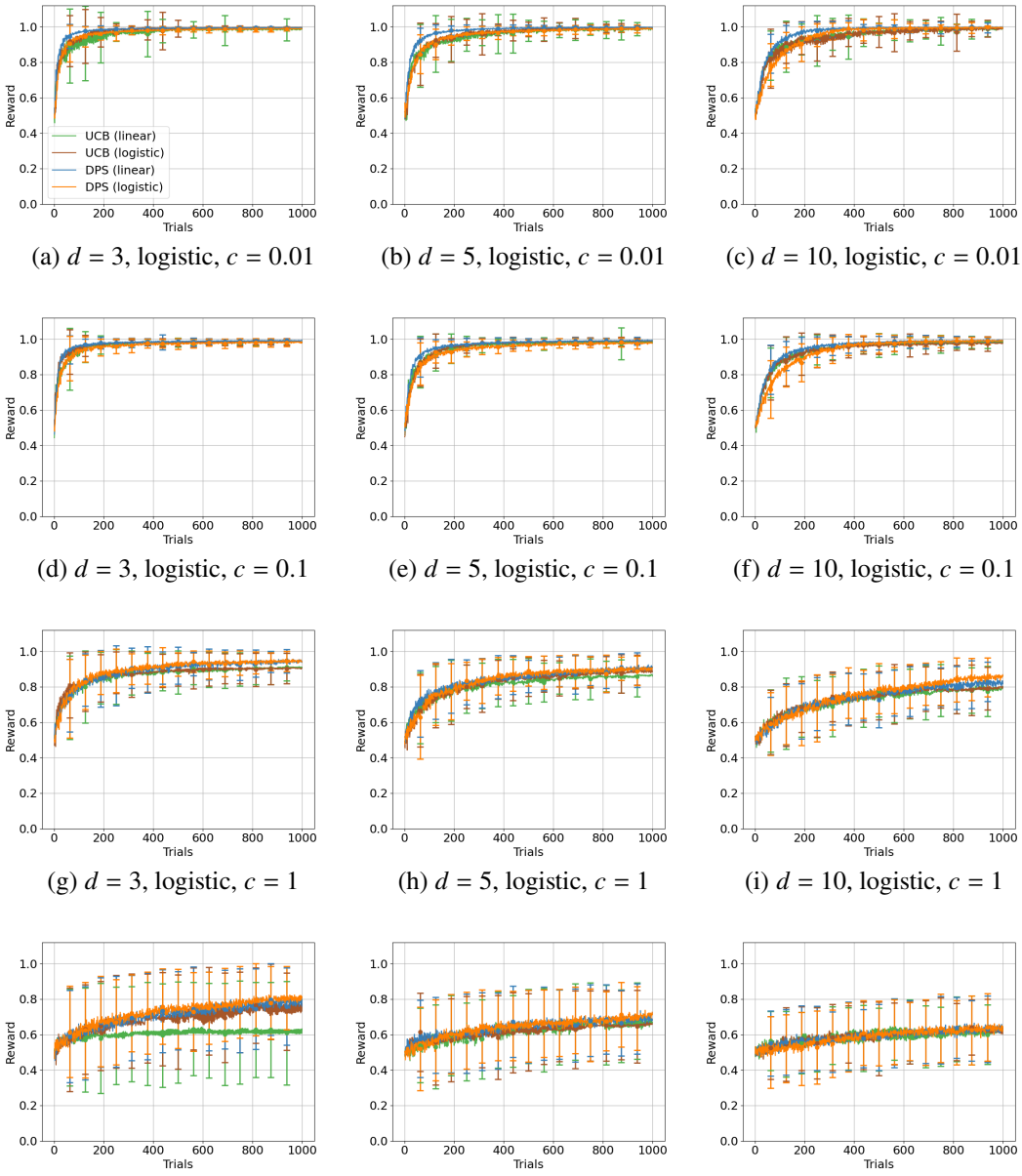
Furthermore, two baseline algorithms are considered for each combination of d and type of user feedback. Both of these baselines utilize the Sparring framework introduced in Ailon, Karnin, and Joachims (2014), in which two standard bandit

algorithms that expect absolute reward feedback compete against one another. In every learning iteration, each of the two bandit algorithms selects an action, and these two actions are dueled against one another. The algorithm whose action was preferred receives a reward of 1, while the algorithm whose action was dominated receives a reward of 0. Sparring is coupled with two upper confidence bound (UCB) algorithms: the linear UCB algorithm from Abbasi-Yadkori, Pál, and Szepesvári (2011) and the generalized linear bandits UCB algorithm from Filippi et al. (2010), specialized to the logistic link function (i.e., logistic UCB). To my knowledge, there is no prior work combining Sparring with Linear or Logistic UCB; however, this seems a natural confidence-based approach for the generalized linear dueling bandit problem, against which DPS can be compared.

The implementation details for these two baseline algorithms are given in Appendix C. Notably, for both baselines, the algorithmic hyperparameters were optimized independently for each dimension d and user feedback model. In contrast, the linear DPS and logistic DPS simulations each use a single set of hyperparameters, which was found to be well-performing across all dimensions d and user feedback models considered. In this respect, the UCB simulations were given an advantage over DPS.

Results. Figure 4.13 depicts performance curves for DPS with linear and logistic utility inference over the different action space dimensions and user feedback models considered. Appendix C also contains details about the hyperparameter ranges tested for the different methods and their performance. DPS yields robust performance under noisy preference feedback, and performs favorably compared to the two baselines considered. This may be because unlike DPS, Sparring involves two competing learning algorithms, which each only receive information about whether their actions "beat" the competing algorithms' actions, but do not account for the competing algorithm's choices of action. For all algorithms tested, performance tends to degrade as noise in the preference feedback increases.

Notably, with DPS, both the linear and logistic utility inference methods yield competitive performance when the preference feedback is generated via linear and logistic noise models. This suggests that the performance of DPS is not overly-sensitive to the model by which feedback is generated. Overall, these results imply that DPS yields promising performance in the generalized linear dueling bandit setting, and is robust under various types of preference feedback.



(a) $d = 3$, logistic, $c = 0.01$ (b) $d = 5$, logistic, $c = 0.01$ (c) $d = 10$, logistic, $c = 0.01$

(d) $d = 3$, logistic, $c = 0.1$ (e) $d = 5$, logistic, $c = 0.1$ (f) $d = 10$, logistic, $c = 0.1$

(g) $d = 3$, logistic, $c = 1$ (h) $d = 5$, logistic, $c = 1$ (i) $d = 10$, logistic, $c = 1$

(j) $d = 3$, linear, $c = 4$ (k) $d = 5$, linear, $c = 4$ (l) $d = 10$, linear, $c = 4$

Figure 4.13: Empirical performance of DPS in the generalized linear dueling bandit setting (mean ± std over 100 runs). The plots show DPS with utility inference via both Bayesian logistic and linear regression, and under both logistic (a-i) and linear (j-l) user feedback noise. The plots normalize the rewards for each simulation run such that the best and worst actions in $\mathcal{A}$ have rewards of 1 and 0, respectively. For both baselines, the hyperparameters were optimized independently in each of the 12 cases, while for linear and logistic DPS, the plots depict results for a single set of well-performing hyperparameters. DPS is competitive against both baselines and is robust to the utility inference method and to noise in the preference feedback.

Preference-Based Reinforcement Learning Simulations

The empirical performance of DPS for preference-based RL is validated in three simulated domains with varying degrees of preference noise and using three alternative credit assignment models. The results show that generally, DPS performs well and compares favorably against standard preference-based RL baselines. Python code for reproducing the experiments in this section is located online (Novoseller et al., 2020a).

Experimental setup. DPS is evaluated on three simulated environments: RiverSwim and random MDPs (described in Osband, Russo, and Van Roy, 2013) and the Mountain Car problem as detailed in Wirth (2017). The RiverSwim environment has six states and two actions (actions 0 and 1); the optimal policy always chooses action 1, which maximizes the probability of reaching a goal state-action pair. Meanwhile, a suboptimal policy—yielding a small reward compared to the goal—is quickly and easily discovered and incentivizes the agent to always select action 0. The algorithm must demonstrate sufficient exploration to have hope of discovering the optimal policy quickly.

The second environment consists of randomly-generated MDPs with 10 states and 5 actions. The transition dynamics and rewards are, respectively, generated from Dirichlet (all parameters set to 0.1) and exponential (rate parameter = 5) distributions. These distribution parameters were chosen to generate MDPs with sparse dynamics and rewards. For each random MDP, the sampled reward values were shifted and normalized so that the minimum reward is zero and their mean is one.

Thirdly, in the Mountain Car problem, an under-powered car in a valley must reach the top of a hill by accelerating in both directions to build its momentum. The state space is two-dimensional (position and velocity), while there are three actions (left, right, and neutral). The implementation, which is as described in Wirth (2017), begins each episode in a uniformly-random state and has a maximum episode length of 500. The state space is discretized into 10 states in each dimension. Each episode terminates either when the car reaches the goal or after 500 steps, and rewards are -1 in every step.

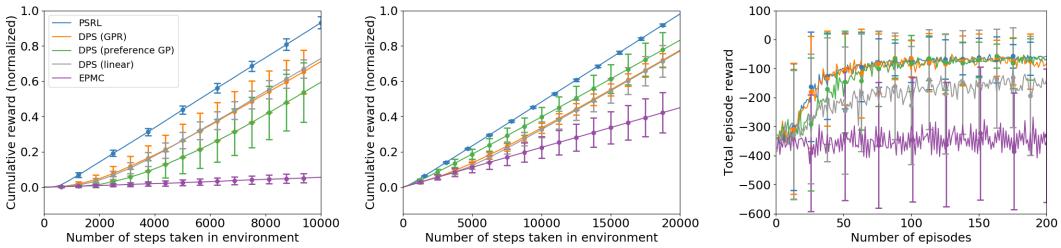
In each environment, preferences between trajectory pairs were generated by (noisily) comparing their total accrued rewards; this reward information was hidden from the learning algorithm, which observed only the trajectory preferences and state transitions. For trajectories τ_i and τ_j with total rewards $u(\tau_i)$ and $u(\tau_j)$, two models were considered for generating preferences: a) a logistic model, $P(\tau_i \succ \tau_j) = \{1 +$

$\exp[-(u(\tau_i) - u(\tau_j))/c]\}^{-1}$, and b) a linear model, $P(\tau_i \succ \tau_j) = (u(\tau_i) - u(\tau_j))/c$, where in both cases, the temperature c controls the degree of noisiness. In the linear case, c is assumed to be large enough that $P(\tau_i \succ \tau_j) \in [0, 1]$. Note that in ties where $u(\tau_i) = u(\tau_j)$, preferences are uniformly-random.

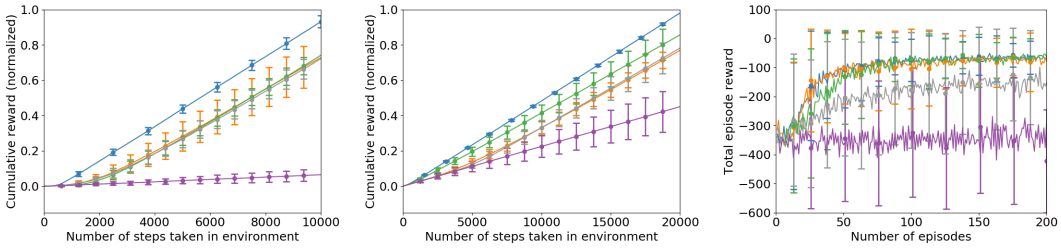
Methods compared. DPS was evaluated under three credit assignment models (described in Appendix A): 1) Bayesian linear regression, 2) Gaussian process regression, and 3) a Gaussian process preference model. User noise was generated via the logistic model, with noise levels $c \in \{10, 2, 1, 0.001\}$ for RiverSwim and random MDPs and $c \in \{100, 20, 10, 0.001\}$ for the Mountain Car. Higher values of c were selected for the Mountain Car because $|u(\tau_i) - u(\tau_j)|$ has a wider range. Additionally, the linear preference noise model was evaluated with $c = 2h\Delta\overline{\boldsymbol{r}}$, where $\Delta\overline{\boldsymbol{r}}$ is the difference between the maximum and minimum element of $\overline{\boldsymbol{r}}$ for each MDP; this choice of c guarantees that $P(\tau_i \succ \tau_j) \in [0, 1]$, but yields noisier preferences than the logistic noise models considered.

As discussed in Section 2.6, many existing preference-based RL algorithms handle a somewhat distinct setting from the one considered here, as they assume that agent-environment interactions can be simulated between preference queries and/or prioritize minimizing preference queries rather than online regret. As a baseline, this work evaluates the Every-Visit Preference Monte Carlo (EPMC) algorithm with probabilistic credit assignment (Wirth and Fürnkranz, 2013a; Wirth, 2017). While EPMC does not require simulations between preference queries, it has several limitations, including: 1) the exploration approach always takes uniformly-random actions with some probability, and thus, the authors' plots do not depict online reward accumulation, and 2) EPMC assumes that compared trajectories start in the same state. Lastly, DPS is compared against the posterior sampling RL algorithm (PSRL) from Osband, Russo, and Van Roy (2013), which observes true numerical rewards at each step, and thus upper-bounds the achievable performance of a preference-based algorithm.

Results. Figure 4.14 depicts performance curves for the three environments, each with two noise models; Appendix D contains additional experimental details, as well as results for the other noise parameters and from varying the algorithm's hyperparameters. DPS performs well in all simulations, and significantly outperforms the EPMC baseline. Typically, performance degrades as noise in the user feedback increases. In RiverSwim, however, most credit assignment models perform best when receiving the second-to-least-noisy user feedback (logistic noise, $c = 1$), since



(a) RiverSwim, $c = 0.0001$ (b) Random MDPs, $c = 0.0001$ (c) Mountain Car, $c = 0.0001$



(d) RiverSwim, $c = 1$ (e) Random MDPs, $c = 1$ (f) Mountain Car, $c = 0.1$

Figure 4.14: Empirical performance of DPS; each simulated environment is shown under the two least-noisy user preference models evaluated. The plots show DPS with three credit assignment models: Gaussian process regression (GPR), Bayesian linear regression, and a Gaussian process preference model. PSRL is an upper bound that receives numerical rewards, while EPMC is a baseline. Plots display the mean +/- one standard deviation over 100 runs of each algorithm tested. Results from the remaining user noise parameters are plotted in Appendix D. For RiverSwim and Random MDPs, normalization is with respect to the total reward achieved by the optimal policy. Overall, DPS performs well and is robust to the choice of credit assignment model.

it is harder to escape the local minimum under the least-noisy preferences. DPS is also competitive with PSRL, which has access to the full cardinal rewards at each state-action. Additionally, while the theoretical guarantees for DPS assume fixed-horizon episodes, the Mountain Car results demonstrate that it also succeeds with variable episode lengths. Finally, the performance of DPS is robust to the choice of credit assignment model, and in fact using Gaussian processes (for which we do not currently have a regret analysis) often leads to the best empirical performance. These results suggest that DPS is a practically-promising approach that can robustly incorporate many models as subroutines.

4.6 Discussion

This work investigates the preference-based reinforcement learning problem, in which an RL agent receives comparative preferences instead of absolute real-valued rewards as feedback. It develops the Dueling Posterior Sampling (DPS) algorithm, which optimizes policies in a highly efficient and flexible way. DPS integrates Bayesian credit assignment with preference-based feedback and posterior sampling, and this work proposes and evaluates several credit assignment models for DPS. The DPS framework also applies to the generalized linear dueling bandit setting, and can straightforwardly be extended to the multi-dueling bandit and RL settings, where in each iteration, the learning agent can select more than two actions or policies to elicit more than one preference. This work applies information-theoretic techniques introduced in Russo and Van Roy (2016) to analyze the regret of DPS. This involves empirically evaluating the information ratio introduced therein to conjecture that it is upper-bounded by the dimension of the observations. As a result, this approach establishes regret bounds both in the generalized linear dueling bandit setting and in preference-based RL with known transition dynamics. DPS also performs well in simulation, making it both a theoretically-justified and practically-promising algorithm.

Chapter 5

MIXED-INITIATIVE LEARNING FOR EXOSKELETON GAIT OPTIMIZATION

This chapter describes the CoSpar and LineCoSpar frameworks for learning personalized exoskeleton walking gaits to maximize user comfort. This work develops a mixed-initiative human-in-the-loop system that learns from both preference and coactive feedback to achieve efficient online optimization. The framework is deployed on the Atalante lower-body exoskeleton to learn customized, user-preferred walking trajectories in human subject experiments.

This work was done jointly with Professor Aaron Ames' research group, and in particular, in collaboration with Maegan Tucker and Myra Cheng. In addition, it is published in Tucker, Novoseller, et al. (2020b) and Tucker, Cheng, et al. (2020b).

5.1 Introduction

The field of human-robot interaction is receiving increasing attention in many application domains, from mobility assistance to autonomous driving, and from education to dialog systems. In many such domains, for a robotic system to interact optimally with a human user, it must adapt to user feedback. In particular, learning from user feedback could help to improve robotic assistive devices.

This work focuses on optimizing walking gaits for a lower-body exoskeleton, Atalante (pictured in Figure 5.1), in order to maximize user comfort. Atalante, developed by Wandercraft (Wandercraft, n.d.), uses 12 actuated joints to restore mobility to individuals with lower-limb mobility impairments. Previous work with Atalante demonstrated dynamically-stable walking using the method of partial hybrid zero dynamics (PHZD), originally designed for bipedal robots (Harib et al., 2018; Gurriet, Finet, et al., 2018; Agrawal, Harib, et al., 2017). While this method generates stable bipedal locomotion, it lacks the ability to optimize for the user's comfort; yet, user comfort should be a critical objective of gait optimization for exoskeleton walking. While existing methods (Ames, 2014) can generate human-like walking gaits for bipedal robots, it is unlikely that these methods fulfill the preferences of individuals using robotic assistance.

The exoskeleton gait optimization problem is challenging, as it involves search-

Figure 5.1: Atalante Exoskeleton with and without a user. The user is wearing a mask to measure metabolic expenditure.

ing over the vast space of all possible walking trajectories, accounting for user feedback reliability, and learning from limited data obtained from time-intensive human trials. Several existing approaches for customizing walking with various exoskeletons optimize quantitative metrics, including body parameters and targeted walking speeds (Wu, Liu, et al., 2018; Ren et al., 2019) and metabolic expenditure (Kim et al., 2017; Zhang et al., 2017). However, since the goal of this work is to optimize for user comfort, our learning approach instead queries the user for preferences between sequential gait trials. Directly incorporating personalized feedback avoids making overly-strong assumptions about gait preference, or optimizing for a numerical quantity not aligned to personalized comfort.

In many real-world settings that involve learning from human feedback, it is challenging or impossible for people to reliably specify numerical scores or to provide demonstrations (Amodei et al., 2016; Argall et al., 2009; Basu, Yang, et al., 2017; Joachims et al., 2005). In particular, this is true in the exoskeleton application, as it is difficult for users to remember many gaits at once. In contrast, the users' *relative preferences* can measure their comfort more accurately. Indeed, previous studies have found preferences to be more reliable than numerical scores in a range of domains, including information retrieval (Chapelle, Joachims, et al., 2012) and autonomous driving (Basu, Yang, et al., 2017). In the exoskeleton domain, query-

ing users for pairwise preferences only requires them to remember the current and previous gait trials. Conversely, prompting users for numerical scores requires them to remember all gait trials to ensure that the scoring is consistent over time.

While interactive preference learning has previously been used to tune parameters for an ankle exoskeleton in Thatte, Duan, and Geyer (2018), the presented approach utilizes domain knowledge to narrow the search space before performing online learning. To generate preference queries, Thatte et al. employ Double Thompson Sampling (Wu and Liu, 2016). This algorithm operates in the K-armed dueling bandit setting, in which outcomes corresponding to different actions are assumed to be independent. Yet, pairwise preferences provide a sparse feedback signal, as the algorithm only receives one bit of information per preference query. In contrast to Thatte, Duan, and Geyer (2018), we aim to optimize gaits over large gait parameter spaces without prior assumptions on the users' preferences.

Building upon techniques from dueling bandits (Sui, Zhuang, et al., 2017; Sui, Zoghi, et al., 2018; Yue, Broder, et al., 2012) and coactive learning (Shivaswamy and Joachims, 2012; Shivaswamy and Joachims, 2015), this work proposes the CoSpar algorithm to learn user-preferred exoskeleton gaits. CoSpar is a mixed-initiative approach, which both queries the user for preferences and allows the user to suggest improvements via coactive feedback. By combining multiple types of user feedback within a Gaussian process-based learning framework, CoSpar is able to identify well-performing gaits within relatively few trials. CoSpar is validated in both simulation and in human subject experiments with the Atalante exoskeleton, in which CoSpar finds user-preferred gaits within a gait library.

This work also presents the LineCoSpar algorithm, which integrates CoSpar with techniques from high-dimensional Bayesian optimization (Kirschner, Mutny, et al., 2019) to create a unified framework for performing high-dimensional preference-based learning. In simulation, LineCoSpar exhibits sample-efficient convergence to user-preferred actions in high-dimensional spaces. The algorithm is then deployed experimentally to optimize exoskeleton walking over six gait parameters (shown in Figure 5.2) for six able-bodied subjects.

In summary, the CoSpar and LineCoSpar algorithms perform sample-efficient, mixed-initiative human-in-the-loop learning to identify preferred actions in possibly high-dimensional spaces. This work not only identifies exoskeleton users' preferred walking trajectories, but can also provide insights into their preferences for certain gaits. Such knowledge could potentially help to design more comfortable

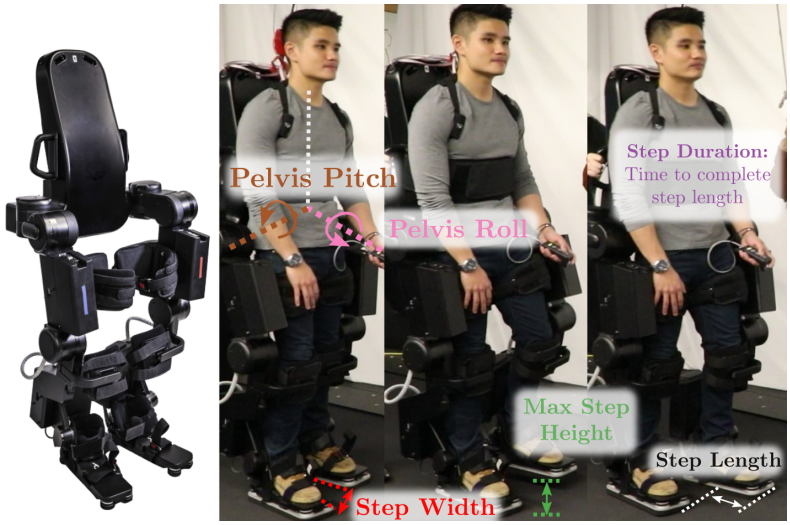


Figure 5.2: Human subject experiments with the LineCoSpar algorithm exploring six exoskeleton gait parameters: step length, step duration, step width, maximum step height, pelvis roll, and pelvis pitch.

exoskeleton gaits in the future.

5.2 Background on the Atalante Exoskeleton and Gait Generation for Bipedal Robots

Atalante (Harib et al., 2018; Duburcq et al., 2019; Gurriet, Tucker, et al., 2019), developed by Wandercraft and pictured in Figure 5.1, has 12 actuated joints: three at each hip, one at each knee, and two in each ankle. Gurriet, Finet, et al. (2018) describe the device's mechanical components and control architecture in detail. In the Atalante exoskeleton, walking is achieved using pre-computed walking gaits, generated using the partial hybrid zero dynamics framework (Ames, 2014) and a nonlinear constrained optimization process that utilizes direct collocation.

The configuration space of the human-exoskeleton system is modeled as $\boldsymbol{q} = (\boldsymbol{p}, \boldsymbol{\phi}, \boldsymbol{q}_b) \in \mathbb{R}^{18}$, where $\boldsymbol{p} \in \mathbb{R}^3$ and $\boldsymbol{\phi} \in \mathbb{SO}^3$ denote the position and orientation of the exoskeleton floating base frame with respect to the world frame, and $\boldsymbol{q}_b \in \mathbb{R}^{12}$ denotes the relative angles of the actuated joints. The generated gaits are realized on the exoskeleton using PD control at the joint level and a high-level controller that adjusts joint targets based on state feedback. The controller is executed by an embedded computer unit running a real-time operating system. Gaits are sent to the exoskeleton over a wireless connection via a custom graphical user interface.

Many existing lower-body exoskeletons either require the use of arm-crutches (Ekso

Bionics, n.d.; ReWalk, n.d.; Indego, n.d.) or use slow static gaits with speeds around 0.05 m/s (Rex Bionics, n.d.). Using the PHZD method, dynamic crutchless exoskeleton walking has been demonstrated to generate dynamically-stable gaits. This section briefly explains this method to illustrate how exoskeleton gaits can be adapted in response to user preferences; for more details on the gait generation process, refer to Harib et al. (2018), Gurriet, Finet, et al. (2018), and Agrawal, Harib, et al. (2017).

Partial Hybrid Zero Dynamics Method

Systems with impulse effects, such as ground impacts, can be represented as *hybrid control systems* (Westervelt, Grizzle, and Koditschek, 2003; Bainov and Simeonov, 1989; Ye, Michel, and Hou, 1998). Summarizing from Gurriet, Finet, et al. (2018), the natural system dynamics can then be represented on an invariant reduced-dimensional surface, termed the *zero dynamics* surface (Westervelt, Grizzle, Chevallereau, et al., 2018), by appropriately defining the *virtual constraints* and using a feedback-linearizing controller to drive them to zero. Since the exoskeleton's desired forward hip velocity is constant and its actual velocity experiences a jump at impact, the *partial zero dynamics* surface is considered. The *virtual constraints* are defined as the difference between the actual and desired outputs:

$$y_1(\boldsymbol{q}, \dot{\boldsymbol{q}}) = y_1^a(\boldsymbol{q}, \dot{\boldsymbol{q}}) - v_d$$
$$y_2(\boldsymbol{q}, \boldsymbol{\alpha}) = y_2^a(\boldsymbol{q}) - y_2^d(\tau(\boldsymbol{q}), \boldsymbol{\alpha}),$$

where the actual outputs y_1^a and y_2^a are velocity-regulating and position-modulating terms, respectively. The output y_1^a is driven to a constant desired velocity v_d, while y_2^a is driven to a vector of desired trajectories, y_2^d. The trajectories y_2^d are represented using a Bézier polynomial with coefficients $\boldsymbol{\alpha}$ and a state-based timing variable $\tau(\boldsymbol{q})$.

According to Theorem 2 in Ames (2014), if there exist virtual constraints that yield an impact-invariant periodic orbit on the *partial zero dynamics* surface, then the outputs—when properly controlled on the exoskeleton—yield stable periodic walking. The orbit is *impact-invariant* if it returns to the partial zero dynamics surface $\mathcal{PZ}_\alpha$ after an impact event. To find the polynomials $\boldsymbol{\alpha}$ that yield an impact-invariant periodic orbit on the reduced-order manifold, we formulate an optimization

problem of the form:

$$\boldsymbol{\alpha}^* = \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \quad \mathcal{J}(\boldsymbol{\alpha}), \tag{5.1}$$

$$\text{s.t.} \quad \Delta(\mathcal{S} \cap \mathcal{PZ}_{\boldsymbol{\alpha}}) \subset \mathcal{PZ}_{\alpha}, \tag{5.2}$$

$$\mathcal{W}_i \boldsymbol{x} \leq b_i, \tag{5.3}$$

where $\mathcal{J}(\boldsymbol{\alpha})$ is a user-determined cost, Eq. (5.2) is the impact invariance condition, Eq. (5.3) contains other physical constraints, $\mathcal{S}$ is the *guard* defining the conditions under which impulsive behavior occurs, and Δ is the *reset map* governing the system's dynamical response to hitting the guard.

The optimization in Eq.s (5.1)-(5.3) produces a gait that can be altered by varying the cost function $\mathcal{J}(\boldsymbol{\alpha})$ and/or adding physical constraints. In bipedal walking, this cost is frequently the mechanical cost of transport defined by Eq.s (17)-(18) in Reher et al. (2020). To create the desired motion, one must add physical constraints such as step length and foot height.

Gait Generation Applied to Lower-Body Exoskeletons

To translate the gait generation process to lower-body exoskeletons, one must choose an optimization cost function and physical constraints to obtain user-preferred gaits. While it is possible to optimize generated gaits for mechanical properties such as the cost of transport, currently, there is no well-understood relationship between user preferences and the parameters of the optimization problem. Additionally, due to the time-consuming nature of gait generation—including both the time to tune the optimization problem's constraints and the time to run the program—the problem of generating human-preferred dynamically-stable walking gaits remains largely unexplored.

Gait Library

It has become increasingly common to pre-compute a set of nominal walking gaits over a grid of various parameters (Da, Hartley, and Grizzle, 2017). These pre-computed gaits are combined to form a "gait library," from which gaits can be selected and executed immediately. For the purpose of exoskeleton walking, a gait library allows the operator to select a gait that is comfortable for the user; however, it is not yet clear how to select an appropriate walking gait to optimize user comfort and preference. Thus, we consider learning from the users' qualitative feedback, as discussed below, to select their most-preferred gaits.

This work utilizes a pre-computed gait library, in which gaits are specified by parameters that include (among others) step dimensions (step length, width, height), step duration, and pelvis roll and pitch.

5.3 The CoSpar Algorithm for Preference-Based Learning

We leverage *preference-based learning* (e.g., does the user prefer gait A over gait B?) to determine the gait parameters most preferred by the user (Sui, Zoghi, et al., 2018; Yue, Broder, et al., 2012; Shivaswamy and Joachims, 2015; Fürnkranz and Hüllermeier, 2010; Sadigh et al., 2017; Fürnkranz, Hüllermeier, et al., 2012), since preference feedback has been shown to be much more reliable than absolute feedback when learning from subjective human responses (Sui, Zoghi, et al., 2018; Joachims et al., 2005). Thus, our goal to personalize the exoskeleton's gait can be framed as a *dueling bandit* (Sui, Zoghi, et al., 2018; Yue, Broder, et al., 2012) and *coactive learning* (Shivaswamy and Joachims, 2012; Shivaswamy and Joachims, 2015) problem.

Our work builds upon the SelfSparring algorithm (Algorithm 3), a Bayesian dueling bandits approach that enjoys both competitive theoretical convergence guarantees and empirical performance (Sui, Zhuang, et al., 2017). SelfSparring learns a Bayesian posterior over each action's utility to the user and draws multiple samples from the model's posterior to "duel" or "spar" via preference elicitation. The SelfSparring algorithm iteratively: a) draws multiple samples from the posterior model of the actions' utilities; b) for each sampled model, executes the action with the highest sampled utility; c) queries for preference feedback between the executed actions; and d) updates the posterior according to the acquired preference data. Section 2.5 describes the SelfSparring algorithm in more detail.

To collect more feedback beyond just one bit per preference query, we also allow the user to suggest improvements during their trials. This approach resembles the *coactive learning* framework (Shivaswamy and Joachims, 2012; Shivaswamy and Joachims, 2015), in which the user identifies an improved action as feedback to each presented action. Coactive learning has been applied to robot trajectory planning (Jain, Sharma, et al., 2015; Somers and Hollinger, 2016), but has not, to our knowledge, yet been applied to robotic gait generation or in concert with either preference learning or Gaussian processes.

To optimize exoskeleton gaits over the gait library (Section 5.2), we propose the CoSpar algorithm, a *mixed-initiative* learning approach (Wolfman et al., 2001;

Algorithm 13 CoSpar

Input: $\mathcal{A}$ = action set, n = number of actions to select in each iteration, b = buffer size, (Σ^{pr}, c) = utility prior parameters, β = coactive feedback weight
$\mathcal{D} = \emptyset$ ▷ Initialize preference dataset
Initialize prior over $\mathcal{A}$: $(\boldsymbol{\mu}_0, \Sigma_0) = (\mathbf{0}, \Sigma^{\text{pr}})$
for $i = 1, 2, \ldots, N$ **do**
    for $j = 1, \ldots, n$ **do**
        Sample utility function f_j from $(\boldsymbol{\mu}_{i-1}, \Sigma_{i-1})$
        Select action $\boldsymbol{x}_j(i) = \text{argmax}_{\boldsymbol{x} \in \mathcal{A}} f_j(\boldsymbol{x})$
    end for
    Execute n actions $\{\boldsymbol{x}_1(i), \ldots, \boldsymbol{x}_n(i)\}$
    Observe pairwise preference feedback matrix $R \in \{0, 1, \emptyset\}\}^{n \times (n+b)}$
    for $j = 1, \ldots, n; k = 1, \ldots, n + b$ **do**
        if $R_{jk} \neq \emptyset$ **then**
            Append preference to dataset $\mathcal{D}$
        end if
    end for
    for $j = 1, \ldots, n$ **do**
        Obtain coactive feedback $\boldsymbol{x}'_j(i) \in \mathcal{A} \cup \emptyset$ ▷ $\emptyset$ = no coactive feedback given
        if $\boldsymbol{x}'_j(i) \neq \emptyset$ **then**
            Add to $\mathcal{D}$: $\boldsymbol{x}'_j(i)$ preferred to $\boldsymbol{x}_j(i)$, weight β
        end if
    end for
    Update Bayesian posterior over $\mathcal{D}$ to obtain $(\boldsymbol{\mu}_i, \Sigma_i)$
end for

Lester, Stone, and Stelling, 1999) which extends the SelfSparring algorithm to incorporate coactive feedback. Similarly to SelfSparring, CoSpar maintains a Bayesian *preference relation function* over the possible actions, which is fitted to observed preference feedback. CoSpar updates this model with user feedback and uses it to select actions for new trials and to elicit feedback. We first define the Bayesian preference model, and then detail the steps of Algorithm 13.

Bayesian Modeling of Utilities from Preference Data

We adopt the preference-based Gaussian process model of Chu and Ghahramani (2005b). Gaussian process modeling is beneficial, as it enables us to model a Bayesian posterior over a class of smooth, non-parametric functions.

Let $\mathcal{A} \subset \mathbb{R}^d$ be the finite set of available actions, with cardinality $A = |\mathcal{A}|$. At any point in time, CoSpar has collected a preference feedback dataset $\mathcal{D} = \{\boldsymbol{x}_{k1} \succ \boldsymbol{x}_{k2} \mid k = 1, \ldots, N\}$ consisting of N preferences, where $\boldsymbol{x}_{k1} \succ \boldsymbol{x}_{k2}$ indicates that the user prefers action $\boldsymbol{x}_{k1} \in \mathcal{A}$ to action $\boldsymbol{x}_{k2} \in \mathcal{A}$ in the k^{th} preference. Furthermore, we assume that each action $\boldsymbol{x} \in \mathcal{A}$ has a latent, underlying utility to

the user, $f(\boldsymbol{x})$. For finite action spaces, the utilities can be written in vector form: $\boldsymbol{f} := [f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), \ldots, f(\boldsymbol{x}_A)]^T$. Given preference data $\mathcal{D}$, we are interested in the posterior probability of $\boldsymbol{f}$:

$$P(\boldsymbol{f} \mid \mathcal{D}) \propto P(\mathcal{D} \mid \boldsymbol{f})P(\boldsymbol{f}). \tag{5.4}$$

We define a Gaussian prior over $\boldsymbol{f}$:

$$P(\boldsymbol{f}) = \frac{1}{(2\pi)^{A/2}|\Sigma^{\text{pr}}|^{1/2}} \exp\left(-\frac{1}{2}\boldsymbol{f}^T[\Sigma^{\text{pr}}]^{-1}\boldsymbol{f}\right),$$

where $\Sigma^{\text{pr}} \in \mathbb{R}^{A\times A}$ is the prior covariance matrix, such that $[\Sigma^{\text{pr}}]_{jk} = \mathcal{K}(\boldsymbol{x}_j, \boldsymbol{x}_k)$ for a kernel function $\mathcal{K}$, for instance the squared exponential kernel given in Eq. (A.4).

For computing the likelihood $P(\mathcal{D} \mid \boldsymbol{f})$, we assume that the user's preference feedback may be corrupted by noise:

$$P(\boldsymbol{x}_{k1} \succ \boldsymbol{x}_{k2} \mid \boldsymbol{f}) = g\left(\frac{f(\boldsymbol{x}_{k1}) - f(\boldsymbol{x}_{k2})}{c}\right), \tag{5.5}$$

where $g(\cdot) \in [0, 1]$ is a monotonically-increasing link function, and $c > 0$ is a hyperparameter indicating the degree of noise in the preferences. Note that the likelihood in Eq. (5.5) generalizes the one given in Chu and Ghahramani (2005b), which corresponds specifically to a Gaussian noise model as described in Section 2.2. The likelihood from Chu and Ghahramani (2005b) is obtained by setting $c = 2\sqrt{\sigma}$ and $g = \Phi$ in Eq. (5.5), where Φ is the standard Gaussian cumulative distribution function.

Thus, the full expression for the likelihood is:

$$P(\mathcal{D} \mid \boldsymbol{f}) = \prod_{k=1}^{N} g\left(\frac{f(\boldsymbol{x}_{k1}) - f(\boldsymbol{x}_{k2})}{c}\right). \tag{5.6}$$

The posterior $P(\boldsymbol{f} \mid \mathcal{D})$ can be estimated via the Laplace approximation as a multivariate Gaussian distribution; see Section 2.1 and Chu and Ghahramani (2005b) for background on the Laplace approximation. The next subsection discusses mathematical details of the Laplace approximation for the specific posterior in Eq. (5.4), and derives a condition on the link function g that is necessary and sufficient in order for the Laplace approximation to exist.

Finally, in formulating the posterior, preferences can be weighted relatively to one another if some are thought to be noisier than others. This is accomplished by

changing c to c_k in Eq. (5.6) to model differing values of the preference noise parameter among the data points, and is analogous to weighted Gaussian process regression (Hong et al., 2017).

The Laplace Approximation

The Laplace approximation yields a Gaussian distribution $\mathcal{N}(\hat{\boldsymbol{f}}, \hat{\Sigma})$ centered at the MAP estimate $\hat{\boldsymbol{f}}$:

$$\hat{\boldsymbol{f}} = \underset{\boldsymbol{f}}{\operatorname{argmin}}\big[-\log P(\boldsymbol{f} \mid \mathcal{D})\big] = \underset{\boldsymbol{f}}{\operatorname{argmin}}\, S(\boldsymbol{f}), \text{ where:}$$

$$S(\boldsymbol{f}) := \frac{1}{2}\boldsymbol{f}^T \Sigma^{\text{pr}} \boldsymbol{f} - \sum_{k=1}^{N} \log\left[g\left(\frac{f(\boldsymbol{x}_{k1}) - f(\boldsymbol{x}_{k2})}{c} \right) \right].$$

Note that $S(\boldsymbol{f})$ simply drops the constant terms from $-\log P(\boldsymbol{f} \mid \mathcal{D})$ that do not depend on $\boldsymbol{f}$. The Laplace approximation's posterior covariance $\hat{\Sigma}$ is the inverse of the Hessian matrix of $S(\boldsymbol{f})$, given by:

$$\hat{\Sigma} = \left(\nabla^2_{\boldsymbol{f}} S(\boldsymbol{f}) \right)^{-1} = (\Sigma^{\text{pr}} + \Lambda)^{-1},$$

$$\text{where } \Lambda := \nabla^2_{\boldsymbol{f}} \left\{ -\sum_{k=1}^{N} \log\left[g\left(\frac{f(\boldsymbol{x}_{k1}) - f(\boldsymbol{x}_{k2})}{c} \right) \right] \right\},$$

$$[\Lambda]_{jl} = \frac{1}{c^2} \sum_{k=1}^{N} s_k(j) s_k(l) \left[\frac{-g''\left(\frac{f(\boldsymbol{x}_{k1}) - f(\boldsymbol{x}_{k2})}{c} \right)}{g\left(\frac{f(\boldsymbol{x}_{k1}) - f(\boldsymbol{x}_{k2})}{c} \right)} + \left(\frac{g'\left(\frac{f(\boldsymbol{x}_{k1}) - f(\boldsymbol{x}_{k2})}{c} \right)}{g\left(\frac{f(\boldsymbol{x}_{k1}) - f(\boldsymbol{x}_{k2})}{c} \right)} \right)^2 \right],$$

$$\text{and } s_k(j) = \begin{cases} 1, j = k1 \\ -1, j = k2, \\ 0, \text{ otherwise} \end{cases} .$$

Thus, each term of $[\Lambda]_{jl}$ is a matrix M with only four nonzero elements, of the form:

$$\begin{cases} [M]_{k1,k1} = [M]_{k2,k2} = a \\ [M]_{k1,k2} = [M]_{k2,k1} = -a \\ [M]_{jl} = 0, \text{ otherwise,} \end{cases} , \tag{5.7}$$

$$\text{where } a = \frac{1}{c^2} \left[\frac{-g''\left(\frac{f(\boldsymbol{x}_{k1}) - f(\boldsymbol{x}_{k2})}{c} \right)}{g\left(\frac{f(\boldsymbol{x}_{k1}) - f(\boldsymbol{x}_{k2})}{c} \right)} + \left(\frac{g'\left(\frac{f(\boldsymbol{x}_{k1}) - f(\boldsymbol{x}_{k2})}{c} \right)}{g\left(\frac{f(\boldsymbol{x}_{k1}) - f(\boldsymbol{x}_{k2})}{c} \right)} \right)^2 \right].$$

It can be shown that the matrix in Eq. (5.7) is positive semidefinite if and only if $a > 0$. Therefore, it suffices to show that: $\frac{-g''\left(\frac{f(\boldsymbol{x}_{k1})-f(\boldsymbol{x}_{k2})}{c}\right)}{g\left(\frac{f(\boldsymbol{x}_{k1})-f(\boldsymbol{x}_{k2})}{c}\right)} + \left(\frac{g'\left(\frac{f(\boldsymbol{x}_{k1})-f(\boldsymbol{x}_{k2})}{c}\right)}{g\left(\frac{f(\boldsymbol{x}_{k1})-f(\boldsymbol{x}_{k2})}{c}\right)}\right)^2 \geq 0.$

Since this condition must hold for all input arguments of $g(\cdot)$, we arrive at the following final necessary and sufficient convexity condition for validity of the Laplace approximation:

$$\frac{-g''(x)}{g(x)} + \left(\frac{g'(x)}{g(x)}\right)^2 \geq 0, \ \forall \ x \in \mathbb{R}. \tag{5.8}$$

Thus, in order to show that the Laplace approximation is valid for some candidate link function g, one must simply calculate its derivatives and show that they satisfy the convexity condition in Eq. (5.8). Both the Gaussian link function $g = \Phi$ and the sigmoidal link function $g_{\log} = (1 + e^{-x})^{-1}$ satisfy Eq. (5.8).

The CoSpar Learning Algorithm

The tuple (Σ^{pr}, c) contains the prior parameters of the Bayesian preference model, as defined above. These parameters are, respectively, the covariance matrix of the Gaussian process prior and a hyperparameter quantifying the degree of noise in the user's preferences. From these parameters, one obtains the prior mean and covariance, $(\boldsymbol{\mu}_0, \Sigma_0)$ (Line 3 in Alg. 13). In each iteration i, CoSpar updates the utility model (Line 22) via the Laplace approximation to the posterior in Eq. (5.4) to obtain $\mathcal{N}(\boldsymbol{\mu}_i, \Sigma_i)$.

To select actions in the i^{th} iteration (Lines 5-8), the algorithm first draws n samples from the posterior, $\mathcal{N}(\boldsymbol{\mu}_{i-1}, \Sigma_{i-1})$. Each of these is a utility function f_j, $j \in \{1, \ldots, n\}$, which assigns a utility value to each action in $\mathcal{A}$. The corresponding selected action is simply the one maximizing f_j (Line 7): $\boldsymbol{x}_j(i) = \text{argmax}_{\boldsymbol{x} \in \mathcal{A}} f_j(\boldsymbol{x})$ for $j \in \{1, \ldots, n\}$. The n actions are executed (Line 9), and the user provides pairwise preference feedback between pairs of these actions (the user can also state "no preference").

We extend SelfSparring (Sui, Zhuang, et al., 2017) to extract more preference comparisons from the available trials by assuming that additionally, the user can remember the b actions *preceding* the current n actions:

Assumption 7 (Recall buffer)**.** *The user remembers the b trials preceding the current iteration, and can therefore give preferences (or state "no preference") between any pair of actions among the n trials in the current iteration and the b previous trials.*

The user thus provides preferences between any combination of actions within the current n trials and the previous b trials. For instance, with $n = 1$, $b > 0$, one can interpret b as a buffer of previous trials that the user remembers, such that each new sample is compared against all actions in the buffer. For $n = b = 1$, the user can report preferences between any pair of two consecutive trials, i.e., the user is asked, "Did you like this trial more or less than the previous trial?" For $n = 1$, $b = 2$, the user would additionally be asked, "Did you like this trial more or less than the second-to-last trial?" Compared to $b = 0$, a positive buffer size tends to extract more information from the available trials.

We expect that setting $n = 1$ while increasing b to as many trials as the user can accurately remember would minimize the trials required to reach a preferred gait. In Line 9, the pairwise preferences from iteration i form a matrix $R \in \{0, 1, \emptyset\}^{n \times (n+b)}$; the values 0 and 1 express preference information, while $\emptyset$ denotes the lack of a preference between the actions concerned.

Finally, the user can suggest improvements in the form of coactive feedback (Line 17). For example, the user could request a longer or shorter step length. In Line 17, $\emptyset$ indicates that no coactive feedback was provided. Otherwise, the user's suggestion is appended to the data $\mathcal{D}$ as preferred to the most recently executed action. In learning the model posterior, one can assign the coactive preferences a smaller weight relative to pairwise preferences via the input parameter $\beta > 0$.

5.4 Simulation Results for CoSpar

CoSpar's performance is evaluated in two sets of simulations, with: (1) the compass-gait biped's cost of transport,[1] and (2) a set of synthetic optimization objective functions.[2] Both cases used a Gaussian link function in their likelihood ($i.e.$, $g = \Phi$), with a preference noise parameter of $\frac{c}{\sqrt{2}} = \sigma = 0.01$ (where σ is the standard deviation of the Gaussian noise as described in Section 2.2). In both cases, CoSpar efficiently converges to the optimum.

Python code for reproducing the experiments in this section is located on Github (Tucker, Novoseller, et al., 2020a).

[1] Gaussian process kernel: squared exponential with lengthscale = 0.025, signal variance = 0.0001, noise variance = 1e-8.

[2] Gaussian process kernel: squared exponential with lengthscale = $[0.15, 0.15]$, signal variance = 0.0001, noise variance = 1e-5.

Optimizing the Compass-Gait Biped's Cost of Transport

We first evaluate our approach with a simulated compass-gait biped, optimizing its cost of transport over the step length via preference feedback (Figure 5.3). Preferences are determined by comparing cost of transport values, calculated by simulating gaits for multiple step lengths, each at a fixed forward hip velocity of 0.2 m/s. These simulated gaits were synthesized via a single-point shooting partial hybrid zero dynamics method (Westervelt, Grizzle, Chevallereau, et al., 2018).

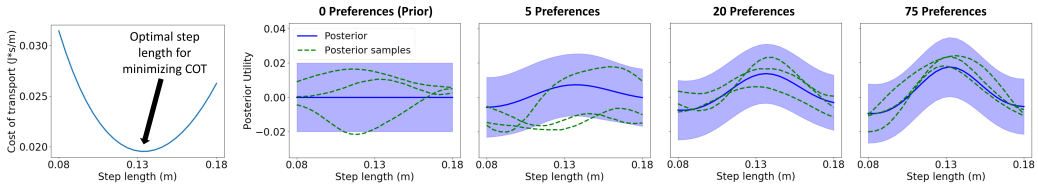


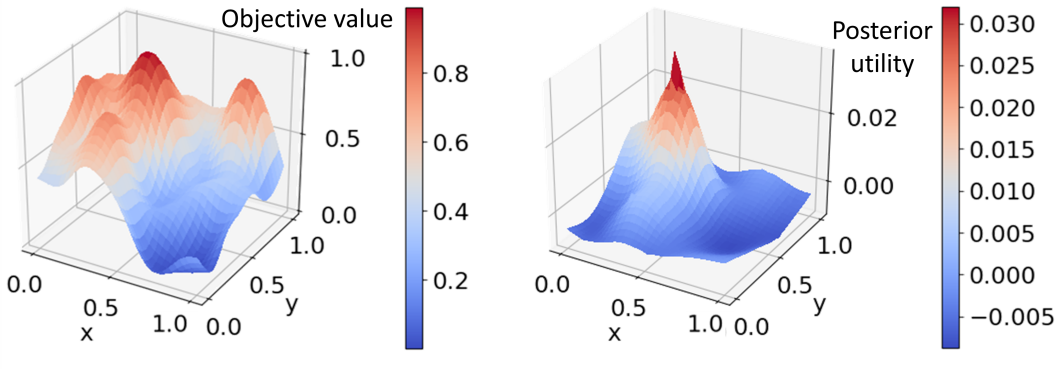
Figure 5.3: Leftmost: Cost of transport (COT) for the compass-gait biped at different step lengths and a fixed 0.2 m/s velocity. Remaining plots: posterior utility estimates of CoSpar ($n = 2$, $b = 0$; without coactive feedback) after varying iterations of learning (posterior mean +/- 2 standard deviations). The plots each show three posterior samples, which lie in the high-confidence region (mean +/- 2 standard deviations) with high probability. The posterior utility estimate quickly converges to identifying the optimal action.

We use CoSpar to minimize the one-dimensional objective function in Figure 5.3, using pairwise preferences obtained by comparing cost of transport values for the different step lengths. Here, we use CoSpar with $n = 2$, $b = 0$, and without coactive feedback. Note that without a buffer or coactive feedback, CoSpar reduces to Self-Sparring (Sui, Zhuang, et al., 2017) coupled with the Gaussian process preference model from Chu and Ghahramani (2005b). At each iteration, two new samples are drawn from the Bayesian posterior, and the resultant two step lengths are compared to elicit a preference. Using the new preferences, CoSpar updates its posterior over the utility of each step length.

Figure 5.3 depicts the evolution of the posterior preference model, where each iteration corresponds to a preference between two new trials. With more preference data, the posterior utility increasingly peaks at the point of lowest cost of transport. These results suggest that CoSpar can efficiently identify high-utility actions from preference feedback alone.

Optimizing Synthetic Two-Dimensional Functions

We next test CoSpar on a set of 100 synthetic two-dimensional utility functions, such as the one shown in Figure 5.4a. Each utility function was generated from a Gaussian process prior on a 30-by-30 grid. These experiments evaluate: 1) the potential to scale CoSpar to higher dimensions, and 2) the advantages of coactive feedback.



(a) Objective function

(b) Model posterior

Figure 5.4: CoSpar models two-dimensional utility functions using preference data. a) Example of a synthetic 2D objective function. b) Utility model posterior learned after 150 iterations of CoSpar in simulation ($n = 1$; $b = 1$; coactive feedback). CoSpar prioritizes identifying and exploring the optimal region, rather than learning a globally-accurate utility landscape.

We compare three settings for CoSpar's (n, b) parameters: $(2, 0)$, $(3, 0)$, $(1, 1)$. For each setting—as well as with and without coactive feedback—we simulate CoSpar on each of the 100 random objective functions. In each case, the number of objective function evaluations, or experimental trials, was held constant at 150.

Coactive feedback suggests improvements to actions selected by the algorithm. In simulation, such feedback is generated using a 2nd-order differencing approximation to the objective function's gradient. In each of the two dimensions, we chose the 50^{th} and 75^{th} percentiles of the gradient magnitudes as thresholds to determine when to give coactive feedback. If CoSpar selects a point at which both dimensions' gradient components have magnitudes below their respective 50^{th} percentile thresholds, then no coactive feedback is given. Otherwise, we consider the higher-magnitude gradient component, and depending on the highest threshold that it exceeds (50^{th} or 75^{th}), simulate coactive feedback as either a 5% or 10% increase in the appropriate direction and dimension.

Figure 5.5 shows the simulation results. In each case, the mixed-initiative simulations involving coactive feedback improve upon those with only preferences. Learning is slowest for $n = 2, b = 0$ (Figure 5.5), since that case elicits the fewest preferences. Figure 5.4b depicts the utility model's posterior mean for the objective function in Figure 5.4a, learned in the simulation with $n = 1$, $b = 1$, and mixed-initiative feedback. In comparing Figure 5.4b to Figure 5.4a, we see that CoSpar learns a sharp peak around the optimum, as it is designed to converge to sampling preferred regions, rather than giving the user undesirable options by exploring elsewhere.

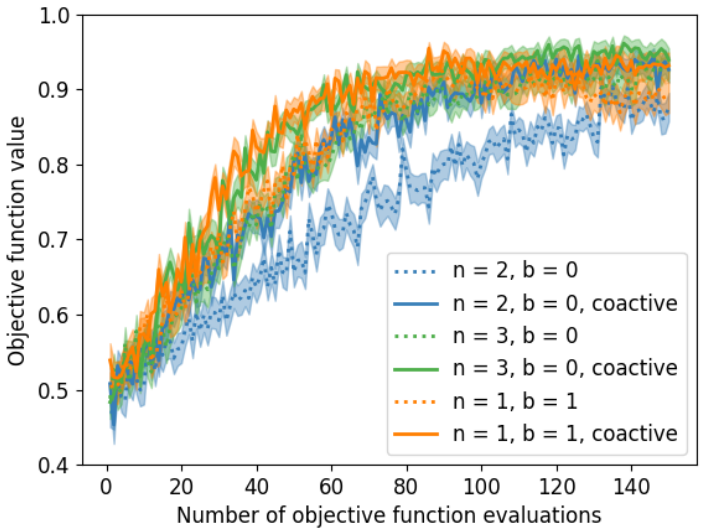


Figure 5.5: CoSpar simulation results on 100 2D synthetic objective functions, comparing CoSpar with and without coactive feedback for three settings of the parameters n and b (see Algorithm 13). Mean +/- standard error of the objective values achieved over the 100 repetitions. The maximal and minimal objective function values are normalized to 0 and 1. We see that coactive feedback always helps, and that $n = 2$, $b = 0$—which receives the fewest preferences—performs worst.

5.5 Deployment of CoSpar in Human Subject Exoskeleton Experiments

After its validation in simulation, CoSpar was deployed on a lower-body exoskeleton, Atalante, in two personalized gait optimization experiments with human subjects.[3] Both experiments aimed to determine gait parameter values that maximize user comfort, as captured by preference and coactive feedback. The first experiment,[4] repeated for three able-bodied subjects, used CoSpar to determine the users' preferred step

[3] A video of the experimental results is located at Tucker, Novoseller, et al. (2020c).

[4] Gaussian process kernel: squared exponential with lengthscale = 0.03, signal variance = 0.005, and noise variance = 1e-7. Preference noise hyperparameter: $\frac{c}{\sqrt{2}} = \sigma = 0.02$.

lengths, i.e., optimizing over a one-dimensional feature space. The second experiment[5] demonstrates CoSpar's effectiveness in two-dimensional feature spaces, and optimizes simultaneously over two different gait feature pairs. Importantly, CoSpar operates independently of the choice of gait features. The subjects' metabolic expenditure was also recorded via direct calorimetry as shown in Figure 5.1, but this data was uninformative of user preferences, as users are not required to expend effort toward walking.

Learning Preferences between Step Lengths

In the first experiment, all three subjects walked inside the Atalante exoskeleton, with CoSpar selecting the gaits. We considered 15 equally-spaced step lengths between 0.08 and 0.18 meters, each with a precomputed gait from the gait library. Feature discretization was based on users' ability to distinguish nearby values. The users decided when to end each trial, so as to be comfortable providing feedback. Since users have difficulty remembering more than two trials at once, we used CoSpar with $n = 1$ and $b = 1$, which corresponds to asking the user to compare each current trial with the preceding one. Additionally, we queried the user for coactive feedback: after each trial, the user could suggest a longer or shorter step length (±20% of the range), a *slightly* longer or shorter step length (±10%), or no feedback. Coactive feedback was added to the dataset and treated as additional preference feedback.

Each participant completed 20 gait trials, providing preference and coactive feedback after each trial. Figure 5.6 illustrates the posterior's evolution over the experiment. After only five exoskeleton trials, CoSpar was already able to identify a relatively-compact preferred step length subregion. After the 20 trials, three points along the utility model's posterior mean were selected: the maximum, mean, and minimum. The user walked in the exoskeleton with each of these step lengths in a randomized ordering, and gave a blind ranking of the three, as shown in Figure 5.6. For each subject, the blind ranking matches the preference posterior obtained by CoSpar, indicating effective learning of individual user preferences.

Learning Preferences over Multiple Features

We further demonstrate CoSpar's practicality to personalize over multiple features, by optimizing over two different feature pairs: 1) step length and step duration and 2) step length and step width. The protocol of the one-dimensional experiment

[5]Gaussian process kernel: same parameters as in [4] except for step duration lengthscale = 0.08 and step width lengthscale = 0.03.

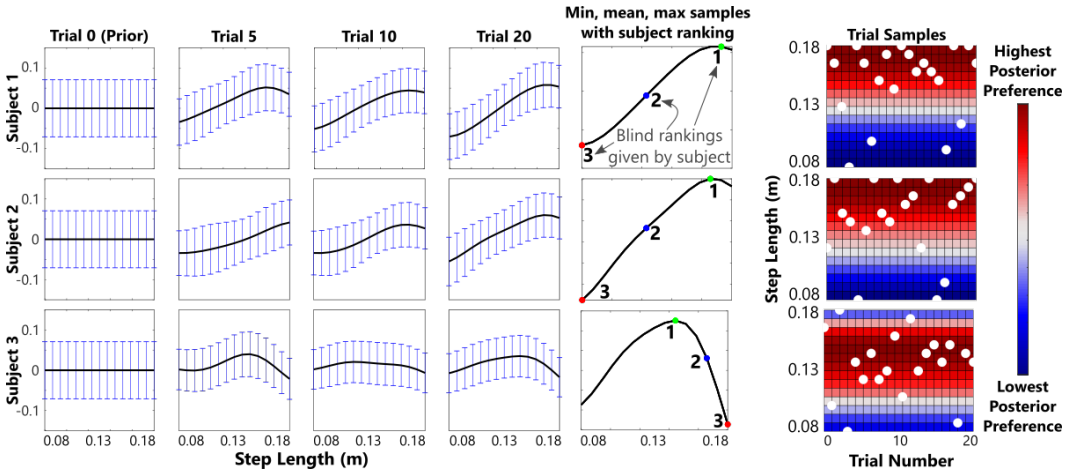


Figure 5.6: Experimental results for optimizing step length with three subjects (one row per subject). Columns 1-4 illustrate the evolution of the preference model posterior (mean +/- standard deviation), shown at various trials. CoSpar converges to similar but distinct optimal gaits for different subjects. Column 5 depicts the subjects' blind ranking of the three gaits executed after 20 trials. The rightmost column displays the experimental trials in chronological order, with the background depicting the posterior preference mean at each step length. CoSpar draws more samples in the region of higher posterior preference.

was repeated for Subject 1, with step lengths discretized as before, step duration discretized into 10 equally-spaced values between 0.85 and 1.15 seconds (with 10% and 20% modifications under coactive feedback), and step width into 6 values between 0.25 and 0.30 meters (20% and 40% modifications). After each trial, the user was queried for both a pairwise preference and coactive feedback. Figure 5.7 shows the results for both feature spaces. The estimated preference values were consistent with a three-sample blind ranking evaluation, suggesting that CoSpar successfully identified user-preferred parameters. Figure 5.8 displays phase diagrams of the gaits with minimum, mean, and maximum posterior utility values to illustrate the difference between preferred and non-preferred gaits.

5.6 The LineCoSpar Algorithm for High-Dimensional Preference-Based Learning

While the CoSpar algorithm reliably identifies user-preferred gaits in one and two-dimensional action spaces, the preference-based gait optimization problem can become intractable in larger action spaces. CoSpar must jointly maintain and sample from a posterior over every action, resulting in a computational complexity that increases exponentially in the action space dimension d. Specifically, CoSpar opti-

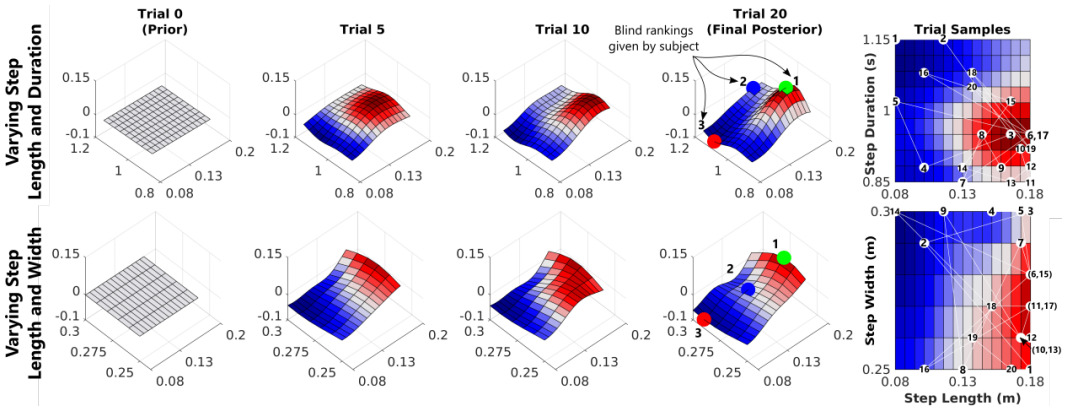


Figure 5.7: Experimental results from two-dimensional feature spaces (top row: step length and duration; bottom row: step length and width). Columns 1-4 illustrate the evolution of the preference model's posterior mean. Column 4 also shows the subject's blind rankings of the three gaits executed after 20 trials. Column 5 depicts the experimental trials in chronological order, with the background as in Figure 5.6. CoSpar draws more samples in the region of higher posterior preference.

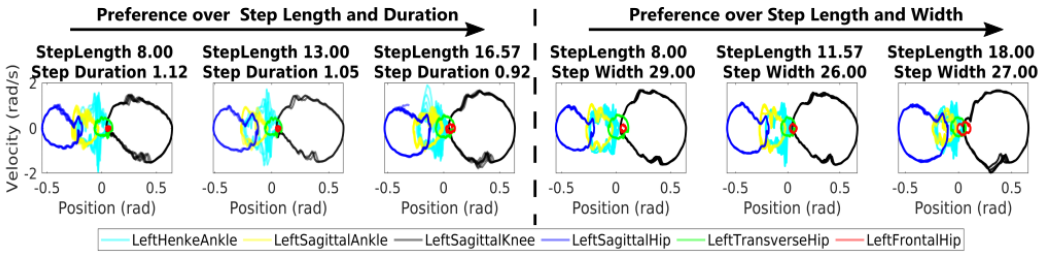


Figure 5.8: Experimental phase diagrams of the left leg joints over 10 seconds of walking. The gaits shown correspond to the maximum, mean, and minimum preference posterior values for both of subject 1's 2D experiments. For instance, Subject 1 preferred gaits with longer step lengths, as shown by the larger range in sagittal hip angles in the phase diagram.

mizes over the d-dimensional action space $\mathcal{A}$ by discretizing the entire space before beginning the learning process. With m uniformly-spaced points in each dimension of $\mathcal{A}$, this discretization results in an action space of cardinality $A = |\mathcal{A}| = m^d$, where larger m enables finer-grained search at a higher computational cost. The Bayesian preference model is updated over all A points during each iteration. This update is intractable for higher values of d, since computing the posterior over all A points involves expensive matrix operations, such as inverting $\Sigma^{\text{pr}}, \Sigma_i \in \mathbb{R}^{A \times A}$.

The LineCoSpar algorithm (Alg. 14) integrates the CoSpar framework with techniques from high-dimensional Gaussian process learning to model users' preferences

in high-dimensional action spaces. Drawing inspiration from the LineBO algorithm in Kirschner, Mutny, et al. (2019), LineCoSpar exploits low-dimensional structure in the search space by sequentially considering one-dimensional subspaces from which to sample actions. This allows the algorithm to maintain its Bayesian preference relation function over a subset of the action space in each iteration. Compared to CoSpar, LineCoSpar learns the model posterior much more efficiently and can be scaled to higher dimensions. Figure 5.9 compares computation times for CoSpar and LineCoSpar.

Algorithm 14 LineCoSpar

Input: $\mathcal{A}$ = action set; utility prior parameters (c and kernel hyperparameters); m = granularity of discretization
$\mathcal{D} = \emptyset, \mathcal{W} = \emptyset$ ▷ $\mathcal{D}$: preference data, $\mathcal{W}$: actions in $\mathcal{D}$
Set $\boldsymbol{p}_1$ to a uniformly-random action in $\mathcal{A}$
for $i = 1, 2, \ldots, N$ **do**
    $\mathcal{L}_i$ = random line through $\boldsymbol{p}_i$, discretized via m
    $\mathcal{V}_i = \mathcal{L}_i \cup \mathcal{W}$ ▷ Points over which to update posterior
    $(\boldsymbol{\mu}_i, \Sigma_i)$ = posterior over points in $\mathcal{V}_i$, given $\mathcal{D}$
    Sample utility function $f_i \sim \mathcal{N}(\boldsymbol{\mu}_i, \Sigma_i)$
    Execute action $\boldsymbol{x}_i = \text{argmax}_{\boldsymbol{x} \in \mathcal{V}_i} f_i(\boldsymbol{x})$
    Add pairwise preference between $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i-1}$ to $\mathcal{D}$
    Add coactive feedback $\boldsymbol{x}_i' \succ \boldsymbol{x}_i$ to $\mathcal{D}$
    Set $\mathcal{W} = \mathcal{W} \cup \{\boldsymbol{x}_i\} \cup \{\boldsymbol{x}_i'\}$ ▷ Update set of actions in $\mathcal{D}$
    Set $\boldsymbol{p}_{i+1} = \text{argmax}_{\boldsymbol{x} \in \mathcal{V}_i} \mu_i(\boldsymbol{x})$
end for

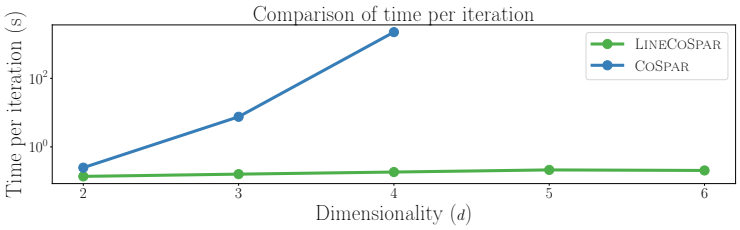


Figure 5.9: Curse of dimensionality for CoSpar. Average time per iteration of CoSpar versus LineCoSpar. The y-axis is on a logarithmic scale. For LineCoSpar, the time is roughly constant in the number of dimensions d, while the runtime of CoSpar increases exponentially. For $d = 4$, the duration of a CoSpar iteration is inconvenient in the human-in-the-loop learning setting, and for $d \geq 5$, it is intractable.

This section provides background on existing approaches for high-dimensional Gaussian process learning, and then describes the LineCoSpar algorithm, includ-

ing 1) defining the posterior updating procedure, 2) achieving high-dimensional learning, and 3) incorporating posterior sampling and coactive feedback.

High-Dimensional Bayesian Optimization

Bayesian optimization is a powerful approach for optimizing expensive-to-evaluate black-box functions. It maintains a model posterior over the unknown function, and cycles through a) using the posterior to acquire actions at which to query the function, b) querying the function, and c) updating the posterior using the obtained data. This procedure is challenging in high-dimensional search spaces due to the computational cost of the acquisition step (a), which often requires solving a non-convex optimization problem over the search space, and maintaining the posterior in the update step (c), which can require manipulating matrices that grow exponentially with the action space's dimension. Dimensionality reduction techniques are therefore an area of active interest. Solutions vary from optimizing variable subsets (DropoutBO) (Li, Gupta, et al., 2017) to projecting into lower-dimensional spaces (REMBO) (Wang et al., 2016) to sequentially optimizing over one-dimensional subspaces (LineBO) (Kirschner, Mutny, et al., 2019). We draw upon the approach of LineBO because of its state-of-the-art performance in high-dimensional spaces. Furthermore, it is especially sample-efficient in spaces with underlying low-dimensional structure. In the case of exoskeleton walking, low-dimensional structure could appear as linear relationships between two gait parameters in the user's utility function, for instance, users who prefer short step lengths also prefer short step durations.

The LineCoSpar Algorithm

Modeling Utilities Using Pairwise Preference Data. Similarly to CoSpar, LineCoSpar uses pairwise comparisons to learn a Bayesian model posterior over the relative utilities of actions (i.e., gait parameter combinations) to the user based upon the Gaussian process preference model in Chu and Ghahramani (2005b). We focus on Gaussian process methods because they model smooth, non-parametric utility functions.

As previously, $\mathcal{A} \subset \mathbb{R}^d$ represents the set of possible actions. In iteration i of the algorithm, we consider a subset of the actions $\mathcal{V}_i \subset \mathcal{A}$, with cardinality $V_i = |\mathcal{V}_i|$. Though we will define $\mathcal{V}_i$ later, we note that it includes all points in the dataset $\mathcal{D}$; the posterior is specifically modeled over points in $\mathcal{V}_i$. As in the CoSpar framework, we assume that each action $\boldsymbol{x} \in \mathcal{A}$ has a latent utility to the user, denoted as $f(\boldsymbol{x})$. Throughout the learning process, LineCoSpar stores a dataset of all user feedback, $\mathcal{D} = \{\boldsymbol{x}_{k1} \succ \boldsymbol{x}_{k2} \,|\, k = 1, \ldots, N\}$, consisting of N preferences, where

$\boldsymbol{x}_{k1} \succ \boldsymbol{x}_{k2}$ indicates that the user prefers action $\boldsymbol{x}_{k1}$ to action $\boldsymbol{x}_{k2}$. The preference data $\mathcal{D}$ is used to update the posterior utilities of the actions in $\mathcal{V}_i$. Defining $\boldsymbol{f} = [f(\boldsymbol{x}_{i_1}), f(\boldsymbol{x}_{i_2}), \ldots, f(\boldsymbol{x}_{i_{V_i}})]^T \in \mathbb{R}^{V_i}$, where $\boldsymbol{x}_{i_j}$ is the j^{th} action in $\mathcal{V}_i$, the utilities $\boldsymbol{f}$ have posterior:

$$P(\boldsymbol{f} \mid \mathcal{D}) \propto P(\mathcal{D} \mid \boldsymbol{f}) P(\boldsymbol{f}). \tag{5.9}$$

In each iteration i, we define a Gaussian process prior over the utilities $\boldsymbol{f}$ of actions in $\mathcal{V}_i$:

$$P(\boldsymbol{f}) = \frac{1}{(2\pi)^{V_i/2} |\Sigma_i^{\text{pr}}|^{1/2}} \exp\left(-\frac{1}{2} \boldsymbol{f}^T [\Sigma_i^{\text{pr}}]^{-1} \boldsymbol{f}\right), \tag{5.10}$$

where $\Sigma_i^{\text{pr}} \in \mathbb{R}^{V_i \times V_i}$ is the prior covariance matrix, which must now be recalculated in each iteration i: $[\Sigma_i^{\text{pr}}]_{jk} = \mathcal{K}(\boldsymbol{a}_{i_j}, \boldsymbol{a}_{i_k})$ for an appropriate kernel function $\mathcal{K}$. Our experiments use the squared exponential kernel.

The likelihood $P(\mathcal{D} \mid \boldsymbol{f})$ is calculated identically to the likelihood in CoSpar. Importantly, $\mathcal{V}_i$ contains all points in the dataset $\mathcal{D}$, and therefore the likelihood is well-defined:

$$P(\boldsymbol{x}_{k1} \succ \boldsymbol{x}_{k2} \mid \boldsymbol{f}) = g\left(\frac{f(\boldsymbol{x}_{k1}) - f(\boldsymbol{x}_{k2})}{c}\right),$$

where $g(\cdot) \in [0, 1]$ is a monotonically-increasing link function, and $c > 0$ is a hyperparameter indicating the magnitude of the preference noise.

While the previous CoSpar results utilize the Gaussian cumulative distribution function for g, we empirically found that using the heavier-tailed sigmoid distribution, $g_{\log}(x) := \frac{1}{1+e^{-x}}$, as the link function improves performance. The sigmoid link function $g_{\log}(x)$ satisfies the convexity conditions for the Laplace approximation described in Section 5.3 and has been used to model preferences in other contexts (Wirth, Akrour, et al., 2017). The full likelihood expression becomes:

$$P(\mathcal{D} \mid \boldsymbol{f}) = \prod_{k=1}^{N} g_{\log}\left(\frac{f(\boldsymbol{x}_{k1}) - f(\boldsymbol{x}_{k2})}{c}\right).$$

As with CoSpar, the posterior in Eq. (5.9) is estimated via the Laplace approximation to yield a multivariate Gaussian distribution, $\mathcal{N}(\boldsymbol{\mu}_i, \Sigma_i)$.

Sampling Approach for Higher Dimensions. Inspired by Kirschner, Mutny, et al. (2019), LineCoSpar overcomes CoSpar's computational intractability by iteratively modeling the posterior over one-dimensional subspaces (lines), rather than considering the full action space $\mathcal{A}$ at once. In each iteration i, LineCoSpar selects

uniformly-spaced points along a new random line $\mathcal{L}_i$ within the action space, which lies along a uniformly-random direction and intersects the action $\boldsymbol{p}_i$ that maximizes the posterior mean. Including $\boldsymbol{p}_i$ in the subspace $\mathcal{L}_i$ encourages exploration of higher-utility areas. The posterior $P(\mathcal{D} \mid \boldsymbol{f})$ is calculated over $\mathcal{V}_i := \mathcal{L}_i \cup \mathcal{W}$, where $\mathcal{W}$ is the set of actions that appear in the preference feedback dataset $\mathcal{D}$.

Critically, this approach reduces the model's covariance matrices $\Sigma_i^{\text{pr}}, \Sigma_i$ from size $A \times A$ to $V_i \times V_i$. Rather than growing exponentially in d, which is impractical for online learning, LineCoSpar's complexity is constant in the dimension d and linear in the number of iterations N. Since queries are expensive in many human-in-the-loop robotics settings, N is typically low.

Posterior Sampling Framework. Utilities are learned using the SelfSparring (Sui, Zhuang, et al., 2017) approach to posterior sampling. Specifically, in each iteration, we calculate the posterior of the utilities $\boldsymbol{f}$ over the points in $\mathcal{V}_i = \mathcal{L}_i \cup \mathcal{W}$, obtaining the posterior $\mathcal{N}(\boldsymbol{\mu}_i, \Sigma_i)$ over $\mathcal{V}_i$. The algorithm then samples a utility function f_i from the posterior, which assigns a utility to each action in $\mathcal{V}_i$. Next, LineCoSpar executes the action $\boldsymbol{x}_i$ that maximizes f_i: $\boldsymbol{x}_i = \text{argmax}_{\boldsymbol{x} \in \mathcal{V}_i} f_i(\boldsymbol{x})$. The user provides a preference (or indicates indifference, i.e. "no preference") between $\boldsymbol{x}_i$ and the preceding action $\boldsymbol{x}_{i-1}$.

In addition, for each executed action $\boldsymbol{x}_i$, the user can provide coactive feedback, specifying the dimension, direction (higher or lower), and degree in which to change $\boldsymbol{x}_i$. The user's suggested action $\boldsymbol{x}_i'$ is added to $\mathcal{W}$, and the feedback is added to $\mathcal{D}$ as $\boldsymbol{x}_i' \succ \boldsymbol{x}_i$. In each iteration, preference and coactive feedback each add at most one action to $\mathcal{W}$. Thus, in iteration i, $\mathcal{V}_i$ contains at most $m + 2(i-1)$ actions, so its size is independent of the dimensionality d. In the subsequent analysis, $\boldsymbol{x}_{\max}$ is defined as the action maximizing the final posterior mean after N iterations, i.e., $\boldsymbol{x}_{\max} := \text{argmax}_{\boldsymbol{x} \in \mathcal{V}_i} \mu_{N+1}(\boldsymbol{x})$.

Note that LineCoSpar can be generalized to include the n and b hyperparameters in CoSpar, which respectively allow the algorithm to sample multiple actions per learning iteration and to query the user for preferences between trials in non-consecutive iterations. The LineCoSpar description in Algorithm 14 sets $n = b = 1$, since it is hard for exoskeleton users to remember more than the current and previous gait trial at any given time.

5.7 Performance of LineCoSpar in Simulation

We validate the performance of LineCoSpar in simulation using both standard Bayesian optimization benchmarks and randomly-generated polynomials.[6] The simulations show that LineCoSpar is sample-efficient, converges to sampling high-valued actions, and learns a preference relation function such that actions with higher objective values have high posterior utilities.

Python code for reproducing the LineCoSpar simulation results is available on Github (Tucker, Cheng, et al., 2020c).

LineCoSpar Performance on Standard Bayesian Optimization Benchmarks

We evaluated the performance of LineCoSpar on the standard Hartmann3 (H3) and Hartmann6 (H6) benchmarks (3 and 6 dimensions, respectively). We do not compare LineCoSpar to other optimization methods because there are no other preference-based Gaussian process methods that are tractable in high dimensions. We validate LineCoSpar with noiseless preferences and then demonstrate its robustness to noisy user preferences. Preferences are generated in simulation by comparing objective function values.

Under ideal preference feedback, $\boldsymbol{x}_{k1} \succ \boldsymbol{x}_{k2}$ if $f(\boldsymbol{x}_{k1}) > f(\boldsymbol{x}_{k2})$. The true objective values f are invisible to the algorithm, which observes only the preference dataset $\mathcal{D}$. Compared to CoSpar, LineCoSpar converges to sampling actions with higher objective values at a faster rate (Figure 5.10). Thus, LineCoSpar not only enables higher-dimensional optimization, but also improves speed and accuracy of learning.

Since human preferences may be noisy, we tested the algorithm's robustness to noisy preference feedback. In simulation, this is modeled via $P(\boldsymbol{x}_{k1} \succ \boldsymbol{x}_{k2}) = \left(1 + e^{-\frac{s_k}{c_h}}\right)^{-1}$, where $s_k = f(\boldsymbol{x}_{k1}) - f(\boldsymbol{x}_{k2})$ and c_h is a hyperparameter capturing the noise level. As $c_h \to \infty$, the preferences approach uniform randomness (i.e. become noisier). Also, actions become less distinguishable when the distance between $f(\boldsymbol{x}_{k1})$ and $f(\boldsymbol{x}_{k2})$ decreases. This reflects human preference generation, since it is more difficult to give consistent preferences between actions with similar utilities. By simulating noisy preferences, we demonstrate that LineCoSpar is robust to noisy feedback. Figure 5.11 displays the results.

[6] All experiments use the squared exponential Gaussian process kernel with lengthscale 0.15 in every dimension, signal variance 1e−4, noise variance 1e−5, and preference noise $c = 0.005$ with a sigmoidal link function.

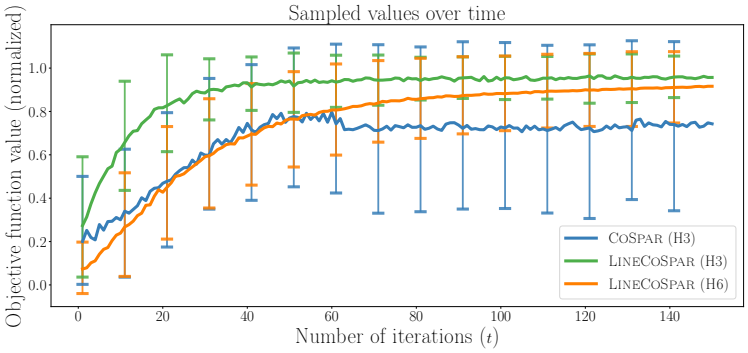


Figure 5.10: Convergence to higher objective values on standard benchmarks. Mean objective value ± standard deviation using H3 and H6, averaged over 100 runs. Compared to CoSpar, LineCoSpar converges to sampling actions with higher objective values at a faster rate, as it employs an improved sampling approach and link function. It is intractable to run CoSpar on a 6-dimensional action space.

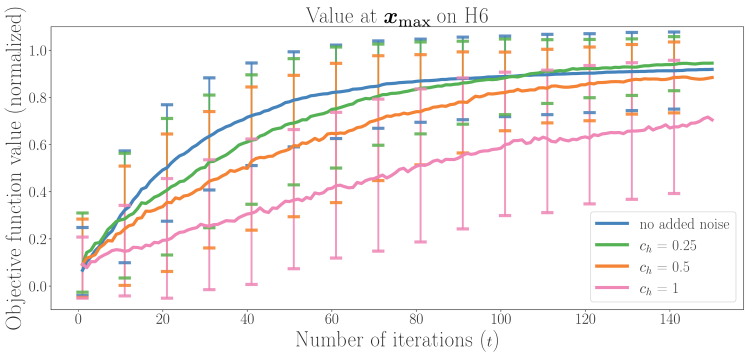


Figure 5.11: Robustness to noisy preferences. Mean objective value ± standard deviation of the action $\boldsymbol{x}_{\max}$ with the highest posterior utility. This is averaged over 100 runs of LineCoSpar on H6 with varying preference noise, as quantified by c_h. Higher performance correlates with less noise (lower c_h). The algorithm is robust to noise up to a certain degree ($c_h \leq 0.5$).

LineCoSpar Simulations with Randomly-Generated Objective Functions

We also tested LineCoSpar using randomly-generated d-dimensional polynomials (for $d = 6$) as objective functions: $p(\boldsymbol{x}) = \sum_{l=1}^{d} \alpha_l \sum_{j=1}^{d} \beta_j x_j$, where x_j denotes the j^{th} element of $\boldsymbol{x} \in \mathcal{A}$, and $\alpha_j, \beta_j, j \in \{1, \ldots, d\}$ are sampled independently from the uniform distribution $\mathcal{U}(-1, 1)$. The dimensions' ranges and discretizations match those in the exoskeleton experiments, so that LineCoSpar's performance in these simulations approximates the number of human trials needed to find optimal

gaits.

Coactive feedback was simulated for each sampled action $\boldsymbol{x}_i$ by finding an action $\boldsymbol{x}'_i$ with a higher objective value that differs from $\boldsymbol{x}_i$ along only one dimension. The action $\boldsymbol{x}'_i$ is determined by randomly choosing a dimension in $\{1, \ldots, d\}$ and direction (positive or negative), and taking a step from $\boldsymbol{x}_i$ along this vector. If the resulting action $\boldsymbol{x}'_i$ has a higher objective value, it is added to the dataset $\mathcal{D}$ as $\boldsymbol{x}'_i \succ \boldsymbol{x}_i$. This is a proxy for the human coactive feedback acquired in the exoskeleton experiments described below, in which the user can suggest a dimension and direction in which to modify an action to obtain an improved gait.

Figure 5.12 displays LineCoSpar's performance over 100 randomly-generated polynomials (10 repetitions each) with computation time shown in Figure 5.9. The results demonstrate that LineCoSpar samples high-valued actions within relatively few iterations ($\approx$ 20 trials with coactive feedback).

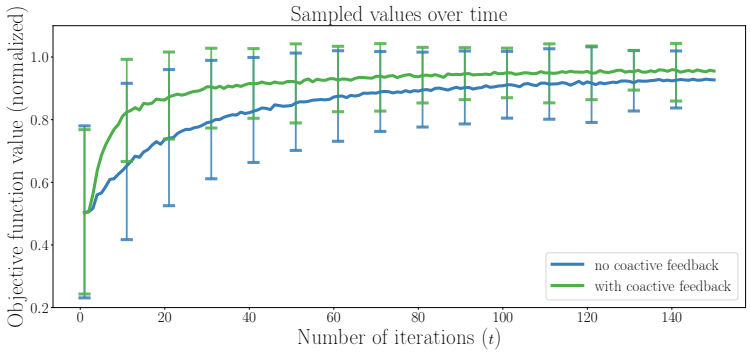


Figure 5.12: Coactive feedback improves convergence of LineCoSpar. Mean objective value ± standard deviation of the sampled actions using random objective functions. Results are averaged over 1,000 runs of LineCoSpar over 100 randomly-generated six-dimensional functions ($d = 6$; 10 runs per synthetic function). The sampled actions converge to high objective values in relatively few iterations, and coactive feedback accelerates this process.

5.8 Deployment of LineCoSpar in Human Subject Exoskeleton Experiments

After LineCoSpar's performance was demonstrated in simulation, the algorithm was experimentally deployed on the lower-body exoskeleton Atalante (Figure 5.2) to optimize six gait parameters for six able-bodied users. [7]

[7] Please see Tucker, Cheng, et al. (2020a) for a video of the experimental results.

Experimental Procedure

LineCoSpar optimized exoskeleton gaits for six self-identified able-bodied subjects over the six gait parameters shown in Figure 5.2: step length, step duration, step width, maximum step height, pelvis roll, and pelvis pitch. These parameters were chosen from the pre-computed gait library because they are relatively intuitive for users to understand when giving coactive feedback. The parameter ranges, respectively, are: 0.08-0.18 meters, 0.85-1.15 seconds, 0.25-0.3 meters, 0.065-0.075 meters, 5.5-9.5 degrees, and 10.5-14.5 degrees. Figure 5.13 illustrates the experimental procedure for testing and validating LineCoSpar.

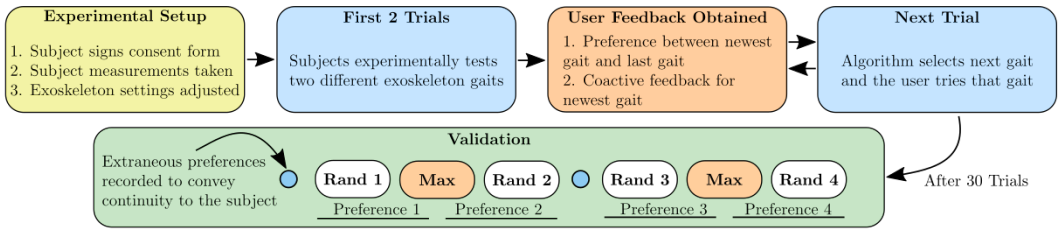


Figure 5.13: LineCoSpar experimental procedure. After setup of the subject-exoskeleton system, subjects were queried for preferences between all consecutive gait pairs, along with coactive feedback, in 30 gait trials (in total, at most 29 pairwise preferences and 30 pieces of coactive feedback). After these 30 trials, the subject unknowingly entered the validation portion of the experiment, in which he/she validated the posterior-maximizing gait, $\boldsymbol{x}_{\max}$, against four randomly-selected gaits.

All subjects were volunteers without prior exoskeleton exposure. For each subject, the testing procedure lasted approximately two hours, with one hour of setup and one hour of exoskeleton testing. The setup consisted of explaining the procedure (including how to provide preference and coactive feedback), measuring subject parameters, and adjusting the thigh and shank lengths of the exoskeleton to the subject. During testing, the subjects had control over initiating and terminating each instance of exoskeleton walking and were instructed to try each walking gait until they felt comfortable giving feedback. The subjects could choose to test each gait multiple times to confirm their preference. They could also specify "no preference" between two trials, in which case no new preference was added to the dataset $\mathcal{D}$.

After completing 30 trials (including trials with no preference, but excluding voluntary gait repetitions), the subject began a set of "validation" trials; the subject was not informed of the start of the validation phase. Validation consisted of six additional trials and yielded four pairwise preferences, each between the posterior-maximizing

action $\boldsymbol{x}_{\max}$ and a randomly-generated action. This validation step verifies that $\boldsymbol{x}_{\max}$ is preferred over other parameter combinations across the search space.

Gait Optimization Results

Figure 5.14 shows that the LineCoSpar algorithm both explores across the gait parameter space and exploits regions with higher posterior utility. Over time, LineCoSpar increasingly samples actions concentrated in regions of the search space that are preferred based on previous feedback. This results in a significant correlation between visitation frequencies and posterior utilities across these regions (Pearson's p-value = 1.22e-10).

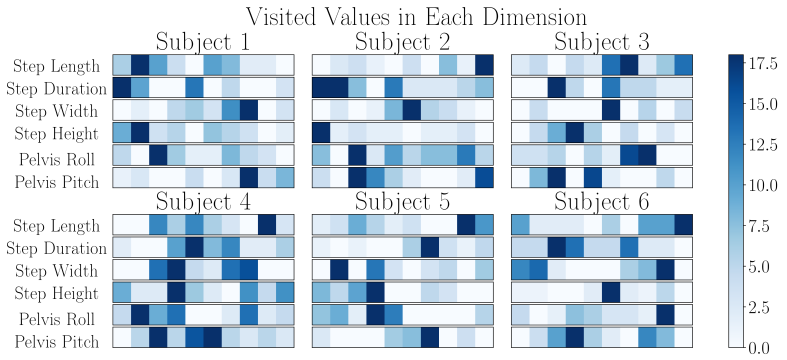


Figure 5.14: Exploration versus exploitation in the LineCoSpar human trials. Each row depicts the distribution of a particular gait parameter's values across all gaits that the subject tested. Each dimension is discretized into 10 bins. Note that the algorithm explores different parts of the action space for each subject. These visitation frequencies exhibit a statistically-significant correlation with the posterior utilities across these regions (Pearson's p-value = 1.22e-10).

For each subject, Table 5.1 lists the parameters of the predicted optimal gaits, $\boldsymbol{x}_{\max}$, identified by LineCoSpar. Table 5.1 also illustrates the results of the validation trials for each subject. These results show that $\boldsymbol{x}_{\max}$ was predominantly preferred over the randomly-selected actions during validation. For four of the six subjects, all four validation preferences matched the posterior, while the other subjects matched three and one of the four preferences, respectively. Incorrect validation preferences may be due to noise in the users' preference feedback or because 30 trials is insufficient to completely explore the entire gait parameter space.

5.9 Discussion

This work presents several contributions. Firstly, it develops the CoSpar interactive learning framework for efficient, mixed-initiative learning from human pref-

Table 5.1: Gait parameters optimizing LineCoSpar's posterior mean ($\boldsymbol{x}_{\max}$) for each able-bodied subject.

Subject	Step Length (m)	Step Duration (s)	Step Width (m)	Max Step Height (m)	Pelvis Roll (deg)	Pelvis Pitch (deg)	Validation Accuracy (%)
1	0.0835	0.943	0.278	0.0674	6.38	10.9	**75**
2	0.136	1.04	0.285	0.0679	6.41	12.4	**100**
3	0.137	0.922	0.279	0.0688	8.56	11.4	**100**
4	0.127	0.989	0.268	0.065	6.68	12.7	**25**
5	0.161	1.05	0.258	0.0689	7.32	13.2	**100**
6	0.177	1.11	0.256	0.0663	7.71	13.5	**100**

erences and coactive feedback. Secondly, building upon the CoSpar algorithm, we develop the LineCoSpar algorithm, which presents the first demonstration of high-dimensional preference-based learning to our knowledge. We evaluate both algorithms in simulation, showing that they learn to select optimal actions efficiently and robustly.

Thirdly, CoSpar and LineCoSpar are deployed in human subject experiments with the Atalante lower-body exoskeleton to learn personalized, user-preferred walking gaits.[8] In particular, LineCoSpar optimizes gaits over six gait parameters for six able-bodied subjects. This work demonstrates the first application of preference-based learning for optimizing dynamic crutchless walking. As seen in Figures 5.6 and 5.7, CoSpar successfully models the users' preferences, identifying compact subregions of preferred gaits. Furthermore, as seen in Table 5.1, different subjects have varying most-preferred gaits, underscoring the importance of individualized gait optimization and of studying the mechanisms underlying exoskeleton users' preferences. This research presents promising advancements for human-in-the-loop optimization in clinical trials and for the broader rehabilitation community.

[8] Videos of the experimental results can be seen at Tucker, Novoseller, et al. (2020c) and Tucker, Cheng, et al. (2020a) for CoSpar and LineCoSpar, respectively.

Chapter 6

CONCLUSIONS AND FUTURE DIRECTIONS

6.1 Conclusion

This dissertation addresses the problem of learning from qualitative, human-in-the-loop feedback, with the goal of developing algorithmic frameworks that robustly and efficiently learn to optimize their interactions with humans in real-world settings. Toward this goal, this thesis begins by presenting the Dueling Posterior Sampling (DPS) framework for learning from human preferences in the generalized linear dueling bandit and preference-based reinforcement learning (RL) settings. The DPS framework integrates preference-based posterior sampling with Bayesian inference of: 1) the utilities underlying the user's preferences and 2) the environment's transition dynamics in the RL setting. Several Bayesian credit assignment models are developed and evaluated together with DPS.

In addition, the DPS algorithm is developed in concert with a theoretical analysis framework, which adapts the information-theoretic posterior sampling analysis in Russo and Van Roy (2016) to the preference-based feedback setting. This type of analysis depends upon upper-bounding the information ratio, a quantity which directly balances between exploration and exploitation. Experiments suggest that DPS has a bounded information ratio when coupled with a linear link function, and therefore, that it has a competitive, finite-time Bayesian regret guarantee. Furthermore, DPS performs well empirically in a range of simulations, making it both a theoretically-grounded and practically-promising algorithm.

This thesis also presents the CoSpar and LineCoSpar algorithms for mixed-initiative learning from pairwise preferences and coactive feedback, in which the user suggests improved actions to the algorithm. These algorithms assume that the user's feedback is explained by an underlying utility function, and use Gaussian process modeling to learn a Bayesian posterior over these utilities. These algorithms demonstrate sample-efficient convergence to well-performing actions and are robust to noise in the users' preferences. Furthermore, the LineCoSpar algorithm reliably learns over high-dimensional action spaces, and presents the first framework for high-dimensional Gaussian process learning from preference feedback to our knowledge.

Both the CoSpar and LineCoSpar algorithms are deployed in human subject ex-

periments with the Atalante exoskeleton to identify gaits that optimize the users' comfort. In particular, the LineCoSpar algorithm optimizes exoskeleton walking over a six-dimensional gait parameter space for six subjects. These algorithms achieve state-of-the-art performance in converging to personalized, user-preferred exoskeleton gaits within relatively few trials. This work provides the first demonstration of preference-based learning for optimizing dynamic crutchless exoskeleton walking.

6.2 Future Work

Theoretical Analysis of Preference-Based RL

There are many interesting directions for extending the work on DPS. Because the current regret analysis relies upon a conjectured upper bound for the information ratio, deriving an analytical information ratio upper bound remains an unsolved problem. It could also be interesting to develop techniques for relating the Bayesian regret and information ratio when DPS uses a sampling distribution that is not an exact Bayesian posterior.

Meanwhile, the presented regret analysis specifically considers the linear link function. Therefore, another direction involves studying regret under other models for credit assignment and user feedback generation, for instance using Bayesian logistic regression or the Gaussian process credit assignment models discussed in Appendix A. More broadly, I expect that DPS would likely perform well with any asymptotically-consistent reward model that sufficiently captures users' preference behavior. Notably, Theorem 2, which relates the Bayesian regret and information ratio, does not depend on a specific credit assignment model. Moreover, the techniques for estimating the information ratio in simulation could be straightforwardly adapted to other credit assignment models and link functions.

Another future direction involves estimating information ratio values for preference-based RL with unknown MDP transition dynamics. In contrast to the current procedure for estimating the information ratio (Algorithm 11), this would require drawing samples from the transition dynamics posterior in addition to the utility posterior. These dynamics samples would be used to estimate the occupancy vectors corresponding to each policy (i.e., the expected number of visits to each state-action pair in a trajectory roll-out), which are needed to estimate the policies' posterior utilities. Furthermore, to determine each policy's posterior probability of optimality, one would need to sample a large number of transition dynamics and utility parameters from their respective posteriors. For each sampled environment, consisting of both

sampled dynamics and utility parameters, one would perform value iteration to determine the optimal policy under that sample. Such results would be of interest, as there is currently no work on applying the information-theoretical regret analysis techniques from Russo and Van Roy (2016) to the RL setting with unknown MDP transition dynamics.

Finally, simulating the information ratio in the RL setting is very computationally intensive for all but the smallest MDPs, as the number of deterministic policies firstly grows exponentially with the state space size and episode length, and secondly is governed by a power relationship with respect to the number of actions. Therefore, developing new techniques to estimate this ratio could also be beneficial.

Preference-Based Learning in More General Settings

DPS assumes that the users' preferences are explained by underlying utilities, and that these utilities govern their preferences via standard link functions. It would be helpful to better understand in what situations these modeling assumptions are valid, and to develop a more general framework that relaxes them. Meanwhile, our team is developing an extension to DPS that is tractable with larger state and action spaces, by incorporating kernelized input spaces and methods such as continuous value iteration (Deisenroth, Rasmussen, and Peters, 2009) to further improve sample efficiency.

Exoskeleton Gait Optimization

Future steps for exoskeleton gait optimization include conducting studies involving patients with paraplegia, whose preferences likely differ from those of able-bodied subjects. Additionally, user preferences could change over time; for instance, in the exoskeleton experiments, we observed that able-bodied users often prefer slower walking gaits when they are less accustomed to walking inside of an exoskeleton. Thus, creating algorithms that account for such adaptations is another important direction for future study. Thirdly, the CoSpar and LineCoSpar frameworks could also be extended beyond working with precomputed gait libraries to generate entirely new gaits or controller designs.

The CoSpar and LineCoSpar algorithms both aim to minimize the regret of online learning by converging to the optimal gait as quickly as possible; therefore, over time, their gait samples become increasingly biased toward user-preferred gaits. For instance, Figure 5.4 illustrates that in synthetic function simulations with 150 iterations, CoSpar's learned objective function posterior consists primarily of a sharp

peak at the optimum. While this behavior is desirable for the gait optimization problem, it does not yield data that helps to understand the mechanisms underlying the users' preferences. Since developing CoSpar and LineCoSpar, our team has developed an active learning approach (ROIAL) that aims to learn the user's preference landscape as accurately as possible, rather than converging to the optimal gait (Li, Tucker, et al., 2020). This extension takes steps toward better understanding the underlying mechanics of user-preferred walking and toward gaining insight into the science of walking with respect to exoskeleton gait design.

Mixed-Initiative Systems

The work on CoSpar and LineCoSpar develops a mixed-initiative system that integrates preference and coactive feedback. It would be interesting to further study the interaction between various types of user feedback in such systems, for instance when different types of feedback queries—preferences, suggestions, ordinal feedback, demonstrations, etc.—should be used, and how their combination impacts the learning process. Additionally, while developing mixed-initiative approaches, it is important to construct theoretical analysis frameworks that account for integrating multiple user feedback modalities. One could also develop frameworks that decide among several different types of feedback queries at every step in order to extract the most information under a limited query budget.

Human Feedback Modeling

The DPS, CoSpar, and LineCoSpar algorithms all learn a Bayesian posterior over the utility function underlying the users' feedback. To use such Bayesian approaches under various types of qualitative feedback, algorithms must robustly model the processes by which users generate such feedback. With coactive feedback, for instance, it is not currently clear how to model the processes behind its generation, and furthermore, these processes could vary depending upon specific application domains and could also depend upon on human psychology. In addition, as mentioned previously, the users' feedback could change over time.

Finally, sample efficiency could be further maximized via learning systems capable not only of individualizing their interactions for different human users, but also of generalizing across users, leveraging existing user feedback to accelerate the learning process for new users.

BIBLIOGRAPHY

Abbasi-Yadkori, Yasin, Dávid Pál, and Csaba Szepesvári. "Improved algorithms for linear stochastic bandits." In: *Conference on Neural Information Processing Systems*. 2011, pp. 2312–2320.

Abeille, Marc and Alessandro Lazaric. "Linear Thompson sampling revisited." In: *Electronic Journal of Statistics* 11.2 (2017), pp. 5165–5197.

Agrawal, Ayush, Omar Harib, et al. "First steps towards translating HZD control of bipedal robots to decentralized control of exoskeletons." In: *IEEE Access* 5 (2017), pp. 9919–9934.

Agrawal, Shipra and Navin Goyal. "Analysis of Thompson sampling for the multi-armed bandit problem." In: *Conference on Learning Theory (COLT)*. 2012, pp. 39–1.

– "Thompson sampling for contextual bandits with linear payoffs." In: *International Conference on Machine Learning*. 2013, pp. 127–135.

Agrawal, Shipra and Randy Jia. "Optimistic posterior sampling for reinforcement learning: Worst-case regret bounds." In: *Conference on Neural Information Processing Systems*. 2017, pp. 1184–1194.

Ailon, Nir, Zohar Karnin, and Thorsten Joachims. "Reducing dueling bandits to cardinal bandits." In: *International Conference on Machine Learning*. 2014, pp. 856–864.

Akrour, Riad, Marc Schoenauer, and Michèle Sebag. "APRIL: Active preference learning-based reinforcement learning." In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2012, pp. 116–131.

Akrour, Riad, Marc Schoenauer, Michèle Sebag, and Jean-Christophe Souplet. "Programming by feedback." In: *International Conference on Machine Learning*. 2014, pp. 1503–1511.

Ames, Aaron D. "Human-inspired control of bipedal walking robots." In: *IEEE Transactions on Automatic Control* 59.5 (2014), pp. 1115–1130.

Amodei, Dario et al. "Concrete problems in AI safety." In: *arXiv preprint arXiv: 1606.06565* (2016).

Argall, Brenna D. et al. "A survey of robot learning from demonstration." In: *Robotics and Autonomous Systems* 57.5 (2009), pp. 469–483.

Armour, Brian S. et al. "Prevalence and causes of paralysis—United States, 2013." In: *American Journal of Public Health* 106.10 (2016), pp. 1855–1857.

Audibert, Jean-Yves and Sébastien Bubeck. "Regret bounds and minimax policies under partial monitoring." In: *The Journal of Machine Learning Research* 11 (2010), pp. 2785–2836.

Audibert, Jean-Yves, Rémi Munos, and Csaba Szepesvári. "Exploration–exploitation tradeoff using variance estimates in multi-armed bandits." In: *Theoretical Computer Science* 410.19 (2009), pp. 1876–1902.

Auer, Peter, Nicolo Cesa-Bianchi, and Paul Fischer. "Finite-time analysis of the multiarmed bandit problem." In: *Machine Learning* 47.2-3 (2002), pp. 235–256.

Bainov, Dromi D. and Pavel S. Simeonov. *Systems with Impulse Effect: Stability, Theory and Applications*. John Wiley & Sons, 1989.

Bartlett, Peter L. and Ambuj Tewari. "REGAL: A regularization based algorithm for reinforcement learning in weakly communicating MDPs." In: *Conference on Uncertainty in Artificial Intelligence*. 2012.

Basu, Chandrayee, Erdem Bıyık, et al. "Active learning of reward dynamics from hierarchical queries." In: *IEEE International Conference on Intelligent Robots and Systems (IROS)*. 2019, pp. 120–127.

Basu, Chandrayee, Qian Yang, et al. "Do you want your autonomous car to drive like you?" In: *ACM/IEEE International Conference on Human-Robot Interaction*. IEEE. 2017, pp. 417–425.

Billingsley, P. *Convergence of Probability Measures*. John Wiley & Sons, 1968.

Bıyık, Erdem, Nicolas Huynh, et al. "Active preference-based Gaussian process regression for reward learning." In: *arXiv preprint arXiv:2005.02575* (2020).

Bıyık, Erdem, Malayandi Palan, et al. "Asking Easy Questions: A User-Friendly Approach to Active Reward Learning." In: *Conference on Robot Learning*. 2020, pp. 1177–1190.

Bridson, Robert. "Fast Poisson disk sampling in arbitrary dimensions." In: *SIGGRAPH Sketches* 10 (2007), p. 1.

Brochu, Eric, Tyson Brochu, and Nando de Freitas. "A Bayesian interactive optimization approach to procedural animation design." In: *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 2010, pp. 103–112.

Brochu, Eric, Vlad M. Cora, and Nando de Freitas. "A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning." In: *arXiv preprint arXiv:1012.2599* (2010).

Brochu, Eric, Nando de Freitas, and Abhijeet Ghosh. "Active preference learning with discrete choice data." In: *Conference on Neural Information Processing Systems*. 2008, pp. 409–416.

Brost, Brian et al. “Multi-dueling bandits and their application to online ranker evaluation.” In: *ACM Conference on Information and Knowledge Management*. 2016, pp. 2161–2166.

Bubeck, Sébastien and Che-Yu Liu. “Prior-free and prior-dependent regret bounds for Thompson sampling.” In: *Conference on Neural Information Processing Systems*. 2013, pp. 638–646.

Burges, Christopher, Tal Shaked, et al. “Learning to rank using gradient descent.” In: *International Conference on Machine Learning*. 2005, pp. 89–96.

Burges, Christopher J., Robert Ragno, and Quoc V. Le. “Learning to rank with nonsmooth cost functions.” In: *Conference on Neural Information Processing Systems*. 2007, pp. 193–200.

Busa-Fekete, Róbert et al. “Preference-based reinforcement learning: Evolutionary direct policy search using a preference-based racing algorithm.” In: *Machine Learning* 97.3 (2014), pp. 327–351.

Cesa-Bianchi, Nicolo and Gábor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

Chapelle, Olivier, Thorsten Joachims, et al. “Large-scale validation and analysis of interleaved search evaluation.” In: *ACM Transactions on Information Systems (TOIS)* 30.1 (2012), 6:1–6:41.

Chapelle, Olivier and Lihong Li. “An empirical evaluation of Thompson sampling.” In: *Conference on Neural Information Processing Systems*. 2011, pp. 2249–2257.

Chowdhury, Sayak Ray and Aditya Gopalan. “On kernelized multi-armed bandits.” In: *International Conference on Machine Learning*. 2017, pp. 844–853.

Christiano, Paul F. et al. “Deep reinforcement learning from human preferences.” In: *Conference on Neural Information Processing Systems*. 2017, pp. 4299–4307.

Chu, Wei and Zoubin Ghahramani. “Gaussian processes for ordinal regression.” In: *Journal of Machine Learning Research* 6 (2005), pp. 1019–1041.

– “Preference learning with Gaussian processes.” In: *International Conference on Machine Learning*. 2005, pp. 137–144.

Chua, Kurtland et al. “Deep reinforcement learning in a handful of trials using probabilistic dynamics models.” In: *Conference on Neural Information Processing Systems*. 2018, pp. 4754–4765.

Clark, Jack and Dario Amodei. *Faulty Reward Functions in the Wild*. `https://openai.com/blog/faulty-reward-functions/`. 2016.

Cover, Thomas M. and Joy A. Thomas. *Elements of Information Theory*. John Wiley & Sons, 2012.

Da, Xingye, Ross Hartley, and Jessy W. Grizzle. "Supervised learning for stabilizing underactuated bipedal robot locomotion, with outdoor experiments on the wave field." In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 3476–3483.

Dani, Varsha, Thomas P. Hayes, and Sham M. Kakade. "Stochastic linear optimization under bandit feedback." In: *Conference On Learning Theory*. 2008, pp. 355–366.

Dann, Christoph and Emma Brunskill. "Sample complexity of episodic fixed-horizon reinforcement learning." In: *Conference on Neural Information Processing Systems*. 2015, pp. 2818–2826.

Deisenroth, Marc Peter, Carl Edward Rasmussen, and Jan Peters. "Gaussian process dynamic programming." In: *Neurocomputing* 72.7-9 (2009), pp. 1508–1524.

Dollar, Aaron M. and Hugh Herr. "Lower extremity exoskeletons and active orthoses: Challenges and state-of-the-art." In: *IEEE Transactions on Robotics* 24.1 (2008), pp. 144–158.

Donati, Ana R. C. et al. "Long-term training with a brain-machine interface-based gait protocol induces partial neurological recovery in paraplegic patients." In: *Scientific Reports* 6 (2016), p. 30383.

Dong, Shi and Benjamin Van Roy. "An information-theoretic analysis for Thompson sampling with many actions." In: *Conference on Neural Information Processing Systems*. 2018, pp. 4157–4165.

Duburcq, Alexis et al. "Online trajectory planning through combined trajectory optimization and function approximation: Application to the exoskeleton Atalante." In: *arXiv preprint arXiv:1910.00514* (2019).

Dunbar, Daniel and Greg Humphreys. *Using scalloped sectors to generate Poisson-disk sampling patterns*. University of Virginia, Department of Computer Science. 2006.

Ebert, Frederik et al. "Visual foresight: Model-based deep reinforcement learning for vision-based robotic control." In: *arXiv preprint arXiv:1812.00568* (2018).

Ekso Bionics. `https://eksobionics.com/`, Last accessed on 2019-09-14.

Facts and Figures at a Glance. National Spinal Cord Injury Statistical Center. Birmingham, AL: University of Alabama at Birmingham. `https://www.sci-info-pages.com/wp-content/media/NSCISC-2019-Spinal-Cord-Injury-Facts-and-Figures-at-a-Glance.pdf`. 2019.

Filippi, Sarah et al. "Parametric bandits: The generalized linear case." In: *Conference on Neural Information Processing Systems*. 2010, pp. 586–594.

Foreman-Mackey, Daniel et al. "emcee: The MCMC hammer." In: *Publications of the Astronomical Society of the Pacific* 125.925 (2013), p. 306.

Frazier, Peter I., Warren B. Powell, and Savas Dayanik. "A knowledge-gradient policy for sequential information collection." In: *SIAM Journal on Control and Optimization* 47.5 (2008), pp. 2410–2439.

Fürnkranz, Johannes and Eyke Hüllermeier. *Preference Learning*. Springer, 2010.

Fürnkranz, Johannes, Eyke Hüllermeier, et al. "Preference-based reinforcement learning: A formal framework and a policy iteration algorithm." In: *Machine Learning* 89.1-2 (2012), pp. 123–156.

Gad, Parag et al. "Weight bearing over-ground stepping in an exoskeleton with non-invasive spinal cord neuromodulation after motor complete paraplegia." In: *Frontiers in Neuroscience* 11 (2017), p. 333.

Garivier, Aurélien and Olivier Cappé. "The KL-UCB algorithm for bounded stochastic bandits and beyond." In: *Proceedings of the 24th Annual Conference on Learning Theory*. 2011, pp. 359–376.

Gittins, John. "A dynamic allocation index for the sequential design of experiments." In: *Progress in Statistics* (1974), pp. 241–266.

– "Bandit processes and dynamic allocation indices." In: *Journal of the Royal Statistical Society: Series B (Methodological)* 41.2 (1979), pp. 148–164.

Goemaere, Stefan et al. "Bone mineral status in paraplegic patients who do or do not perform standing." In: *Osteoporosis International* 4.3 (1994), pp. 138–143.

Golovin, Daniel and Andreas Krause. "Adaptive submodularity: Theory and applications in active learning and stochastic optimization." In: *Journal of Artificial Intelligence Research* 42 (2011), pp. 427–486.

Gopalan, Aditya and Shie Mannor. "Thompson sampling for learning parameterized Markov decision processes." In: *Conference on Learning Theory*. 2015, pp. 861–898.

Gourieroux, Christian and Alain Monfort. "Asymptotic properties of the maximum likelihood estimator in dichotomous logit models." In: *Journal of Econometrics* 17.1 (1981), pp. 83–97.

Granmo, Ole-Christoffer. "Solving two-armed Bernoulli bandit problems using a Bayesian learning automaton." In: *International Journal of Intelligent Computing and Cybernetics* 3.2 (2010), p. 207.

Grimmett, Geoffrey R. and David R. Stirzaker. *Probability and Random Processes*. Oxford University Press, 2001.

Gupta, Shanti S. and Klaus J. Miescke. "Bayesian look ahead one-stage sampling allocations for selection of the best population." In: *Journal of Statistical Planning and Inference* 54.2 (1996), pp. 229–244.

Gurriet, Thomas, Sylvain Finet, et al. "Towards restoring locomotion for paraplegics: Realizing dynamically stable walking on exoskeletons." In: *International Conference on Robotics and Automation*. IEEE. 2018, pp. 2804–2811.

Gurriet, Thomas, Maegan Tucker, et al. "Towards variable assistance for lower body exoskeletons." In: *IEEE Robotics and Automation Letters* 5.1 (2019), pp. 266–273.

Harib, Omar et al. "Feedback control of an exoskeleton for paraplegics: Toward robustly stable, hands-free dynamic walking." In: *IEEE Control Systems Magazine* 38.6 (2018), pp. 61–87.

Hazan, Elad and Kfir Levy. "Bandit convex optimization: Towards tight bounds." In: *Conference on Neural Information Processing Systems*. 2014, pp. 784–792.

Hennig, Philipp and Christian J. Schuler. "Entropy search for information-efficient global optimization." In: *The Journal of Machine Learning Research* 13.1 (2012), pp. 1809–1837.

Herbrich, Ralf, Thore Graepel, and Klaus Obermayer. "Support vector learning for ordinal regression." In: *International Conference on Artificial Neural Networks*. IET. 1999, pp. 97–102.

Hoffman, Matthew, Eric Brochu, and Nando de Freitas. "Portfolio allocation for Bayesian optimization." In: *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*. 2011, pp. 327–336.

Hong, Xiaodan et al. "A weighted Gaussian process regression for multivariate modelling." In: *2017 6th International Symposium on Advanced Control of Industrial Processes (AdCONIP)*. IEEE. 2017, pp. 195–200.

Houlsby, Neil et al. "Bayesian active learning for classification and preference learning." In: *arXiv preprint arXiv:1112.5745* (2011).

Ibarz, Borja et al. "Reward learning from human preferences and demonstrations in Atari." In: *Conference on Neural Information Processing Systems*. 2018, pp. 8011–8023.

Indego. `http://www.indego.com/indego/en/home`, Last accessed on 2019-09-14.

Jain, Ashesh, Shikhar Sharma, et al. "Learning preferences for manipulation tasks from online coactive feedback." In: *The International Journal of Robotics Research* 34.10 (2015), pp. 1296–1313.

Jain, Shomik, Balasubramanian Thiagarajan, et al. "Modeling engagement in long-term, in-home socially assistive robot interventions for children with autism spectrum disorders." In: *Science Robotics* 5.39 (2020). DOI: `10.1126/scirobotics.aaz3791`. eprint: `https://robotics.sciencemag.org/content/5/39/eaaz3791.full.pdf`. URL: `https://robotics.sciencemag.org/content/5/39/eaaz3791`.

Jaksch, Thomas, Ronald Ortner, and Peter Auer. "Near-optimal regret bounds for reinforcement learning." In: *Journal of Machine Learning Research* 11 (2010), pp. 1563–1600.

Joachims, Thorsten et al. “Accurately interpreting clickthrough data as implicit feedback.” In: *SIGIR*. 2005, pp. 154–161.

Jones, Donald R., Matthias Schonlau, and William J. Welch. “Efficient global optimization of expensive black-box functions.” In: *Journal of Global Optimization* 13.4 (1998), pp. 455–492.

Jordan, Michael Irwin. *Learning in Graphical Models*. The MIT Press, 1999.

Kaiser, Lukasz et al. “Model-based reinforcement learning for Atari.” In: *arXiv preprint arXiv:1903.00374* (2019). Published as a conference paper at the International Conference on Learning Representations (ICLR), 2020.

Kanagawa, Motonobu et al. “Gaussian processes and kernel methods: A review on connections and equivalences.” In: *arXiv preprint arXiv:1807.02582* (2018).

Kaufmann, Emilie, Olivier Cappé, and Aurélien Garivier. “On Bayesian upper confidence bounds for bandit problems.” In: *International Conference on Artificial Intelligence and Statistics*. 2012, pp. 592–600.

Kearns, Michael and Satinder Singh. “Near-optimal reinforcement learning in polynomial time.” In: *Machine Learning* 49.2-3 (2002), pp. 209–232.

Kim, Myunghee et al. “Human-in-the-loop Bayesian optimization of wearable device parameters.” In: *PloS One* 12.9 (2017), e0184054.

Kirschner, Johannes and Andreas Krause. “Information directed sampling and bandits with heteroscedastic noise.” In: *Conference on Learning Theory*. 2018, pp. 358–384.

Kirschner, Johannes, Mojmir Mutny, et al. “Adaptive and safe Bayesian optimization in high dimensions via one-dimensional subspaces.” In: *International Conference on Machine Learning*. 2019, pp. 3429–3438.

Kleinberg, Robert, Aleksandrs Slivkins, and Eli Upfal. “Multi-armed bandits in metric spaces.” In: *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*. 2008, pp. 681–690.

Kolter, J. Zico and Andrew Y. Ng. “Near-Bayesian exploration in polynomial time.” In: *Proceedings of the 26th Annual International Conference on Machine Learning*. 2009, pp. 513–520.

Komiyama, Junpei et al. “Regret lower bound and optimal algorithm in dueling bandit problem.” In: *Conference on Learning Theory*. 2015, pp. 1141–1154.

Kupcsik, Andras, David Hsu, and Wee Sun Lee. “Learning dynamic robot-to-human object handover from human feedback.” In: *Robotics Research*. Springer, 2018, pp. 161–176.

Lai, Tze Leung and Herbert Robbins. “Asymptotically efficient adaptive allocation rules.” In: *Advances in Applied Mathematics* 6.1 (1985), pp. 4–22.

Lattimore, Tor and Marcus Hutter. "PAC bounds for discounted MDPs." In: *International Conference on Algorithmic Learning Theory*. Springer. 2012, pp. 320–334.

Lester, James C., Brian A. Stone, and Gary D. Stelling. "Lifelike pedagogical agents for mixed-initiative problem solving in constructivist learning environments." In: *User Modeling and User-Adapted Interaction* 9.1-2 (1999), pp. 1–44.

Li, Cheng, Sunil Gupta, et al. "High dimensional Bayesian optimization using dropout." In: *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 2017, pp. 2096–2102.

Li, Kejun, Maegan Tucker, et al. "ROIAL: Region of Interest Active Learning for Characterizing Exoskeleton Gait Preference Landscapes." In: *arXiv preprint arXiv:2011.04812* (2020).

Lillicrap, Timothy P. et al. "Continuous control with deep reinforcement learning." In: *arXiv preprint arXiv:1509.02971* (2015). Published as a conference paper at the International Conference on Learning Representations (ICLR), 2016.

Liu, Tie-Yan. "Learning to rank for information retrieval." In: *Foundations and Trends® in Information Retrieval* 3.3 (2009), pp. 225–331.

May, Benedict C., Nathan Korda, et al. "Optimistic Bayesian sampling in contextual-bandit problems." In: *The Journal of Machine Learning Research* 13.1 (2012), pp. 2069–2106.

May, Benedict C. and David S. Leslie. "Simulation studies in optimistic Bayesian sampling in contextual-bandit problems." In: *Statistics Group, Department of Mathematics, University of Bristol* 11.02 (2011).

McCullagh, Peter and John A Nelder. *Generalized Linear Models*. 2nd Edition. London, UK: Chapman and Hall, 1989.

Mehrholz, Jan et al. "Electromechanical-assisted training for walking after stroke (review)." In: *Cochrane Database of Systematic Reviews* 5 (2017). John Wiley & Sons, pp. 1–125.

Metropolis, Nicholas et al. "Equation of state calculations by fast computing machines." In: *The Journal of Chemical Physics* 21.6 (1953), pp. 1087–1092.

Minka, Thomas and John Lafferty. "Expectation-propagation for the generative aspect model." In: *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence*. 2002, pp. 352–359.

Minka, Thomas P. "Expectation propagation for approximate Bayesian inference." In: *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*. 2001, pp. 362–369.

Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Alex Graves, et al. "Playing Atari with deep reinforcement learning." In: *arXiv preprint arXiv:1312.5602* (2013).

Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, et al. "Human-level control through deep reinforcement learning." In: *Nature* 518.7540 (2015), pp. 529–533.

Murphy, Kevin P. *Conjugate Bayesian Analysis of the Gaussian Distribution*. Tech. rep. University of British Columbia, 2007.

– *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

Neu, Gergely and Gábor Bartók. "An efficient algorithm for learning with semi-bandit feedback." In: *International Conference on Algorithmic Learning Theory*. Springer. 2013, pp. 234–248.

Nikolov, Nikolay et al. "Information-directed exploration for deep reinforcement learning." In: *arXiv preprint arXiv:1812.07544* (2018). Published as a conference paper at the International Conference on Learning Representations (ICLR), 2019.

Novoseller, Ellen R. et al. *Dueling Posterior Sampling for Preference-Based Reinforcement Learning*. Code for the DPS algorithm. Available at: `https://github.com/ernovoseller/DuelingPosteriorSampling`. 2020.

– "Dueling posterior sampling for preference-based reinforcement learning." In: *Conference on Uncertainty in Artificial Intelligence (UAI)*. PMLR. 2020, pp. 1029–1038. URL: `http://proceedings.mlr.press/v124/novoseller20a.html`.

Osband, Ian, Charles Blundell, et al. "Deep exploration via bootstrapped DQN." In: *Conference on Neural Information Processing Systems*. 2016, pp. 4026–4034.

Osband, Ian, Daniel Russo, and Benjamin Van Roy. "(More) efficient reinforcement learning via posterior sampling." In: *Conference on Neural Information Processing Systems*. 2013, pp. 3003–3011.

Osband, Ian and Benjamin Van Roy. "Why is posterior sampling better than optimism for reinforcement learning?" In: *International Conference on Machine Learning*. 2017, pp. 2701–2710.

Osband, Ian, Benjamin Van Roy, and Zheng Wen. "Generalization and exploration via randomized value functions." In: *International Conference on Machine Learning*. 2016, pp. 2377–2386.

Ouyang, Yi et al. "Learning unknown Markov decision processes: A Thompson sampling approach." In: *Conference on Neural Information Processing Systems*. 2017, pp. 1333–1342.

Payne, John W. et al. *The Adaptive Decision Maker*. Cambridge University Press, 1993.

Plaat, Aske, Walter Kosters, and Mike Preuss. "Model-based deep reinforcement learning for high-dimensional problems, a survey." In: *arXiv preprint arXiv:2008.05598* (2020).

Popov, Ivaylo et al. "Data-efficient deep reinforcement learning for dexterous manipulation." In: *arXiv preprint arXiv:1704.03073* (2017).

Radlinski, Filip and Thorsten Joachims. "Query chains: Learning to rank from implicit feedback." In: *ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*. ACM. 2005, pp. 239–248.

Raman, Karthik et al. "Stable coactive learning via perturbation." In: *International Conference on Machine Learning*. 2013, pp. 837–845.

Rasmussen, Carl Edward and Christopher K. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

Reher, Jacob P. et al. "Algorithmic foundations of realizing multi-contact locomotion on the humanoid robot DURUS." In: *Algorithmic Foundations of Robotics XII*. Springer, Cham, 2020, pp. 400–415. URL: http://ames.caltech.edu/reher2020algorithmic.pdf.

Ren, Shixin et al. "Personalized gait trajectory generation based on anthropometric features using Random Forest." In: *Journal of Ambient Intelligence and Humanized Computing* (2019), pp. 1–12. DOI: https://doi.org/10.1007/s12652-019-01390-3.

ReWalk. https://rewalk.com/, Last accessed on 2019-09-14.

Rex Bionics. https://www.rexbionics.com/, Last accessed on 2019-09-14.

Robbins, Herbert. "Some aspects of the sequential design of experiments." In: *Bulletin of the American Mathematical Society* 58.5 (1952), pp. 527–535.

Ruan, Xiaogang et al. "Mobile robot navigation based on deep reinforcement learning." In: *2019 Chinese Control and Decision Conference (CCDC)*. IEEE. 2019, pp. 6174–6178.

Rusmevichientong, Paat and John N. Tsitsiklis. "Linearly parameterized bandits." In: *Mathematics of Operations Research* 35.2 (2010), pp. 395–411.

Russo, Daniel and Benjamin Van Roy. "An information-theoretic analysis of Thompson sampling." In: *The Journal of Machine Learning Research* 17.1 (2016), pp. 2442–2471.

– "Learning to optimize via information-directed sampling." In: *Conference on Neural Information Processing Systems*. 2014, pp. 1583–1591.

– "Learning to optimize via posterior sampling." In: *Mathematics of Operations Research* 39.4 (2014), pp. 1221–1243.

Ryzhov, Ilya O., Warren B. Powell, and Peter I. Frazier. "The knowledge gradient algorithm for a general class of online learning problems." In: *Operations Research* 60.1 (2012), pp. 180–195.

Sadigh, Dorsa et al. "Active preference-based learning of reward functions." In: *Conference on Robotics: Science and Systems XIII*. July 12 - July 16, 2017, Cambridge, Massachusetts, USA. 2017. URL: http://www.roboticsproceedings.org/rss13/p53.pdf.

Saha, Ankan and Ambuj Tewari. "Improved regret guarantees for online smooth convex optimization with bandit feedback." In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. 2011, pp. 636–642.

Schuth, Anne, Harrie Oosterhuis, et al. "Multileave gradient descent for fast online learning to rank." In: *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*. 2016, pp. 457–466.

Schuth, Anne, Floor Sietsma, et al. "Multileaved comparisons for fast online evaluation." In: *Proceedings of the 23rd ACM International Conference on Information and Knowledge Management*. 2014, pp. 71–80.

Shivaswamy, Pannaga and Thorsten Joachims. "Coactive learning." In: *Journal of Artificial Intelligence Research* 53 (2015), pp. 1–40.

– "Online structured prediction via coactive learning." In: *International Conference on Machine Learning*. 2012, pp. 59–66.

Somers, Thane and Geoffrey A. Hollinger. "Human–robot planning and learning for marine data collection." In: *Autonomous Robots* 40.7 (2016), pp. 1123–1137.

Srinivas, Niranjan et al. "Gaussian process optimization in the bandit setting: No regret and experimental design." In: *International Conference on Machine Learning (ICML)*. 2010, pp. 1015–1022.

Stroke Facts. Centers for Disease Control and Prevention. `https://www.cdc.gov/stroke/facts.htm`. 2020.

Sui, Yanan and Joel W. Burdick. "Clinical online recommendation with subgroup rank feedback." In: *Proceedings of the 8th ACM Conference on Recommender Systems*. 2014, pp. 289–292.

Sui, Yanan, Yisong Yue, and Joel W. Burdick. "Correlational dueling bandits with application to clinical treatment in large decision spaces." In: *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*. 2017, pp. 2793–2799.

Sui, Yanan, Vincent Zhuang, et al. "Multi-dueling bandits with dependent arms." In: *Proceedings of the Conference on Uncertainty in Artificial Intelligence*. Aug. 11-15, 2017, Sydney, Australia. 2017. URL: `http://auai.org/uai2017/proceedings/papers/155.pdf`.

– "Stagewise safe Bayesian optimization with Gaussian processes." In: *International Conference on Machine Learning*. 2018, pp. 4781–4789.

Sui, Yanan, Masrour Zoghi, et al. "Advancements in dueling bandits." In: *International Joint Conference on Artificial Intelligence*. 2018, pp. 5502–5510.

Sutton, Richard S. and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 2018.

Szörényi, Balázs et al. “Online rank elicitation for Plackett-Luce: A dueling bandits approach.” In: *Conference on Neural Information Processing Systems*. 2015, pp. 604–612.

Thatte, Nitish, Helei Duan, and Hartmut Geyer. “A method for online optimization of lower limb assistive devices with high dimensional parameter spaces.” In: *International Conference on Robotics and Automation*. IEEE. 2018, pp. 1–6.

Thompson, William R. “On the likelihood that one unknown probability exceeds another in view of the evidence of two samples.” In: *Biometrika* 25.3/4 (1933), pp. 285–294.

Tucker, Maegan, Myra Cheng, et al. *Human Preference-Based Learning for High-dimensional Optimization of Exoskeleton Walking Gaits*. Video of the experimental results for Tucker, Cheng, et al., “Human preference-based learning for high-dimensional optimization of exoskeleton walking gaits,” IROS. Available at: https://www.youtube.com/watch?v=c6a0kXMyML0&feature=youtu.be. 2020.

– “Human preference-based learning for high-dimensional optimization of exoskeleton walking gaits.” In: *IEEE International Conference on Intelligent Robots and Systems (IROS)*. 2020. URL: https://arxiv.org/pdf/2003.06495.pdf.

– *LineCoSpar*. Code for the LineCoSpar algorithm. Available at: https://github.com/myracheng/linecospar. 2020.

Tucker, Maegan, Ellen R. Novoseller, et al. *CoSpar: Online Learning from Human Preference and Coactive Feedback*. Code for the CoSpar algorithm. Available at: https://github.com/ernovoseller/CoSpar. 2020.

– “Preference-based learning for exoskeleton gait optimization.” In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2020. DOI: 10.1109/ICRA40945.2020.9196661. URL: https://ieeexplore.ieee.org/document/9196661.

– *Preference-Based Learning on Lower-Body Exoskeletons*. Video of the experimental results for Tucker, Novoseller, et al., “Preference-based learning for exoskeleton gait optimization,” ICRA. Available at: https://www.youtube.com/watch?v=-27sHXsvONE&feature=youtu.be. 2020.

Urvoy, Tanguy et al. “Generic exploration and k-armed voting bandits.” In: *International Conference on Machine Learning (ICML)*. 2013, pp. 91–99.

Wainwright, Martin J. *High-Dimensional Statistics: A Non-Asymptotic Viewpoint*. Vol. 48. Cambridge University Press, 2019.

Wainwright, Martin J. and Michael I. Jordan. “Graphical models, exponential families, and variational inference.” In: *Machine Learning* 1.1-2 (2008), pp. 1–305.

Wandercraft. http://www.wandercraft.eu/, Last accessed on 2017-09-15.

Wang, Ziyu et al. “Bayesian optimization in a billion dimensions via random embeddings.” In: *Journal of Artificial Intelligence Research* 55 (2016), pp. 361–387.

Westervelt, Eric R., Jessy W. Grizzle, Christine Chevallereau, et al. *Feedback Control of Dynamic Bipedal Robot Locomotion*. CRC press, 2018.

Westervelt, Eric R., Jessy W. Grizzle, and Daniel E. Koditschek. “Hybrid zero dynamics of planar biped walkers.” In: *IEEE Transactions on Automatic Control* 48.1 (2003), pp. 42–56.

Wilson, Aaron, Alan Fern, and Prasad Tadepalli. “A Bayesian approach for policy learning from trajectory preference queries.” In: *Conference on Neural Information Processing Systems*. 2012, pp. 1133–1141.

Wirth, Christian. “Efficient Preference-based Reinforcement Learning.” PhD thesis. Technische Universität, 2017.

Wirth, Christian, Riad Akrour, et al. “A survey of preference-based reinforcement learning methods.” In: *The Journal of Machine Learning Research* 18.1 (2017), pp. 4945–4990.

Wirth, Christian and Johannes Fürnkranz. “A policy iteration algorithm for learning from preference-based feedback.” In: *International Symposium on Intelligent Data Analysis*. Springer. 2013, pp. 427–437.

– “EPMC: Every visit preference Monte Carlo for reinforcement learning.” In: *Asian Conference on Machine Learning*. 2013, pp. 483–497.

Wirth, Christian, Johannes Fürnkranz, and Gerhard Neumann. “Model-free preference-based reinforcement learning.” In: *Thirtieth AAAI Conference on Artificial Intelligence*. 2016, pp. 2222–2228.

Wolff, Jamie et al. “A survey of stakeholder perspectives on exoskeleton technology.” In: *Journal of Neuroengineering and Rehabilitation* 11.1 (2014), p. 169.

Wolfman, Steven A. et al. “Mixed initiative interfaces for learning tasks: SMARTedit talks back.” In: *Proceedings of the 6th International Conference on Intelligent User Interfaces*. ACM. 2001, pp. 167–174.

Wu, Huasen and Xin Liu. “Double Thompson sampling for dueling bandits.” In: *Conference on Neural Information Processing Systems*. 2016, pp. 649–657.

Wu, Xinyu, Du-Xin Liu, et al. “Individualized gait pattern generation for sharing lower limb exoskeleton robot.” In: *IEEE Transactions on Automation Science and Engineering* 15.4 (2018), pp. 1459–1470.

Ye, Hui, Anthony N. Michel, and Ling Hou. “Stability theory for hybrid dynamical systems.” In: *IEEE Transactions on Automatic Control* 43.4 (1998), pp. 461–474.

Yue, Yisong, Josef Broder, et al. “The k-armed dueling bandits problem.” In: *Journal of Computer and System Sciences* 78.5 (2012), pp. 1538–1556.

Yue, Yisong, Thomas Finley, et al. “A support vector method for optimizing average precision.” In: *International SIGIR Conference on Research and Development in Information Retrieval*. ACM. 2007, pp. 271–278.

Yue, Yisong and Thorsten Joachims. “Beat the Mean Bandit.” In: *International Conference on Machine Learning (ICML)*. 2011, pp. 241–248.

– “Interactively optimizing information retrieval systems as a dueling bandits problem.” In: *International Conference on Machine Learning (ICML)*. 2009, pp. 1201–1208.

Zanette, Andrea and Rahul Sarkar. *Information Directed Reinforcement Learning*. Tech. rep. Stanford University, 2017.

Zhang, Juanjuan et al. “Human-in-the-loop optimization of exoskeleton assistance during walking.” In: *Science* 356.6344 (2017), pp. 1280–1284.

Zoghi, Masrour, Zohar S. Karnin, et al. “Copeland dueling bandits.” In: *Conference on Neural Information Processing Systems*. 2015, pp. 307–315.

Zoghi, Masrour, Shimon A. Whiteson, Maarten De Rijke, et al. “Relative confidence sampling for efficient on-line ranker evaluation.” In: *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*. 2014, pp. 73–82.

Zoghi, Masrour, Shimon A. Whiteson, Rémi Munos, et al. “Relative upper confidence bound for the k-armed dueling bandit problem.” In: *International Conference on Machine Learning*. 2014, pp. 10–18.

Zoghi, Masrour, Shimon A. Whiteson, and Maarten de Rijke. “MergeRUCB: A method for large-scale online ranker evaluation.” In: *ACM International Conference on Web Search and Data Mining (WSDM)*. 2015, pp. 17–26.

Appendix A

MODELS FOR UTILITY INFERENCE AND CREDIT ASSIGNMENT

This appendix contains the mathematical details of the credit assignment models evaluated in the Dueling Posterior Sampling (DPS) experiments, presented in Section 4.5. These Bayesian models can be used together with either the preference-based generalized linear bandit or RL settings.

A.1 Bayesian Linear Regression

While Bayesian linear regression was already introduced in Section 4.3, this section briefly reviews the details of the experimental implementation.

Define $X \in \mathbb{R}^{N \times d}$ as the observation matrix after N preferences, in which the i^{th} row contains observation $\boldsymbol{x}_i = \boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}$, while $\boldsymbol{y} \in \mathbb{R}^N$ is the vector of corresponding preference labels, with i^{th} element $y_i \in \left\{-\frac{1}{2}, \frac{1}{2}\right\}$.

Section 4.3 defines the Bayesian linear regression credit assignment models to which the theoretical guarantees apply. Because the $\beta_i(\delta)$ factor necessary for the asymptotic consistency guarantees results in a conservative covariance matrix leading to over-exploration, the DPS simulations implement the more practical variant given in Eq. (4.1) and restated here. A Gaussian prior is defined over the reward vector $\boldsymbol{r} \in \mathbb{R}^d$: $\boldsymbol{r} \sim \mathcal{N}(0, \lambda^{-1} I)$. The likelihood of the data conditioned upon $\boldsymbol{r}$ is also Gaussian:

$$p(\boldsymbol{y}|X, \boldsymbol{r}; \sigma^2) = \frac{1}{(2\pi\sigma^2)^{\frac{N}{2}}} \exp\left(-\frac{1}{2\sigma^2}||\boldsymbol{y} - X\boldsymbol{r}||^2\right).$$

This conjugate prior and likelihood lead to the following closed-form posterior:

$$\boldsymbol{r}|X, \boldsymbol{y}, \sigma^2, \lambda \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma), \text{ where}$$
$$\boldsymbol{\mu} = (X^T X + \sigma^2 \lambda I)^{-1} X^T \boldsymbol{y} \text{ and } \Sigma = \sigma^2 (X^T X + \sigma^2 \lambda I)^{-1}.$$

A.2 Gaussian Process Regression

Credit assignment via Gaussian processes (Rasmussen and Williams, 2006) extends the linear credit assignment model in A.1 to larger numbers of features d by generalizing across similar features. For instance, in the RL setting, one could

learn over larger state and action spaces by generalizing across nearby states and actions. This section and the following one consider two Gaussian process-based credit assignment approaches.

To perform credit assignment via Gaussian process regression, one can assign binary labels to each observation based on whether it is preferred or dominated. A Gaussian process prior is placed upon the underlying utilities $\overline{\boldsymbol{r}}$; for instance, in the RL setting, this prior is placed on the utilities of the individual state-action pairs. Using that a bandit action or RL trajectory's total utility is a sum over the utilities in each dimension (e.g., over each state-action pair in RL), this section shows how to perform inference over sums of Gaussian process variables to infer the component utilities in $\overline{\boldsymbol{r}}$ from the total utilities. As the total utility of each bandit action or RL trajectory is not observed in practice, the obtained binary preference labels are substituted as approximations in their place.

To avoid having to constantly distinguish between the bandit and RL settings, the rest of this section adapts all notation and terminology for preference-based RL. For instance, the dimensions of $\overline{\boldsymbol{r}}$ are referred to as utilities of state-action pairs, while in the bandit setting, they are utility weights corresponding to each dimension of the action space. Similarly, in the RL setting, the observations $\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}$ correspond to trajectory features, while in the bandit setting, these are actions. However, the derived posterior update equations (Eq.s (A.2) and (A.3)) apply as-is to the generalized linear dueling bandit setting, and the two cases are mathematically-identical with respect to the methods introduced in this section.

Let $\{\tilde{s}_1, \ldots, \tilde{s}_d\}$ denote the $d = SA$ state-action pairs. In this section, the data matrix $Z \in \mathbb{R}^{2N \times d}$ holds all state-action visitation vectors $\boldsymbol{x}_{k1}, \boldsymbol{x}_{k2}$, for DPS iterations $k \in \{1, \ldots, N\}$. (This contrasts with the other credit assignment methods, which learn from their differences, $\boldsymbol{x}_{k2} - \boldsymbol{x}_{k1}$.) Let $\boldsymbol{z}_i^T$ be the i^{th} row of Z, such that $Z = [\boldsymbol{z}_1 \ldots, \boldsymbol{z}_{2N}]^T$, and $\boldsymbol{z}_i = \boldsymbol{x}_{kj}$ for some DPS iteration k and $j \in \{1, 2\}$, that is, $\boldsymbol{z}_i$ contains the state-action visit counts for the i^{th} trajectory rollout. In particular, the ij^{th} matrix element $z_{ij} = [Z]_{ij}$ is the number of times that the i^{th} observed trajectory $\boldsymbol{z}_i$ visits state-action $\tilde{s}_j$.

The label vector is $\boldsymbol{y}' \in \mathbb{R}^{2N}$, where the i^{th} element y'_i is the preference label corresponding to the i^{th}-observed trajectory. For instance, if $\boldsymbol{x}_{i2} \succ \boldsymbol{x}_{i1}$, then $\boldsymbol{x}_{i2}$ receives a label of $\frac{1}{2}$, while $\boldsymbol{x}_{i1}$ is labelled $-\frac{1}{2}$. As before, $\overline{r}(\tilde{s})$ denotes the underlying utility of state-action pair $\tilde{s}$, with $u(\tau)$ being trajectory τ's total utility along the

state-action pairs it encounters.[1] To infer $\overline{\boldsymbol{r}}$, each total utility $u(\tau_i)$ is approximated with its preference label y'_i.

A Gaussian process prior is placed upon the rewards $\overline{\boldsymbol{r}}$: $\overline{\boldsymbol{r}} \sim \mathcal{GP}(\boldsymbol{\mu_r}, K_r)$, where $\boldsymbol{\mu_r} \in \mathbb{R}^d$ is the prior mean and $K_r \in \mathbb{R}^{d\times d}$ is the prior covariance matrix, such that $[K_r]_{ij}$ models the prior covariance between $\overline{r}(\tilde{s}_i)$ and $\overline{r}(\tilde{s}_j)$. The total utility of trajectory τ_i, denoted $u(\tau_i)$, is modeled as a sum over the latent state-action utilities: $u(\tau_i) = \sum_{j=1}^{d} z_{ij}\overline{r}(\tilde{s}_j)$. Let R_i be a noisy version of $u(\tau_i)$: $R_i = u(\tau_i) + \varepsilon_i$, where $\varepsilon_i \sim \mathcal{N}(0, \sigma_\varepsilon^2)$ is i.i.d. noise. Then, given rewards $\overline{\boldsymbol{r}}$:

$$R_i = \sum_{j=1}^{d} z_{ij}\overline{r}(\tilde{s}_j) + \varepsilon_i.$$

Because any linear combination of jointly Gaussian variables is Gaussian, R_i is a Gaussian process over the values $\{z_{i1}, \ldots, z_{id}\}$. Let $\boldsymbol{R} \in \mathbb{R}^{2N}$ be the vector with i^{th} element equal to R_i. This section will calculate the relevant expectations and covariances to show that $\overline{\boldsymbol{r}} \sim \mathcal{GP}(\boldsymbol{\mu_r}, K_r)$ and $\boldsymbol{R}$ have the following jointly-Gaussian distribution:

$$\begin{bmatrix} \overline{\boldsymbol{r}} \\ \boldsymbol{R} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu_r} \\ Z\boldsymbol{\mu_r} \end{bmatrix}, \begin{bmatrix} K_r & K_r Z^T \\ ZK_r^T & ZK_rZ^T + \sigma_\varepsilon^2 I \end{bmatrix} \right). \tag{A.1}$$

The standard approach for obtaining a conditional distribution from a joint Gaussian distribution (Rasmussen and Williams, 2006) yields $\overline{\boldsymbol{r}}|\boldsymbol{R} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$, where:

$$\boldsymbol{\mu} = \boldsymbol{\mu_r} + K_r Z^T [ZK_rZ^T + \sigma_\varepsilon^2 I]^{-1}(\boldsymbol{R} - Z\boldsymbol{\mu_r}) \tag{A.2}$$

$$\Sigma = K_r - K_r Z^T [ZK_rZ^T + \sigma_\varepsilon^2 I]^{-1} ZK_r^T. \tag{A.3}$$

In practice, the variable $\boldsymbol{R}$ is not observed. Instead, $\boldsymbol{R}$ is approximated with the observed preference labels $\boldsymbol{y}'$, $\boldsymbol{R} \approx \boldsymbol{y}'$, to perform credit assignment inference.

Next, this section derives the posterior inference equations (A.2) and (A.3) used in Gaussian process regression credit assignment. The state-action rewards $\overline{\boldsymbol{r}}$ are inferred given noisy observations $\boldsymbol{R}$ of the trajectories' total utilities via the following four steps, corresponding to the next four subsections:

[1] The concept of a trajectory's total utility is analogous to a d-dimensional action's utility in the bandit setting, $\overline{\boldsymbol{r}}^T\boldsymbol{x}$ for an action $\boldsymbol{x} \in \mathcal{A}$. A state-action utility $\overline{r}(\tilde{s})$ is equal to a particular component of $\overline{\boldsymbol{r}}$: $\overline{\boldsymbol{r}}^T\boldsymbol{e}_j$ for some j, where $\boldsymbol{e}_j$ is a vector with 1 in the j^{th} component and zeros elsewhere. A state-action utility $\overline{r}(\tilde{s})$ corresponds to the utility weight of an action space dimension in the bandit setting, which is also $\overline{\boldsymbol{r}}^T\boldsymbol{e}_j$ (for some j).

A) Model the state-action utilities $\overline{r}(\tilde{s})$ as a Gaussian process over state-action pairs $\tilde{s}$.

B) Model the trajectory utilities $\boldsymbol{R}$ as a Gaussian process that results from summing the state-action utilities $\overline{r}(\tilde{s})$.

C) Using the two Gaussian processes defined in A) and B), obtain the covariance matrix between the values of $\{\overline{r}(\tilde{s})|\tilde{s} \in 1, \ldots, d\}$ and $\{R_i|i \in 1, \ldots, 2N\}$.

D) Write the joint Gaussian distribution in Eq. (A.1) between the values of $\{\overline{r}(\tilde{s})|\tilde{s} \in 1, \ldots, d\}$ and $\{R_i|i \in 1, \ldots, 2N\}$, and obtain the posterior distribution of $\overline{\boldsymbol{r}}$ over all state-action pairs given $\boldsymbol{R}$ (Eq.s (A.2) and (A.3)).

The State-Action Utility Gaussian Process

The state-action utilities $\overline{\boldsymbol{r}}$ are modeled as a Gaussian process over $\tilde{s}$, with mean $\mathbb{E}[\overline{r}(\tilde{s})] = \mu_r(\tilde{s})$ and covariance kernel $\text{Cov}(\overline{r}(\tilde{s}_i), \overline{r}(\tilde{s}_j)) = \mathcal{K}_r(\tilde{s}_i, \tilde{s}_j)$ for all state-action pairs $\tilde{s}_i, \tilde{s}_j$. For instance, $\mathcal{K}_r$ could be the squared exponential kernel:

$$\mathcal{K}_r(\tilde{s}_i, \tilde{s}_j) = \sigma_f^2 \exp\left(-\frac{1}{2}\left(\frac{||f(\tilde{s}_i) - f(\tilde{s}_j)||}{l}\right)^2\right) + \sigma_n^2 \delta_{ij}, \tag{A.4}$$

where σ_f^2 is the signal variance, l is the kernel lengthscale, σ_n^2 is the noise variance, δ_{ij} is the Kronecker delta function, and $f : \{1, \ldots, S\} \times \{1, \ldots, A\} \longrightarrow \mathbb{R}^m$ maps each state-action pair to an m-dimensional representation that encodes proximity between the state-action pairs. For instance, in the Mountain Car problem, each state-action pair could be represented by a position and velocity (encoding the state) and a one-dimensional action, so that $m = 3$. Thus,

$$\overline{r}(\tilde{s}_i) \sim \mathcal{GP}(\mu_r(\tilde{s}_i), \mathcal{K}_r(\tilde{s}_i, \tilde{s}_j)).$$

Define $\boldsymbol{\mu}_r \in \mathbb{R}^d$ such that the i^{th} element is $[\boldsymbol{\mu}_r]_i = \mu_r(\tilde{s}_i)$, the prior mean of state-action $\tilde{s}_i$'s utility. Let $K_r \in \mathbb{R}^{d \times d}$ be the covariance matrix over state-action utilities, such that $[K_r]_{ij} = \mathcal{K}_r(\tilde{s}_i, \tilde{s}_j)$. Therefore, the reward vector $\overline{\boldsymbol{r}}$ is also a Gaussian process:

$$\overline{\boldsymbol{r}} \sim \mathcal{GP}(\boldsymbol{\mu}_r, K_r).$$

The Trajectory Utility Gaussian Process

By assumption, the trajectory utilities $\boldsymbol{R} \in \mathbb{R}^{2N}$ are sums of the latent state-action utilities via the following relationship between $\boldsymbol{R}$ and $\overline{\boldsymbol{r}}$:

$$R(z_i) := R_i = \sum_{j=1}^{d} z_{ij} \overline{r}(\tilde{s}_j) + \varepsilon_i,$$

where ε_i are i.i.d. noise variables distributed according to $\mathcal{N}(0, \sigma_\varepsilon^2)$. Note that $R(z_i)$ is a Gaussian process over $z_i \in \mathbb{R}^d$ because $\{\overline{r}(\tilde{s}_j), \forall j\}$ are jointly normally distributed by definition of a Gaussian process, and any linear combination of jointly Gaussian variables has a univariate normal distribution. Next, the expectation and covariance of $\boldsymbol{R}$ is calculated. The expectation of the i^{th} element $R_i = R(z_i)$ can be expressed as:

$$\mathbb{E}[R_i] = \mathbb{E}\left[\sum_{j=1}^{d} z_{ij} \overline{r}(\tilde{s}_j) + \varepsilon_i\right] = \sum_{j=1}^{d} z_{ij} \mathbb{E}[\overline{r}(\tilde{s}_j)] = \sum_{j=1}^{d} z_{ij} \mu_r(\tilde{s}_j).$$

The expectation over $\boldsymbol{R}$ can thus be written as $\mathbb{E}[\boldsymbol{R}(Z)] = Z\boldsymbol{\mu}_r$. Next, the covariance matrix of $\boldsymbol{R}$ is computed. The ij^{th} element of this matrix is the covariance of $R(z_i)$ and $R(z_j)$:

$$\begin{aligned}
&\text{Cov}(R(z_i), R(z_j)) = \mathbb{E}[R(z_i)R(z_j)] - \mathbb{E}[R(z_i)]\mathbb{E}[R(z_j)] \\
&\quad = \mathbb{E}\left[\left(\sum_{k=1}^{d} z_{ik} \overline{r}(\tilde{s}_k) + \varepsilon_i\right)\left(\sum_{m=1}^{d} z_{jm} \overline{r}(\tilde{s}_m) + \varepsilon_j\right)\right] - \left(\sum_{k=1}^{d} z_{ik} \mu_r(\tilde{s}_k)\right)\left(\sum_{m=1}^{d} z_{jm} \mu_r(\tilde{s}_m)\right) \\
&\quad = \sum_{k=1}^{d}\sum_{m=1}^{d} z_{ik} z_{jm} \mathbb{E}[\overline{r}(\tilde{s}_k) r(\tilde{s}_m)] + \mathbb{E}[\varepsilon_i \varepsilon_j] - \sum_{k=1}^{d}\sum_{m=1}^{d} z_{ik} z_{jm} \mu_r(\tilde{s}_k)\mu_r(\tilde{s}_m) \\
&\quad = \sum_{k=1}^{d}\sum_{m=1}^{d} \left\{z_{ik} z_{jm} [\text{Cov}(\overline{r}(\tilde{s}_k), \overline{r}(\tilde{s}_m)) + \mu_r(\tilde{s}_k)\mu_r(\tilde{s}_m)] - z_{ik} z_{jm} \mu_r(\tilde{s}_k)\mu_r(\tilde{s}_m) + \sigma_\varepsilon^2 \mathbb{I}_{[i=j]}\right\} \\
&\quad = \sum_{k=1}^{d}\sum_{m=1}^{d} z_{ik} z_{jm} \text{Cov}(\overline{r}(\tilde{s}_k), \overline{r}(\tilde{s}_m)) + \sigma_\varepsilon^2 \mathbb{I}_{[i=j]} \\
&\quad = \sum_{k=1}^{d}\sum_{m=1}^{d} z_{ik} z_{jm} \mathcal{K}_r(\tilde{s}_k, \tilde{s}_m) + \sigma_\varepsilon^2 \mathbb{I}_{[i=j]} = z_i^T K_r z_j + \sigma_\varepsilon^2 \mathbb{I}_{[i=j]}.
\end{aligned}$$

One can then write the covariance matrix of $\boldsymbol{R}$ as K_R, where:

$$[K_R]_{ij} := \text{Cov}(R(z_i), R(z_j)) = z_i^T K_r z_j + \sigma_\varepsilon^2 \mathbb{I}_{[i=j]}.$$

From here, it can be seen that $K_R = ZK_rZ^T + \sigma_\varepsilon^2 I$:

$$ZK_rZ^T = \begin{bmatrix} z_1^T \\ z_2^T \\ \vdots \\ z_{2N}^T \end{bmatrix} K_r \begin{bmatrix} z_1 & z_2 & \dots & z_{2N} \end{bmatrix} = \begin{bmatrix} z_1^T K_r z_1 & \dots & z_1^T K_r z_{2N} \\ \vdots & \ddots & \vdots \\ z_{2N}^T K_r z_1 & \dots & z_{2N}^T K_r z_{2N} \end{bmatrix} = K_R - \sigma_\varepsilon^2 I.$$

Covariance between State-Action and Trajectory Utilities

This subsection considers the covariance between $\overline{\boldsymbol{r}}$ and $\boldsymbol{R}$, denoted $K_{r,R}$:

$$[K_{r,R}]_{ij} = \mathrm{Cov}([\overline{\boldsymbol{r}}]_i, [\boldsymbol{R}]_j) = \mathrm{Cov}(\overline{r}(\tilde{s}_i), R(z_j)).$$

This covariance matrix can be expressed in terms of Z, K_r, and $\boldsymbol{\mu}_r$:

$$\begin{aligned}
[K_{r,R}]_{ij} &= \mathrm{Cov}(\overline{r}(\tilde{s}_i), R(z_j)) = \mathrm{Cov}\left(\overline{r}(\tilde{s}_i), \sum_{k=1}^{d} z_{jk}\overline{r}(\tilde{s}_k) + \varepsilon_j\right) \\
&= \mathbb{E}\left[\overline{r}(\tilde{s}_i)\sum_{k=1}^{d} z_{jk}\overline{r}(\tilde{s}_k) + \varepsilon_j\overline{r}(\tilde{s}_i)\right] - \mathbb{E}[\overline{r}(\tilde{s}_i)]\mathbb{E}\left[\sum_{k=1}^{d} z_{jk}\overline{r}(\tilde{s}_k) + \varepsilon_j\right] \\
&= \sum_{k=1}^{d} z_{jk}\mathbb{E}[\overline{r}(\tilde{s}_i)\overline{r}(\tilde{s}_k)] - [\mu_r(\tilde{s}_i)][z_j^T\boldsymbol{\mu}_r] \\
&= \sum_{k=1}^{d} z_{jk}\{\mathrm{Cov}(\overline{r}(\tilde{s}_i), \overline{r}(\tilde{s}_k)) + \mathbb{E}[\overline{r}(\tilde{s}_i)]\mathbb{E}[\overline{r}(\tilde{s}_k)]\} - \mu_r(\tilde{s}_i)z_j^T\boldsymbol{\mu}_r \\
&= \sum_{k=1}^{d} z_{jk}[\mathcal{K}_r(\tilde{s}_i, \tilde{s}_k) + \mu_r(\tilde{s}_i)\mu_r(\tilde{s}_k)] - \mu_r(\tilde{s}_i)z_j^T\boldsymbol{\mu}_r \\
&= \sum_{k=1}^{d} z_{jk}\mathcal{K}_r(\tilde{s}_i, \tilde{s}_k) + \mu_r(\tilde{s}_i)z_j^T\boldsymbol{\mu}_r - \mu_r(\tilde{s}_i)z_j^T\boldsymbol{\mu}_r = \sum_{k=1}^{d} z_{jk}\mathcal{K}_r(\tilde{s}_i, \tilde{s}_k) = z_j^T[K_r]_{i,:}^T,
\end{aligned}$$

where $[K_r]_{i,:}^T$ is the column vector obtained by transposing the i^{th} row of K_r. It is evident that $K_{r,R} = K_rZ^T$.

Posterior Inference over State-Action Utilities

Merging the previous three subsections' results, one obtains the following joint probability density between $\overline{\boldsymbol{r}}$ and $\boldsymbol{R}$:

$$\begin{bmatrix} \overline{\boldsymbol{r}} \\ \boldsymbol{R} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu}_r \\ Z\boldsymbol{\mu}_r \end{bmatrix}, \begin{bmatrix} K_r & K_rZ^T \\ ZK_r^T & ZK_rZ^T + \sigma_\varepsilon^2 I \end{bmatrix}\right).$$

This relationship expresses all components of the joint Gaussian density in terms of Z, K_r, and $\boldsymbol{\mu}_r$, or in other words, in terms of the observed state-action visitation

counts (i.e., Z) and the Gaussian process prior on $\overline{\boldsymbol{r}}$. The standard approach for obtaining a conditional distribution from a joint Gaussian distribution yields $\overline{\boldsymbol{r}}|\boldsymbol{R} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$, where the expressions for $\boldsymbol{\mu}$ and Σ are given by Eq.s (A.2) and (A.3) above. By substituting $\boldsymbol{y}'$ for $\boldsymbol{R}$, the conditional posterior density of $\overline{\boldsymbol{r}}$ can be expressed in terms of Z, $\boldsymbol{y}'$, K_r, and $\boldsymbol{\mu_r}$, that is, in terms of observed data and the Gaussian process prior parameters.

A.3 Gaussian Process Preference Model

Lastly, this section shows how to extend the preference-based Gaussian process model defined in Chu and Ghahramani (2005b) from the dueling bandit setting to the preference-based RL setting to perform credit assignment. As in the previous section, the notation and terminology is adapted to the preference-based RL setting; however, the same mathematics apply to the generalized linear dueling bandit problem.

Similarly to Section A.2, this approach places a Gaussian prior over possible utility vectors $\overline{\boldsymbol{r}}$; in contrast, however, this method explicitly models the likelihood of the observed preferences given $\overline{\boldsymbol{r}}$, and thus it is a more theoretically-justified approach for handling preference data.

After N preferences have been elicited, the algorithm has collected a preference feedback dataset $\mathcal{D} = \{(\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}, y_i) \,|\, i = 1, ..., N\}$, where $y_i = \frac{1}{2}$ indicates that $\boldsymbol{x}_{i2} \succ \boldsymbol{x}_{i1}$, that is, trajectory τ_{i2} is preferred to τ_{i1} in preference i. As before, each state-action pair $\tilde{s}_j, j \in \{1, \ldots, d\}$, is assumed to have a latent, underlying utility $\overline{r}(\tilde{s}_j)$. In vector form, these are written: $\overline{\boldsymbol{r}} = [\overline{r}(\tilde{s}_1), \overline{r}(\tilde{s}_2), ..., \overline{r}(\tilde{s}_d)]^T$. A Gaussian process prior is defined over the utilities $\overline{\boldsymbol{r}}$:

$$p(\overline{\boldsymbol{r}}) = \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}\overline{\boldsymbol{r}}^T \Sigma^{-1} \overline{\boldsymbol{r}}\right), \tag{A.5}$$

where $\Sigma \in \mathbb{R}^{d \times d}$ and $[\Sigma]_{ij} = \mathcal{K}(\overline{r}(\tilde{s}_i), \overline{r}(\tilde{s}_j))$ for some kernel function $\mathcal{K}$, such as the squared exponential kernel defined in Eq. (A.4). Next, the likelihood of the i^{th} preference given utilities $\overline{\boldsymbol{r}}$ is assumed to take the following form, where $y'_i = 2y_i \in \{-1, 1\}$:

$$P(\tau_{i2} \succ \tau_{i1} \mid \overline{\boldsymbol{r}}) = P(\boldsymbol{x}_{i2} \succ \boldsymbol{x}_{i1} \mid \overline{\boldsymbol{r}}) = g\left(y'_i * \frac{u(\tau_{i2}) - u(\tau_{i1})}{c}\right),$$

where $g(\cdot)$ is a monotonically-increasing link function with a range within $[0, 1]$, and $c > 0$ is a model hyperparameter controlling the degree of preference noise. The total return $u(\tau_{i1})$ of trajectory τ_{i1} can be written in terms of the corresponding

state-action visit counts, $\boldsymbol{x}_{i1}$: $u(\tau_{i1}) = \overline{\boldsymbol{r}}^T \boldsymbol{x}_{i1}$. Thus, the full likelihood expression is:

$$P(\mathcal{D} \mid \overline{\boldsymbol{r}}) = \prod_{i=1}^{N} g(z_i), \tag{A.6}$$

$$z_i := \frac{y_i'(u(\tau_{i2}) - u(\tau_{i1}))}{c} = \frac{y_i' \overline{\boldsymbol{r}}^T (\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1})}{c} = \frac{y_i' \overline{\boldsymbol{r}}^T \boldsymbol{x}_i}{c}.$$

Given the preference dataset $\mathcal{D}$, one can model the posterior probability of $\overline{\boldsymbol{r}}$:

$$p(\overline{\boldsymbol{r}} \mid \mathcal{D}) \propto P(\mathcal{D} \mid \overline{\boldsymbol{r}}) p(\overline{\boldsymbol{r}}),$$

where the expressions for the prior $p(\overline{\boldsymbol{r}})$ and likelihood $P(\mathcal{D} \mid \overline{\boldsymbol{r}})$ are given by Eq.s (A.5) and (A.6), respectively. This posterior can be estimated by the Laplace approximation, from which samples $\tilde{\boldsymbol{r}}$ of the utilities $\overline{\boldsymbol{r}}$ can easily be drawn:

$$\tilde{\boldsymbol{r}} \sim \mathcal{N}(\hat{\boldsymbol{r}}_{\text{MAP}}, \alpha \Sigma_{\text{MAP}}), \text{ where:} \tag{A.7}$$

$$\hat{\boldsymbol{r}}_{\text{MAP}} = \text{argmin}_{\boldsymbol{r}} S(\boldsymbol{r}), \tag{A.8}$$

$$\Sigma_{\text{MAP}} = \left(\nabla^2_{\boldsymbol{r}} S(\boldsymbol{r}) |_{\hat{\boldsymbol{r}}_{\text{MAP}}} \right)^{-1}, \tag{A.9}$$

and $S(\boldsymbol{r}) := \frac{1}{2} \boldsymbol{r}^T \Sigma^{-1} \boldsymbol{r} - \sum_{i=1}^{N} \log g(z_i)$ is the negative log posterior, neglecting constant terms with respect to $\boldsymbol{r}$; lastly, $\alpha > 0$ is a tunable hyperparameter that influences the balance between exploration and exploitation. In order for the Laplace approximation to be valid, $S(\boldsymbol{r})$ must be convex in $\boldsymbol{r}$: this guarantees that the optimization problem in Eq. (A.8) is convex and that the covariance matrix defined by Eq. (A.9) is positive definite, and therefore a valid Gaussian covariance matrix. Convexity of $S(\boldsymbol{r})$ can be established by demonstrating that its Hessian matrix is positive definite. It can be shown that for any $\boldsymbol{r}$, $\nabla^2_{\boldsymbol{r}} S(\boldsymbol{r}) = \Sigma^{-1} + \Lambda$, where:

$$[\Lambda]_{mn} := \frac{1}{c^2} \sum_{i=1}^{N} [\boldsymbol{x}_i]_m [\boldsymbol{x}_i]_n \left[-\frac{g''(z_i)}{g(z_i)} + \left(\frac{g'(z_i)}{g(z_i)} \right)^2 \right], \tag{A.10}$$

for $\boldsymbol{x}_i = \boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}$. Because the prior covariance Σ is positive definite, to show that $\nabla^2_{\boldsymbol{r}} S(\boldsymbol{r})$ is positive definite, it suffices to show that Λ is positive semidefinite. From Eq. (A.10), one can see that:

$$\Lambda = \frac{1}{c^2} \sum_{i=1}^{N} \left[-\frac{g''(z_i)}{g(z_i)} + \left(\frac{g'(z_i)}{g(z_i)} \right)^2 \right] \boldsymbol{x}_i \boldsymbol{x}_i^T.$$

Clearly, $\boldsymbol{x}_i \boldsymbol{x}_i^T$ is positive semidefinite, and thus, the following statement is a sufficient condition for convexity of $S(\boldsymbol{r})$:

$$\left[-\frac{g''(z)}{g(z)} + \left(\frac{g'(z)}{g(z)} \right) \right] \geq 0 \text{ for all } z \in \mathbb{R}.$$

In particular, this condition is satisfied for the Gaussian link function, $g_{\text{Gaussian}}(\cdot) = \Phi(\cdot)$, where Φ is the standard Gaussian CDF, as well as for the sigmoidal link function, $g_{\log}(x) := \sigma(x) = \frac{1}{1+\exp(-x)}$. In this work, the experiments utilize the sigmoidal link function.

Bayesian Logistic Regression

Notably, the Bayesian logistic regression inference model discussed in Section 4.3 is a special case of the Gaussian process preference model, in which $c = 1$, g is the sigmoidal link function, and the prior covariance matrix is diagonal, i.e. $\Sigma = \lambda I$; for instance, the latter condition occurs with the squared exponential kernel defined in Eq. (A.4) when its lengthscale l is set to zero. In this thesis, a number of the experiments with the Gaussian process preference model fall under the special case of Bayesian logistic regression, and therefore, this model is briefly reviewed here.

In Bayesian logistic regression, the Gaussian prior over possible reward vectors $\boldsymbol{r} \in \mathbb{R}^d$ is: $\boldsymbol{r} \sim \mathcal{N}(\mathbf{0}, \lambda I)$, where $\lambda > 0$. Setting the i^{th} preference label y_i equal to $\frac{1}{2}$ if $\boldsymbol{x}_{i2} \succ \boldsymbol{x}_{i1}$, while $y_i = -\frac{1}{2}$ if $\boldsymbol{x}_{i1} \succ \boldsymbol{x}_{i2}$, the logistic regression likelihood is:

$$p(\mathcal{D}|\boldsymbol{r}) = \prod_{i=1}^{N} p(y_i|\boldsymbol{r}, \boldsymbol{x}_i) = \prod_{i=1}^{N} \frac{1}{1 + \exp(-2y_i \boldsymbol{x}_i^T \boldsymbol{r})}.$$

The experiments approximate the posterior, $p(\boldsymbol{r} \mid \mathcal{D}) \propto p(\mathcal{D} \mid \boldsymbol{r}) p(\boldsymbol{r})$, as Gaussian via the Laplace approximation:

$$\begin{aligned} &p(\boldsymbol{r} \mid \mathcal{D}) \approx \mathcal{N}(\hat{\boldsymbol{r}}^{\text{MAP}}, \alpha \Sigma^{\text{MAP}}), \text{ where:} \\ &\quad \hat{\boldsymbol{r}}^{\text{MAP}} = \operatorname*{argmin}_{\boldsymbol{r}} f(\boldsymbol{r}), \quad f(\boldsymbol{r}) := -\log p(\mathcal{D}, \boldsymbol{r}) = -\log p(\boldsymbol{r}) - \log p(\mathcal{D}|\boldsymbol{r}), \\ &\quad \Sigma^{\text{MAP}} = \left(\nabla^2_{\boldsymbol{r}} f(\boldsymbol{r}) \Big|_{\hat{\boldsymbol{r}}} \right)^{-1}, \end{aligned} \tag{A.11}$$

where the optimization in Eq. (A.11) is convex, and $\alpha > 0$ is a tunable hyperparameter that influences the balance between exploration and exploitation. Note that multiplying the covariance by a well-tuned α is more practical than using the $\beta_i(\delta)$ parameters considered in the asymptotic consistency analysis (Section 4.4), as the latter results in overly-conservative covariance matrices in practice.

Appendix B

PROOFS OF ASYMPTOTIC CONSISTENCY FOR DUELING POSTERIOR SAMPLING

This appendix proves the asymptotic consistency results stated in Section 4.4. The details are organized into three sections, which prove:

1. In the preference-based RL setting, samples from the model posterior over transition dynamics parameters converge in distribution to the true transition probabilities.
2. In both the preference-based generalized linear bandit and RL settings, samples from the utility posterior converge in distribution to the true utilities.
3. DPS's selected policies converge in distribution to the optimal policy in the preference-based RL setting. DPS's selected actions converge to the optimal action in the generalized linear bandit setting with a finite action space $\mathcal{A}$.

Please refer to Section 4.2 to review relevant notation, e.g. for the posterior samples drawn in each iteration. In addition, the following notation is used for the value function and for policies given by value iteration:

Definition 5 (Value function given transition dynamics, rewards, and a policy)**.** *Define $V(\boldsymbol{p}, \boldsymbol{r}, \pi)$ as the value function over a length-h episode—i.e., the expected total reward in the episode—under transition dynamics $\boldsymbol{p} \in \mathbb{R}^{S^2A}$, rewards $\boldsymbol{r} \in \mathbb{R}^{SA}$, and policy π:*

$$V(\boldsymbol{p}, \boldsymbol{r}, \pi) = \sum_{s \in \mathcal{S}} p_0(s) \mathbb{E}\left[\sum_{t=1}^{h} \overline{r}(s_t, \pi(s_t, t)) \,\middle|\, s_1 = s, \overline{\boldsymbol{p}} = \boldsymbol{p}, \overline{\boldsymbol{r}} = \boldsymbol{r} \right].$$

Definition 6 (Optimal deterministic policy given transition dynamics and rewards)**.** *Define $\pi_{vi}(\boldsymbol{p}, \boldsymbol{r}) := \operatorname{argmax}_\pi V(\boldsymbol{p}, \boldsymbol{r}, \pi)$ as the optimal deterministic policy given transition dynamics $\boldsymbol{p} \in \mathbb{R}^{S^2A}$ and rewards $\boldsymbol{r} \in \mathbb{R}^{SA}$ (breaking ties randomly if multiple deterministic policies achieve the maximum). Note that $\pi_{vi}(\boldsymbol{p}, \boldsymbol{r})$ can be found via finite-horizon value iteration: defining $V_{\pi,t}(s)$ as in Eq.* (3.6)*, set $V_{\pi,h+1}(s) := 0$ for each $s \in \mathcal{S}$ and use the Bellman equation to calculate $V_{\pi,t}(s)$*

successively for $t \in \{h, h-1, \ldots, 1\}$ *given* $\boldsymbol{p}$ *and* $\boldsymbol{r}$*:*

$$\pi(s,t) = \text{argmax}_{a \in \mathcal{A}} \left[\overline{r}(s,a) + \sum_{s' \in \mathcal{S}} P(s_{t+1} = s' \mid s_t = s, a_t = a) V_{\pi,t+1}(s') \right],$$

$$V_{\pi,t}(s) = \sum_{a \in \mathcal{A}} \mathbb{I}_{[\pi(s,t)=a]} \left[\overline{r}(s,a) + \sum_{s' \in \mathcal{S}} P(s_{t+1} = s' \mid s_t = s, a_t = a) V_{\pi,t+1}(s') \right].$$

As value iteration results in only deterministic policies, of which there are finitely-many (more precisely, there are A^{Sh}*), the maximum argument* $\pi_{vi}(\boldsymbol{p}, \boldsymbol{r}) := \text{argmax}_{\pi}$ $V(\boldsymbol{p}, \boldsymbol{r}, \pi)$ *is taken over a finite policy class.*

Recall that $M_n := \lambda I + \sum_{i=1}^{n-1} \boldsymbol{x}_i \boldsymbol{x}_i^T$ (see Eq. (4.2)). For the linear link function, the posterior sampling distribution's covariance is given by $\Sigma^{(i)} = \beta_n(\delta)^2 M_n^{-1}$. For the logistic link function, the posterior covariance is given by $\Sigma^{(i)} = \beta_n(\delta)^2 M_n'$, where $M_n' = \lambda I + \sum_{i=1}^{n-1} \tilde{g}(2y_i \boldsymbol{x}_i^T \hat{\boldsymbol{r}}_n) \boldsymbol{x}_i \boldsymbol{x}_i^T$, $\tilde{g}(x) := \left(\frac{g_{\log}'(x)}{g_{\log}(x)} \right)^2 - \frac{g_{\log}''(x)}{g_{\log}(x)}$ comes from the Laplace approximation, and $g_{\log}$ is the sigmoid function.

As in Section 4.4, it is notationally convenient to define a matrix $\tilde{M}_n \in \mathbb{R}^{d \times d}$ such that:

$$\begin{cases} \tilde{M}_n = M_n = \lambda I + \sum_{i=1}^{n-1} \boldsymbol{x}_i \boldsymbol{x}_i^T & \text{for the linear link function, and} \\ \tilde{M}_n = M_n' = \lambda I + \sum_{i=1}^{n-1} \tilde{g}(2y_i \boldsymbol{x}_i^T \hat{\boldsymbol{r}}_n) \boldsymbol{x}_i \boldsymbol{x}_i^T & \text{for the logistic link function.} \end{cases} \tag{B.1}$$

Then, under either link function, the posterior sampling distribution has covariance $\Sigma^{(n)} = \beta_n(\delta)^2 \tilde{M}_n^{-1}$.

Finally, notation is defined for the eigenvectors and eigenvalues of the matrices M_i and $\tilde{M}_i$:

Definition 7 (Eigenvalue notation)**.** *Let* $\nu_j^{(i)}$ *refer to the* j^{th}*-largest eigenvalue of* M_i*, and* $\boldsymbol{u}_j^{(i)}$ *denote its corresponding eigenvector. Similarly, let* $\lambda_j^{(i)}$ *refer to the* j^{th}*-largest eigenvalue of* $\tilde{M}_i$*, and* $\boldsymbol{v}_j^{(i)}$ *denote its corresponding eigenvector. Note that* M_i^{-1} *also has eigenvectors* $\boldsymbol{u}_j^{(i)}$*, with corresponding eigenvalues* $\frac{1}{\nu_j^{(i)}}$*. Because* M_i *is positive definite, the eigenvectors* $\{\boldsymbol{u}_j^{(i)}\}$ *form an orthonormal basis, and* $\nu_j^{(i)} > 0$ *for all* i, j*. The equivalent statements also hold for* $\tilde{M}_i$*, which is also positive definite because* $\tilde{g}(x) > 0$ *for all possible inputs.*

B.1 Facts about Convergence in Distribution

Before proceeding with the asymptotic consistency proofs, two facts about convergence in distribution are reviewed; these will be applied later.

Recall that for a random variable X and a sequence of random variables (X_n), $n \in \mathbb{N}$, $X_n \xrightarrow{D} X$ denotes that X_n converges to X in distribution, while $X_n \xrightarrow{P} X$ denotes that X_n converges to X in probability.

Fact 8 (Billingsley, 1968)**.** *For random variables $\boldsymbol{x}, \boldsymbol{x}_n, \in \mathbb{R}^d$, where $n \in \mathbb{N}$, and any continuous function $g : \mathbb{R}^d \longrightarrow \mathbb{R}$, if $\boldsymbol{x}_n \xrightarrow{D} \boldsymbol{x}$, then $g(\boldsymbol{x}_n) \xrightarrow{D} g(\boldsymbol{x})$.*

Fact 9 (Billingsley, 1968)**.** *For random variables $\boldsymbol{x}_n \in \mathbb{R}^d$, $n \in \mathbb{N}$, and constant vector $\boldsymbol{c} \in \mathbb{R}^d$, $\boldsymbol{x}_n \xrightarrow{D} \boldsymbol{c}$ is equivalent to $\boldsymbol{x}_n \xrightarrow{P} \boldsymbol{c}$. Convergence in probability means that for any $\varepsilon > 0$, $P(||\boldsymbol{x}_n - \boldsymbol{c}||_2 \geq \varepsilon) \longrightarrow 0$ as $n \longrightarrow \infty$.*

B.2 Asymptotic Consistency of the Transition Dynamics in DPS in the Preference-Based RL Setting

DPS models state transition dynamics independently for each state-action pair. For a given state-action pair, a Dirichlet model estimates the probability of transitioning to each possible subsequent state. The prior and posterior distributions are both Dirichlet; because the Dirichlet and multinomial distributions are conjugate (see Section 2.1), each state-action pair's posterior can be updated easily using the observed transitions from that state-action. Each time that DPS draws a sample from the dynamics distribution, values are sampled for all S^2A transition parameters, $\{P(s_{t+1} = s' \mid s_t = s, a_t = a) \mid s, s' \in \mathcal{S}, a \in \mathcal{A}\}$.

To demonstrate convergence in distribution of the sampled transition dynamics parameters, first, Lemma 3 shows that if every state-action pair is visited infinitely often, the desired result holds. Then, Lemma 6 completes the argument by showing that DPS indeed visits each state-action pair infinitely often.

Lemma 3. *If every state-action pair is visited infinitely often, then the sampled transition dynamics parameters converge in distribution to their true values: $\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{p}}_{i2} \xrightarrow{D} \overline{\boldsymbol{p}}$ as $i \longrightarrow \infty$, where $\xrightarrow{D}$ denotes convergence in distribution.*

Proof. Denote the $d = SA$ state-action pairs as $\tilde{s}_1, \ldots, \tilde{s}_d$. At a particular DPS episode, let n_j be the number of visits to $\tilde{s}_j$ and n_{jk} be the number of observed transitions from $\tilde{s}_j$ to the k^{th} subsequent state. For the j^{th} state-action pair at iteration i, let $\overline{\boldsymbol{p}}^{(j)}$, $\tilde{\boldsymbol{p}}^{(j)}$, $\hat{\boldsymbol{p}}^{(j)}$, $\hat{\boldsymbol{p}}'^{(j)} \in \mathbb{R}^S$ be the true, sampled, posterior mean, and maximum likelihood

dynamics parameters, respectively (hiding the dependency on the DPS episode $i1$ or $i2$ for the latter three quantities); thus, $[\overline{\boldsymbol{p}}^{(j)}]_k$ denotes the true probability of transitioning from state-action pair $\tilde{s}_j$ to the k^{th} state, and analogously for the k^{th} elements of $\tilde{\boldsymbol{p}}^{(j)}$, $\hat{\boldsymbol{p}}^{(j)}$, and $\hat{\boldsymbol{p}}'^{(j)}$. Then, from the Dirichlet model,

$$[\hat{\boldsymbol{p}}^{(j)}]_k = \frac{n_{jk} + \alpha_{jk,0}}{n_j + \sum_{m=1}^{S} \alpha_{jm,0}},$$

where the prior for $\overline{\boldsymbol{p}}^{(j)}$ is $\frac{1}{\sum_{m=1}^{S} \alpha_{jm,0}} [\alpha_{j1,0}, \ldots, \alpha_{jS,0}]^T$ for user-defined hyperparameters $\alpha_{jk,0} > 0$. Meanwhile, the maximum likelihood is given by $[\hat{\boldsymbol{p}}'^{(j)}]_k = \frac{n_{jk}}{\max(n_j,1)}$ (this is equivalent to $[\hat{\boldsymbol{p}}^{(j)}]_k$, except with the prior parameters set to zero). Consider the sampled dynamics at state-action pair $\tilde{s}_j$. For any $\varepsilon > 0$,

$$\begin{aligned}
P\big(||\tilde{\boldsymbol{p}}^{(j)} - \overline{\boldsymbol{p}}^{(j)}||_1 \geq \varepsilon\big) &= P\big(||\tilde{\boldsymbol{p}}^{(j)} - \hat{\boldsymbol{p}}^{(j)} + \hat{\boldsymbol{p}}^{(j)} - \hat{\boldsymbol{p}}'^{(j)} + \hat{\boldsymbol{p}}'^{(j)} - \overline{\boldsymbol{p}}^{(j)}||_1 \geq \varepsilon\big) \\
&\overset{(a)}{\leq} P\big(||\tilde{\boldsymbol{p}}^{(j)} - \hat{\boldsymbol{p}}^{(j)}||_1 + ||\hat{\boldsymbol{p}}^{(j)} - \hat{\boldsymbol{p}}'^{(j)}||_1 + ||\hat{\boldsymbol{p}}'^{(j)} - \overline{\boldsymbol{p}}^{(j)}||_1 \geq \varepsilon\big) \\
&\leq P\Big(||\tilde{\boldsymbol{p}}^{(j)} - \hat{\boldsymbol{p}}^{(j)}||_1 \geq \frac{\varepsilon}{3} \bigcup ||\hat{\boldsymbol{p}}^{(j)} - \hat{\boldsymbol{p}}'^{(j)}||_1 \geq \frac{\varepsilon}{3} \bigcup ||\hat{\boldsymbol{p}}'^{(j)} - \overline{\boldsymbol{p}}^{(j)}||_1 \geq \frac{\varepsilon}{3}\Big) \\
&\overset{(b)}{\leq} P\Big(||\tilde{\boldsymbol{p}}^{(j)} - \hat{\boldsymbol{p}}^{(j)}||_1 \geq \frac{\varepsilon}{3}\Big) + P\Big(||\hat{\boldsymbol{p}}^{(j)} - \hat{\boldsymbol{p}}'^{(j)}||_1 \geq \frac{\varepsilon}{3}\Big) + P\Big(||\hat{\boldsymbol{p}}'^{(j)} - \overline{\boldsymbol{p}}^{(j)}||_1 \geq \frac{\varepsilon}{3}\Big),
\end{aligned} \tag{B.2}$$

where (a) holds due to the triangle inequality and (b) follows from the union bound. This proof will upper-bound each term in Eq. (B.2) in terms of n_j and show that it decays as $n_j \longrightarrow \infty$, that is, as $\tilde{s}_j$ is visited infinitely often. For the first term, this bound is achieved via Chebyshev's inequality:

$$\begin{aligned}
P\Big(||\tilde{\boldsymbol{p}}^{(j)} - \hat{\boldsymbol{p}}^{(j)}||_1 \geq \frac{\varepsilon}{3}\Big) &\leq P\left(\bigcup_{k=1}^{S} \left\{\left|[\tilde{\boldsymbol{p}}^{(j)}]_k - [\hat{\boldsymbol{p}}^{(j)}]_k\right| \geq \frac{\varepsilon}{3S}\right\}\right) \\
&\overset{(a)}{\leq} \sum_{k=1}^{S} P\left(\left|[\tilde{\boldsymbol{p}}^{(j)}]_k - [\hat{\boldsymbol{p}}^{(j)}]_k\right| \geq \frac{\varepsilon}{3S}\right) \overset{(b)}{\leq} \sum_{k=1}^{S} \frac{9S^2}{\varepsilon^2} \text{Var}\left[[\tilde{\boldsymbol{p}}^{(j)}]_k\right],
\end{aligned}$$

where (a) follows from the union bound and (b) is an application of Chebyshev's inequality. For a Dirichlet variable X with parameters $(\alpha_1, \ldots, \alpha_S)$, $\alpha_k > 0$ for each k, the variance of the k^{th} component X_k is given by:

$$\text{Var}[X_k] = \frac{\tilde{\alpha}_k(1 - \tilde{\alpha}_k)}{1 + \sum_{m=1}^{S} \alpha_m} \leq \frac{1}{2} * \frac{1}{1 + \sum_{m=1}^{S} \alpha_m},$$

where $\tilde{\alpha}_k := \frac{\alpha_k}{\sum_{m=1}^{S} \alpha_m}$. In the DPS algorithm, $\tilde{\boldsymbol{p}}^{(j)}$ is drawn from a Dirichlet distribu-

tion with parameters $(\alpha_{j1}, \ldots, \alpha_{jS}) = (\alpha_{j1,0} + n_{j1}, \ldots, \alpha_{jS,0} + n_{jS})$, so that,

$$\begin{aligned}\text{Var}\left[[\tilde{\boldsymbol{p}}^{(j)}]_k\right] &\leq \frac{1}{2} * \frac{1}{1 + \sum_{m=1}^{S} \alpha_{jm}} = \frac{1}{2} * \frac{1}{1 + \sum_{m=1}^{S} (\alpha_{jm,0} + n_{jm})} \\ &\leq \frac{1}{2} * \frac{1}{1 + \sum_{m=1}^{S} n_{jm}} = \frac{1}{2(1 + n_j)}.\end{aligned}$$

Thus,

$$P\left(||\tilde{\boldsymbol{p}}^{(j)} - \hat{\boldsymbol{p}}^{(j)}||_1 \geq \frac{\varepsilon}{3}\right) \leq \sum_{k=1}^{S} \frac{9S^2}{\varepsilon^2} \frac{1}{2(1 + n_j)} = \frac{9S^3}{2\varepsilon^2(1 + n_j)}.$$

Considering the second term in Eq. (B.2),

$$\begin{aligned}P\left(||\hat{\boldsymbol{p}}^{(j)} - \hat{\boldsymbol{p}}'^{(j)}||_1 \geq \frac{\varepsilon}{3}\right) &\leq P\left(\bigcup_{k=1}^{S} \left\{\left|[\hat{\boldsymbol{p}}^{(j)} - \hat{\boldsymbol{p}}'^{(j)}]_k\right| \geq \frac{\varepsilon}{3S}\right\}\right) \\ &\overset{(a)}{\leq} \sum_{k=1}^{S} P\left(\left|[\hat{\boldsymbol{p}}^{(j)}]_k - [\hat{\boldsymbol{p}}'^{(j)}]_k\right| \geq \frac{\varepsilon}{3S}\right) \overset{(b)}{\leq} \sum_{k=1}^{S} P\left(\frac{\alpha_{jk,0} + \sum_{m=1}^{S} \alpha_{jm,0}}{n_j + \sum_{m=1}^{S} \alpha_{jm,0}} \geq \frac{\varepsilon}{3S}\right),\end{aligned}$$

where (a) holds via the union bound and (b) follows for $n_j \geq 1$ because when $n_j \geq 1$:

$$\begin{aligned}\left|[\hat{\boldsymbol{p}}^{(j)}]_k - [\hat{\boldsymbol{p}}'^{(j)}]_k\right| &= \left|\frac{n_{jk} + \alpha_{jk,0}}{n_j + \sum_{m=1}^{S} \alpha_{jm,0}} - \frac{n_{jk}}{n_j}\right| = \left|\frac{\alpha_{jk,0}}{n_j + \sum_{m=1}^{S} \alpha_{jm,0}} - \frac{n_{jk} \sum_{m=1}^{S} \alpha_{jm,0}}{n_j(n_j + \sum_{m=1}^{S} \alpha_{jm,0})}\right| \\ &\leq \frac{\alpha_{jk,0}}{n_j + \sum_{m=1}^{S} \alpha_{jm,0}} + \frac{n_{jk}}{n_j} \frac{\sum_{m=1}^{S} \alpha_{jm,0}}{n_j + \sum_{m=1}^{S} \alpha_{jm,0}} \leq \frac{\alpha_{jk,0} + \sum_{m=1}^{S} \alpha_{jm,0}}{n_j + \sum_{m=1}^{S} \alpha_{jm,0}}.\end{aligned}$$

For the third term in Eq. (B.2), one can apply the following concentration inequality for Dirichlet variables (see Appendix C.1 in Jaksch, Ortner, and Auer, 2010):

$$P(||\hat{\boldsymbol{p}}'^{(j)} - \overline{\boldsymbol{p}}^{(j)}||_1 \geq \varepsilon) \leq (2^S - 2) \exp\left(\frac{-n_j \varepsilon^2}{2}\right).$$

Therefore:

$$P\left(||\hat{\boldsymbol{p}}'^{(j)} - \overline{\boldsymbol{p}}^{(j)}||_1 \geq \frac{\varepsilon}{3}\right) \leq (2^S - 2) \exp\left(\frac{-n_j \varepsilon^2}{18}\right).$$

Thus, to upper-bound the right-hand side of Eq. (B.2), for any $\varepsilon > 0$:

$$P\left(||\tilde{\boldsymbol{p}}^{(j)} - \overline{\boldsymbol{p}}^{(j)}||_1 \geq \varepsilon\right) \leq \frac{9S^3}{2\varepsilon^2(n_j + 1)} + \sum_{k=1}^{S} P\left(\frac{\alpha_{jk,0} + \sum_{m=1}^{S} \alpha_{jm,0}}{n_j + \sum_{m=1}^{S} \alpha_{jm,0}} \geq \frac{\varepsilon}{3S}\right) + (2^S - 2) \exp\left(\frac{-n_j \varepsilon^2}{18}\right).$$

On the right hand side, the first and third terms clearly decay as $n_j \longrightarrow \infty$. The middle term is identically zero for n_j large enough, since the $\alpha_{jk,0}$ values are user-defined constants. Given this inequality, it is clear that for any $\varepsilon > 0$, as $n_j \longrightarrow \infty$,

$P\left(||\tilde{\boldsymbol{p}}^{(j)} - \overline{\boldsymbol{p}}^{(j)}||_1 \geq \varepsilon\right) \longrightarrow 0$. If every state-action pair is visited infinitely often, then $n_j \longrightarrow \infty$ for each j, and therefore, $\tilde{\boldsymbol{p}}^{(j)}$ converges in probability to $\overline{\boldsymbol{p}}^{(j)}$: $\tilde{\boldsymbol{p}}^{(j)} \xrightarrow{P} \overline{\boldsymbol{p}}^{(j)}$. Convergence in probability implies convergence in distribution, the desired result. □

To continue proving that DPS's model of the transition dynamics converges, this analysis uses that the magnitude of the utility estimator $||\hat{\boldsymbol{r}}_n||_2$, the mean of the utility posterior sampling distribution, is uniformly upper-bounded; in other words, there exists $b < \infty$ such that $||\hat{\boldsymbol{r}}_n||_2 \leq b$.

Lemma 4. *When preferences are given by a linear or logistic link function, across all $n \geq 1$, there exists some $b < \infty$ such that estimated reward at DPS trial n is bounded by b: $||\hat{\boldsymbol{r}}_n||_2 \leq b$.*

Proof. Firstly, if the link function is logistic, the desired result holds automatically by the definition of $\hat{\boldsymbol{r}}_n$ given in Eq. (4.10): the quantity is projected onto the compact set $\Theta \subset \mathbb{R}^d$ of all possible values of $\overline{\boldsymbol{r}}$, and a compact set on $\mathbb{R}^d$ must be bounded.

Secondly, the result is proven in the case of a linear link function. In this case, recall that the MAP reward estimate $\hat{\boldsymbol{r}}_n$ is the solution to a ridge regression problem:

$$\hat{\boldsymbol{r}}_n = \arg\inf_{\boldsymbol{r}} \left\{ \sum_{i=1}^{n-1} (\boldsymbol{x}_i^T \boldsymbol{r} - y_i)^2 + \lambda ||\boldsymbol{r}||_2^2 \right\} = \arg\inf_{\boldsymbol{r}} \left\{ \sum_{i=1}^{n-1} \left[(\boldsymbol{x}_i^T \boldsymbol{r} - y_i)^2 + \frac{1}{n-1} \lambda ||\boldsymbol{r}||_2^2 \right] \right\}. \tag{B.3}$$

The desired result is proven by contradiction. Assuming that there exists no upper bound b, the proof will identify a subsequence $(\hat{\boldsymbol{r}}_{n_i})$ of MAP estimates whose lengths increase unboundedly, but whose directions converge. Then, it will show that such vectors fail to minimize the objective in Eq. (B.3), achieving a contradiction.

Firstly, the vectors $\boldsymbol{x}_i = \boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}$ have bounded magnitude: in the bandit case, $\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2} \in \mathcal{A}$, and the action space $\mathcal{A}$ is compact, while in the RL setting, $||\boldsymbol{x}_{ij}||_1 = h$ for $j \in \{1, 2\}$. The binary labels y_i are also bounded, as they take values in $\left\{-\frac{1}{2}, \frac{1}{2}\right\}$. Note that for $\boldsymbol{r} = \boldsymbol{0}$, $(\boldsymbol{x}_i^T \boldsymbol{r} - y_i)^2 + \frac{1}{n-1}\lambda ||\boldsymbol{r}||_2^2 = \frac{1}{4}$. The desired statement is proven by contradiction: assume that there is no $b < \infty$ such that $||\hat{\boldsymbol{r}}_n||_2 \leq b$ for all n. Then, the sequence $\hat{\boldsymbol{r}}_1, \hat{\boldsymbol{r}}_2, \ldots$ must have a subsequence indexed by (n_i) such that $\lim_{i \longrightarrow \infty} ||\hat{\boldsymbol{r}}_{n_i}||_2 = \infty$. Consider the sequence of unit vectors $\frac{\hat{\boldsymbol{r}}_{n_i}}{||\hat{\boldsymbol{r}}_{n_i}||_2}$. This sequence lies within the compact set of unit vectors in $\mathbb{R}^d$, so it must have a convergent subsequence; we index this subsequence of the sequence (n_i) by (n_{i_j}). Then, the

sequence $(\hat{\boldsymbol{r}}_{i_j})$ is such that $\lim_{j\longrightarrow\infty} ||\hat{\boldsymbol{r}}_{i_j}||_2 = \infty$ and $\lim_{j\longrightarrow\infty} \frac{\hat{\boldsymbol{r}}_{i_j}}{||\hat{\boldsymbol{r}}_{i_j}||_2} = \hat{\boldsymbol{r}}_{unit}$, where $\hat{\boldsymbol{r}}_{unit} \in \mathbb{R}^d$ is a fixed unit vector.

For any $\boldsymbol{x}_i$ such that $|\boldsymbol{x}_i^T \hat{\boldsymbol{r}}_{unit}| \neq 0$, $\lim_{n_{i_j}\longrightarrow\infty} (\boldsymbol{x}_i^T \hat{\boldsymbol{r}}_{n_{i_j}} - y_i)^2 = \infty$, and thus, the corresponding terms in Eq. (B.3) approach infinity. However, a lower value of the optimization objective in Eq. (B.3) can be realized by replacing $\hat{\boldsymbol{r}}_{n_{i_j}}$ with the assignment $\boldsymbol{r} = \boldsymbol{0}$. Meanwhile, for any $\boldsymbol{x}_i$ such that $|\boldsymbol{x}_i^T \hat{\boldsymbol{r}}| = 0$, replacing $\hat{\boldsymbol{r}}_{n_{i_j}}$ with $\boldsymbol{r} = \boldsymbol{0}$ would also decrease the value of the optimization objective in Eq. (B.3). Therefore, for large j, $\boldsymbol{r} = \boldsymbol{0}$ results in a smaller objective function value than $\hat{\boldsymbol{r}}_{n_{i_j}}$. This is a contradiction, proving that the elements of the sequence $\hat{\boldsymbol{r}}_{n_{i_j}}$ cannot have arbitrarily large magnitudes. Thus, the elements of the original sequence $\hat{\boldsymbol{r}}_i$ also cannot become arbitrarily large, and $||\hat{\boldsymbol{r}}_i|| \leq b$ for some $b < \infty$. □

The next intermediate result relates the matrix $\tilde{M}_n$, defined in Eq. (B.1), and the matrix $M_n = \lambda I + \sum_{i=1}^{n-1} \boldsymbol{x}_i \boldsymbol{x}_i^T$.

Lemma 5. *On iteration n of DPS, the posterior covariance matrix for the rewards is* $\Sigma^{(n)} = \beta_n(\delta)^2 \tilde{M}_n$*; if the link function g is linear, then* $\tilde{M}_n = M_n$*, while if g is logistic, then* $\tilde{M}_n = \lambda I + \sum_{i=1}^{n-1} \tilde{g}(2y_i \boldsymbol{x}_i^T \hat{\boldsymbol{r}}_n) \boldsymbol{x}_i \boldsymbol{x}_i^T$*. In both cases, there exist two constants* $m_{\min}, m_{\max}$ *such that* $0 < m_{\min} \leq m_{\max} < \infty$ *and* $m_{\min} M_n \preceq \tilde{M}_n \preceq m_{\max} M_n$*.*

Proof. Firstly, if g is linear, then $\tilde{M}_n = M_n$, so the desired result clearly holds with $m_{\min} = m_{\max} = 1$.

If g is logistic, the desired statement is equivalent to:

$$m_{\min} \left(\lambda I + \sum_{i=1}^{n-1} \boldsymbol{x}_i \boldsymbol{x}_i^T \right) \preceq \lambda I + \sum_{i=1}^{n-1} \tilde{g}(2y_i \boldsymbol{x}_i^T \hat{\boldsymbol{r}}_n) \boldsymbol{x}_i \boldsymbol{x}_i^T \preceq m_{\max} \left(\lambda I + \sum_{i=1}^{n-1} \boldsymbol{x}_i \boldsymbol{x}_i^T \right).$$

By definition of $\tilde{g}$, $\tilde{g}(x) \in (0, \infty)$ for all $x \in \mathbb{R}$. Moreover, the domain of $\tilde{g}$ has bounded magnitude, since all possible inputs to $\tilde{g}$ are of the form $2y_i \boldsymbol{x}_i^T \hat{\boldsymbol{r}}_n$, in which $|y_i| = \frac{1}{2}$, $\boldsymbol{x}_i$ belongs to a compact set, and $||\hat{\boldsymbol{r}}_n|| \leq b$ by Lemma 4. Therefore, all possible inputs to $\tilde{g}$ belong to a compact set. A continuous function over a compact set always attains its maximum and minimum values; therefore, there exist values $\tilde{g}_{\min}, \tilde{g}_{\max}$ such that $0 < \tilde{g}_{\min} \leq \tilde{g}(x) \leq \tilde{g}_{\max} < \infty$ for all possible inputs x to $\tilde{g}(x)$.

Therefore,

$$\lambda I + \sum_{i=1}^{n-1} \tilde{g}(2y_i \boldsymbol{x}_i^T \hat{\boldsymbol{r}}_n) \boldsymbol{x}_i \boldsymbol{x}_i^T \succeq \lambda I + \sum_{i=1}^{n-1} \tilde{g}_{\min} \boldsymbol{x}_i \boldsymbol{x}_i^T \succeq \min\{\tilde{g}_{\min}, 1\} \left[\lambda I + \sum_{i=1}^{n-1} \boldsymbol{x}_i \boldsymbol{x}_i^T \right], \text{ and}$$

$$\lambda I + \sum_{i=1}^{n-1} \tilde{g}(2y_i \boldsymbol{x}_i^T \hat{\boldsymbol{r}}_n) \boldsymbol{x}_i \boldsymbol{x}_i^T \preceq \lambda I + \sum_{i=1}^{n-1} \tilde{g}_{\max} \boldsymbol{x}_i \boldsymbol{x}_i^T \preceq \max\{\tilde{g}_{\max}, 1\} \left[\lambda I + \sum_{i=1}^{n-1} \boldsymbol{x}_i \boldsymbol{x}_i^T \right],$$

which proves the desired result for $m_{\min} = \min\{\tilde{g}_{\min}, 1\}$ and $m_{\max} = \max\{\tilde{g}_{\max}, 1\}$. □

To finish proving convergence of the transition dynamics Bayesian model, Lemma 6 demonstrates that every state-action pair is visited infinitely often.

Lemma 6. *Under DPS with preference-based RL, assume that the dynamics are modeled via a Dirichlet model and that the utilities are modeled either via the linear or logistic link functions, with posterior sampling distributions given in Eq.* (4.13)*. Then, every state-action pair is visited infinitely often.*

This consistency result also holds when removing the $\beta_n(\delta)$ and $\beta'_n(\delta)$ factors from the distributions in Eq. (4.13)*.*

Proof. The proof proceeds by assuming that there exists a state-action pair that is visited only finitely-many times. This assumption will lead to a contradiction[1]: once this state-action pair is no longer visited, the posterior sampling distribution for the utilities $\overline{\boldsymbol{r}}$ is no longer updated with respect to it. Then, DPS is guaranteed to eventually sample a high enough reward for this state-action that the resultant policy will prioritize visiting it.

First, note that DPS is guaranteed to reach at least one state-action pair infinitely often: given the problem's finite state and action spaces, at least one state-action pair must be visited infinitely often during DPS execution. If all state-actions are *not* visited infinitely often, there must exist a state-action pair (s, a) such that s is visited infinitely often, while (s, a) is not. Otherwise, if all actions are selected infinitely often in all infinitely-visited states, the finitely-visited states are unreachable (in which case these states are irrelevant to the learning process and regret minimization,

[1]Note that in finite-horizon MDPs, the concept of visiting a state finitely-many times is not the same as that of a transient state in an infinite Markov chain, because: 1) due to a finite horizon, the state is resampled from the initial state distribution $p_0(s)$ every h timesteps, and 2) the policy—which determines which state-action pairs can be reached in an episode—is also resampled every h timesteps.

and can be ignored). Without loss of generality, this state-action pair (s, a) is labeled as $\tilde{s}_1$. To reach a contradiction, it suffices to show that $\tilde{s}_1$ is visited infinitely often.

Let $\boldsymbol{r}_1$ be the utility vector with a reward of 1 in state-action pair $\tilde{s}_1$ and rewards of zero elsewhere. From Definition 6, $\pi_{vi}(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1)$ is the policy that maximizes the expected number of visits to $\tilde{s}_1$ under dynamics $\tilde{\boldsymbol{p}}$ and utility vector $\boldsymbol{r}_1$:

$$\pi_{vi}(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1) = \text{argmax}_{\pi} V(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1, \pi),$$

where $V(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1, \pi)$ is the expected total reward of a length-h trajectory under $\tilde{\boldsymbol{p}}, \boldsymbol{r}_1$, and π, or equivalently (by definition of $\boldsymbol{r}_1$), the expected number of visits to state-action $\tilde{s}_1$.

Next, it will be shown that there exists a $\rho > 0$ such that $P(\pi = \pi_{vi}(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1)) > \rho$ for all possible values of $\tilde{\boldsymbol{p}}$. That is, for any sampled parameters $\tilde{\boldsymbol{p}}$, the probability of selecting policy $\pi_{vi}(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1)$ is uniformly lower-bounded, implying that DPS must eventually select $\pi_{vi}(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1)$.

Let $\tilde{r}_j$ be the sampled utility (also referred to as reward) associated with state-action pair $\tilde{s}_j$ in a particular DPS episode, for each state-action $j \in \{1, \ldots, d\}$, with $d = SA$. The proof will show that conditioned on $\tilde{\boldsymbol{p}}$, there exists $v > 0$ such that if $\tilde{r}_1$ exceeds $\max\{v\tilde{r}_2, v\tilde{r}_3, \ldots, v\tilde{r}_d\}$, then value iteration returns the policy $\pi_{vi}(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1)$, which is the policy maximizing the expected amount of time spent in $\tilde{s}_1$. This can be seen by setting $v := \frac{h}{\rho_1}$, where h is the time horizon and ρ_1 is the expected number of visits to $\tilde{s}_1$ under $\pi_{vi}(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1)$. Under this definition of v, the event $\{\tilde{r}_1 \geq \max\{v\tilde{r}_2, v\tilde{r}_3, \ldots, v\tilde{r}_d\}\}$ is equivalent to $\{\tilde{r}_1\rho_1 \geq h \max\{\tilde{r}_2, \tilde{r}_3, \ldots, \tilde{r}_d\}\}$; the latter inequality implies that given $\tilde{\boldsymbol{p}}$ and $\tilde{\boldsymbol{r}}$, the expected reward accumulated solely in state-action $\tilde{s}_1$ exceeds the reward gained by repeatedly (during all h time-steps) visiting the state-action pair in the set $\{\tilde{s}_2, \ldots, \tilde{s}_d\}$ having the highest sampled reward. Clearly, in this situation, value iteration results in the policy $\pi_{vi}(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1)$.

Next, it is shown that $v = \frac{h}{\rho_1}$ is continuous in the sampled dynamics $\tilde{\boldsymbol{p}}$ by showing that ρ_1 is continuous in $\tilde{\boldsymbol{p}}$. Recall that ρ_1 is defined as expected number of visits to $\tilde{s}_1$ under $\pi_{vi}(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1)$. This is equivalent to the expected reward for following $\pi_{vi}(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1)$ under dynamics $\tilde{\boldsymbol{p}}$ and rewards $\boldsymbol{r}_1$:

$$\rho_1 = V(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1, \pi_{vi}(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1)) = \max_{\pi} V(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1, \pi). \tag{B.4}$$

The value of any policy π is continuous in the transition dynamics parameters, so $V(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1, \pi)$ is continuous in $\tilde{\boldsymbol{p}}$. The maximum in Eq. (B.4) is taken over the finite set

of deterministic policies; because a maximum over a finite number of continuous functions is also continuous, ρ_1 is continuous in $\tilde{\boldsymbol{p}}$.

Next, recall that a continuous function on a compact set achieves its maximum and minimum values on that set. The set of all possible dynamics parameters $\tilde{\boldsymbol{p}}$ is such that for each state-action pair j, $\sum_{k=1}^{S} p_{jk} = 1$ and $p_{jk} \geq 0 \,\forall\, k$; the set of all possible vectors $\tilde{\boldsymbol{p}}$ is clearly closed and bounded, and hence compact. Therefore, v achieves its maximum and minimum values on this set, and for any $\tilde{\boldsymbol{p}}$, $v \in [v_{\min}, v_{\max}]$, where $v_{\min} > 0$ (v is nonnegative by definition, and $v = 0$ is impossible, as it would imply that $\tilde{s}_1$ is unreachable).

Then, $P(\pi = \pi_{vi}(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1))$ can be expressed in terms of v and the parameters of the reward posterior. Firstly,

$$P(\pi = \pi_{vi}(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1)) \geq P(\tilde{r}_1 > \max\{v\tilde{r}_2, v\tilde{r}_3, \ldots, v\tilde{r}_d\}) \geq \prod_{j=2}^{d} P(\tilde{r}_1 > v\tilde{r}_j).$$

In the n^{th} DPS iteration, the sampled rewards are drawn from a jointly Gaussian posterior: $\tilde{\boldsymbol{r}} \sim \mathcal{N}(\boldsymbol{\mu}^{(n)}, \Sigma^{(n)})$ for some $\boldsymbol{\mu}^{(n)}$ and $\Sigma^{(n)}$, where $[\boldsymbol{\mu}^{(n)}]_j = \mu_j^{(n)}$ and $[\Sigma^{(n)}]_{jk} = \Sigma_{jk}^{(n)}$. Then, $(\tilde{r}_1 - v\tilde{r}_j) \sim \mathcal{N}(\mu_1^{(n)} - v\mu_j^{(n)}, \Sigma_{11}^{(n)} + v^2\Sigma_{jj}^{(n)} - 2v\Sigma_{1j}^{(n)})$, so that:

$$\begin{aligned} P(\pi_{n1} = \pi_{vi}(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1)) &\geq \prod_{j=2}^{d} \left[1 - \Phi\left(\frac{-\mu_1^{(n)} + v\mu_j^{(n)}}{\sqrt{\Sigma_{11}^{(n)} + v^2\Sigma_{jj}^{(n)} - 2v\Sigma_{1j}^{(n)}}} \right) \right] \\ &= \prod_{j=2}^{d} \Phi\left(\frac{\mu_1^{(n)} - v\mu_j^{(n)}}{\sqrt{\Sigma_{11}^{(n)} + v^2\Sigma_{jj}^{(n)} - 2v\Sigma_{1j}^{(n)}}} \right), \end{aligned} \tag{B.5}$$

where Φ is the standard Gaussian cumulative distribution function. For the right-hand expression in Eq. (B.5) to have a lower bound greater than zero, the argument of $\Phi(\cdot)$ must be lower-bounded. It suffices to upper-bound the numerator's magnitude and to lower-bound the denominator above zero for each product factor j and over all iterations n.

The numerator can be upper-bounded using Lemma 4. Since $\boldsymbol{\mu}^{(n)}$ equals $\hat{\boldsymbol{r}}_n$ at iteration n, $||\boldsymbol{\mu}^{(n)}||_2 \leq b$; therefore, $|\mu_1^{(n)}|, |\mu_j^{(n)}| \leq b$. Because $0 < v \leq v_{\max}$, $|\mu_1 - v\mu_j| \leq |\mu_1^{(n)}| + v|\mu_j^{(n)}| \leq (1 + v_{\max})b$.

To lower-bound the denominator, first note that it is equal to $\sqrt{\boldsymbol{w}_j^T \Sigma^{(n)} \boldsymbol{w}_j}$, in which $\boldsymbol{w}_j \in \mathbb{R}^d$ is defined as a vector with 1 in the first position, $-v$ in the j^{th} position for

some $j \in \{2, \ldots, d\}$, and zero elsewhere:

$$\boldsymbol{w}_j := [1, 0, \ldots, 0, -v, 0, \ldots, 0]^T. \tag{B.6}$$

Equivalently, it must be shown that $\boldsymbol{w}_j^T \Sigma^{(n)} \boldsymbol{w}_j$ is lower-bounded above zero. By Lemma 5, it holds that $\Sigma^{(n)} \geq \frac{\beta_n(\delta)^2}{m_{\max}} M_n^{-1}$, implying that $\boldsymbol{w}_j^T \Sigma^{(n)} \boldsymbol{w}_j \geq \frac{\beta_n(\delta)^2}{m_{\max}} \boldsymbol{w}_j^T M_n^{-1} \boldsymbol{w}_j$. Because $m_{\max}$ is a constant and $\beta_n(\delta)$, defined in Eq. (4.4), is non-decreasing in n, it suffices to prove that $\boldsymbol{w}_j^T M_n^{-1} \boldsymbol{w}_j$ is lower-bounded above zero. (Thus, the result holds regardless of the presence of $\beta_n(\delta)$ in the utility sampling distribution.)

Recall from Definition 7 that the eigenvectors of M_n^{-1} are $\boldsymbol{u}_1^{(n)}, \ldots, \boldsymbol{u}_d^{(n)}$, with corresponding eigenvalues $\left(\nu_1^{(n)}\right)^{-1}, \ldots, \left(\nu_d^{(n)}\right)^{-1}$. The vector $\boldsymbol{w}_j$ can be written in terms of the orthonormal basis formed by the eigenvectors $\{\boldsymbol{u}_k^{(n)}\}$:

$$\boldsymbol{w}_j = \sum_{k=1}^{d} \alpha_k^{(n)} \boldsymbol{u}_k^{(n)}, \tag{B.7}$$

for some coefficients $\alpha_k^{(n)} \in \mathbb{R}$. Using Eq. (B.7), the quantity to be lower-bounded can now be written as:

$$\begin{aligned} \boldsymbol{w}_j^T M_n^{-1} \boldsymbol{w}_j &= \left(\sum_{k=1}^{d} \alpha_k^{(n)} \boldsymbol{u}_k^{(n)T} \right) \left(\sum_{l=1}^{d} \frac{1}{\nu_l^{(n)}} \boldsymbol{u}_l^{(n)} \boldsymbol{u}_l^{(n)T} \right) \left(\sum_{m=1}^{d} \alpha_m^{(n)} \boldsymbol{u}_m^{(n)} \right) \\ &\overset{(a)}{=} \sum_{k=1}^{d} \left(\alpha_k^{(n)}\right)^2 \frac{1}{\nu_k^{(n)}} \overset{(b)}{\geq} \left(\alpha_{k_0}^{(n)}\right)^2 \frac{1}{\nu_{k_0}^{(n)}}, \end{aligned} \tag{B.8}$$

where equality (a) follows by orthonormality of the eigenvector basis, and (b) holds for any $k_0 \in \{1, \ldots, d\}$ due to positivity of the eigenvalues $(\nu_k)^{-1}$. Therefore, to show that the denominator is bounded away from zero, it suffices to show that for every n, there exists some k_0 such that $\left(\alpha_{k_0}^{(n)}\right)^2 \left(\nu_{k_0}^{(n)}\right)^{-1}$ is bounded away from zero.

To prove the previous statement, note that by definition of M_n, the eigenvalues $(\nu_k^{(n)})^{-1}$ are non-increasing in n. Below, the proof will show that for any eigenvalue $(\nu_k^{(n)})^{-1}$ such that $\lim_{n \longrightarrow \infty} (\nu_k^{(n)})^{-1} = 0$, the first element of its corresponding eigenvector, $\left[\boldsymbol{u}_k^{(n)}\right]_1$, also converges to zero. Since the first element of $\boldsymbol{w}_j$ equals 1, Eq. (B.6) implies that there must exist some k_0 such that $\left[\boldsymbol{u}_{k_0}^{(n)}\right]_1 \not\longrightarrow 0$ and $\alpha_{k_0}^{(n)}$ is bounded away from 0. If these implications did not hold, then $\boldsymbol{w}_j$ would not have a value of 1 in its first element, contradicting its definition. These observations imply that for every n, there must be some k_0 such that as $n \longrightarrow \infty$, $(\nu_{k_0}^{(n)})^{-1} \not\longrightarrow 0$ and $\alpha_{k_0}^{(n)}$ is bounded away from zero.

Let X_n denote the observation matrix after $n-1$ observations: $X_n := \begin{bmatrix} \boldsymbol{x}_1 & \dots & \boldsymbol{x}_{n-1} \end{bmatrix}^T$. Then, $M_n^{-1} = (X_n^T X_n + \lambda I)^{-1}$. The matrices M_n^{-1} and $X_n^T X_n$ have the same eigenvectors. Meanwhile, for each eigenvalue $(\nu_i^{(n)})^{-1}$ of M_n^{-1}, $X_n^T X_n$ has an eigenvalue $\xi_i^{(n)} := \nu_i^{(n)} - \lambda \geq 0$ corresponding to the same eigenvector. We aim to characterize the eigenvectors of M_n^{-1} whose eigenvalues approach zero. Since these eigenvectors are identical to those of $X_n^T X_n$ whose eigenvalues approach infinity, the latter can be considered instead.

Without loss of generality, assume that all finitely-visited state-action pairs (including $\tilde{s}_1$) occur in the first $m < n-1$ iterations, and index these finitely-visited state-action pairs from 1 to $r \geq 1$, so that the finitely-visited state-actions are: $\{\tilde{s}_1, \tilde{s}_2, \cdots, \tilde{s}_r\}$. Let $X_{1:m} \in \mathbb{R}^{m \times d}$ denote the matrix containing the first m rows of X_n, while $X_{m+1:n} \in \mathbb{R}^{n-m \times d}$ denotes the remaining rows of X_n. With this notation,

$$X_n^T X_n = \sum_{i=1}^{n-1} \boldsymbol{x}_i \boldsymbol{x}_i^T = X_{1:m}^T X_{1:m} + X_{m+1:n}^T X_{m+1:n}.$$

Because the first r state-action pairs, $\{\tilde{s}_1, \tilde{s}_2, \cdots, \tilde{s}_r\}$, are unvisited after iteration m, the first r elements of $\boldsymbol{x}_i$ are zero for all $i > m$. Therefore, $X_{m+1:n}^T X_{m+1:n}$ can be written in the following block matrix form:

$$X_{m+1:n}^T X_{m+1:n} = \begin{bmatrix} O_{r \times r} & O_{r \times (d-r)} \\ O_{(d-r) \times r} & A_n \end{bmatrix},$$

where $O_{a \times b}$ denotes the all-zero matrix with dimensions $a \times b$. The matrix A_n includes elements that are unbounded as $n \longrightarrow \infty$. In particular, the diagonal elements of A_n approach infinity as $n \longrightarrow \infty$. The matrix $X_n^T X_n$ can be written in the following block matrix form:

$$\begin{aligned} X_n^T X_n &= X_{1:m}^T X_{1:m} + X_{m+1:n}^T X_{m+1:n} \\ &= \begin{bmatrix} [X_{1:m}^T X_{1:m}]_{(1:r,1:r)} & [X_{1:m}^T X_{1:m}]_{(1:r,r+1:d)} \\ [X_{1:m}^T X_{1:m}]_{(r+1:d,1:r)} & [X_{1:m}^T X_{1:m}]_{(r+1:d,r+1:d)} + A_n \end{bmatrix} := \begin{bmatrix} B & C \\ C^T & D_n \end{bmatrix}, \end{aligned}$$

where $M_{(a:b,c:d)}$ denotes the submatrix of M obtained by extracting rows a through b and columns c through d. Because matrices B and C only depend upon $X_{1:m}$, they are fixed as n increases, while matrix D_n contains values that grow towards infinity with increasing n. In particular, all elements along D_n's diagonal are unbounded. Intuitively, in the limit, B and C are close to zero compared to D_n, and $X_n^T X_n$ (when normalized) increasingly resembles a matrix in which only the bottom-right block is nonzero. This intuitive notion is formalized next.

Consider an eigenpair $(\boldsymbol{u}_i^{(n)}, \xi_i^{(n)})$ of $X_n^T X_n$ such that $\lim_{n\longrightarrow\infty} \xi_i^{(n)} = \infty$. The following argument shows that the first element of $\boldsymbol{u}_i^{(n)}$ must approach 0. Letting $\boldsymbol{u}_i^{(n)} = \begin{bmatrix} \boldsymbol{z}_i^{(n)T} & \boldsymbol{q}_i^{(n)T} \end{bmatrix}^T$, where $\boldsymbol{z}_i^{(n)} \in \mathbb{R}^m$ and $\boldsymbol{q}_i^{(n)} \in \mathbb{R}^{n-1-m}$:

$$(X_n^T X_n)\boldsymbol{u}_i^{(n)} = X_n^T X_n \begin{bmatrix} \boldsymbol{z}_i^{(n)} \\ \boldsymbol{q}_i^{(n)} \end{bmatrix} = \begin{bmatrix} B & C \\ C^T & D_n \end{bmatrix} \begin{bmatrix} \boldsymbol{z}_i^{(n)} \\ \boldsymbol{q}_i^{(n)} \end{bmatrix} = \begin{bmatrix} B\boldsymbol{z}_i^{(n)} + C\boldsymbol{q}_i^{(n)} \\ C^T\boldsymbol{z}_i^{(n)} + D_n\boldsymbol{q}_i^{(n)} \end{bmatrix} = \xi_i^{(n)} \begin{bmatrix} \boldsymbol{z}_i^{(n)} \\ \boldsymbol{q}_i^{(n)} \end{bmatrix}.$$

Dividing both sides by $\xi_i^{(n)}$,

$$\frac{1}{\xi_i^{(n)}} X_n^T X_n \begin{bmatrix} \boldsymbol{z}_i^{(n)} \\ \boldsymbol{q}_i^{(n)} \end{bmatrix} = \begin{bmatrix} \frac{1}{\xi_i^{(n)}} \left(B\boldsymbol{z}_i^{(n)} + C\boldsymbol{q}_i^{(n)} \right) \\ \frac{1}{\xi_i^{(n)}} \left(C^T\boldsymbol{z}_i^{(n)} + D_n\boldsymbol{q}_i^{(n)} \right) \end{bmatrix} = \begin{bmatrix} \boldsymbol{z}_i^{(n)} \\ \boldsymbol{q}_i^{(n)} \end{bmatrix}.$$

In the upper matrix block: $\lim_{n\longrightarrow\infty} \xi_i^{(n)} = \infty$, B and C are fixed as n increases, and $\boldsymbol{z}_i^{(n)}$ and $\boldsymbol{q}_i^{(n)}$ have upper-bounded elements because $\boldsymbol{u}_i^{(n)}$ is a unit vector. Thus, $\lim_{n\longrightarrow\infty} \boldsymbol{z}_i^{(n)} = \lim_{n\longrightarrow\infty} \frac{1}{\xi_i^{(n)}} \left(B\boldsymbol{z}_i^{(n)} + C\boldsymbol{q}_i^{(n)} \right) = \boldsymbol{0}$. In particular, the first element of $\boldsymbol{z}_i^{(n)}$ converges to zero, implying that the same is true of $\boldsymbol{u}_i^{(n)}$.

As justified above, this result implies that for each iteration n, there exists an index $k_0 \in \{1, \ldots, d\}$ such that the right-hand side of Eq. (B.8) has a lower bound above zero. This completes the proof that the denominator in Eq. (B.5) does not decay to zero. As a result, there exists some $\rho > 0$ such that $P(\pi = \pi_{vi}(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1)) \geq \rho > 0$.

In consequence, DPS is guaranteed to infinitely often sample pairs $(\tilde{\boldsymbol{p}}, \pi)$ such that $\pi = \pi_{vi}(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1)$. As a result, DPS infinitely often samples policies that prioritize reaching $\tilde{s}_1$ as quickly as possible. Such a policy always takes action a in state s. Furthermore, because s is visited infinitely often, either a) $p_0(s) > 0$ or b) the infinitely-visited state-action pairs include a path with a nonzero probability of reaching s. In case a), since the initial state distribution is fixed, the MDP will infinitely often begin in state s under the policy $\pi = \pi_{vi}(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1)$, so $\tilde{s}_1$ will be visited infinitely often. In case b), due to Lemma 3, the transition dynamics parameters for state-actions along the path to s converge to their true values (intuitively, the algorithm knows how to reach s). In episodes with the policy $\pi = \pi_{vi}(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1)$, DPS is thus guaranteed to reach $\tilde{s}_1$ infinitely often. Since DPS selects $\pi_{vi}(\tilde{\boldsymbol{p}}, \boldsymbol{r}_1)$ infinitely often, it must reach $\tilde{s}_1$ infinitely often. This presents a contradiction, proving that every state-action pair must be visited infinitely often. □

The direct combination of Lemmas 3 and 6 prove asymptotic consistency of the transition dynamics model:

Proposition 3. *Assume that DPS is executed in the preference-based RL setting, with transition dynamics modeled via a Dirichlet model, utilities modeled via either the linear or logistic link function, and utility posterior sampling distributions given in Eq.* (4.13)*. Then, the sampled transition dynamics* $\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{p}}_{i2}$ *converge in distribution to the true dynamics,* $\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{p}}_{i2} \xrightarrow{D} \overline{\boldsymbol{p}}$*. This consistency result also holds when removing the* $\beta_n(\delta)$ *factors from the distributions in Eq.* (4.13)*.*

B.3 Asymptotic Consistency of the Utilities in DPS

This section shows that in both the bandit and RL cases, reward samples drawn from the utility model posterior converge to the true utility values, $\overline{\boldsymbol{r}}$. The analysis first considers the bandit setting, and then utilizes asymptotic consistency of the MDP transition dynamics to extend the results to the preference-based RL problem.

Asymptotic Consistency of the Utilities for Generalized Linear Dueling Bandits

The first step, encapsulated in the lemma below, leverages results from Abbasi-Yadkori, Pál, and Szepesvári (2011) and Filippi et al. (2010) to derive a sufficient condition for asymptotic consistency of the utilities.

Lemma 7. *If* $\frac{\beta_i(\delta)^2}{\lambda_d^{(i)}} \xrightarrow{D} 0$ *as* $i \longrightarrow \infty$*, where* $\lambda_d^{(i)}$ *is the minimum eigenvalue of* $\tilde{M}_i$*, then* $\tilde{\boldsymbol{r}}_{i1}, \tilde{\boldsymbol{r}}_{i2} \xrightarrow{D} \overline{\boldsymbol{r}}$ *with probability* $1-\delta$*.*

Proof. From Propositions 1 and 2, with probability at least $1-\delta$, the utility estimator $\hat{\boldsymbol{r}}_i$ belongs to a confidence ellipsoid centered at $\overline{\boldsymbol{r}}$: $||\hat{\boldsymbol{r}}_i - \overline{\boldsymbol{r}}||_{M_i} \leq \beta_i(\delta)$. The proof will show that under this high-probability event, $\tilde{\boldsymbol{r}}_{i1}, \tilde{\boldsymbol{r}}_{i2} \xrightarrow{D} \overline{\boldsymbol{r}}$.

Firstly, by Lemma 5, $M_i \geq \frac{1}{m_{\max}}\tilde{M}_i$; thus, $||\hat{\boldsymbol{r}}_i - \overline{\boldsymbol{r}}||_{\frac{1}{m_{\max}}\tilde{M}_i} = \frac{1}{m_{\max}}||\hat{\boldsymbol{r}}_i - \overline{\boldsymbol{r}}||_{\tilde{M}_i} \leq ||\hat{\boldsymbol{r}}_i - \overline{\boldsymbol{r}}||_{M_i}$. Meanwhile, the posterior sampling distribution is given by,

$$\tilde{\boldsymbol{r}}_{i1}, \tilde{\boldsymbol{r}}_{i2} \sim \mathcal{N}\left(\hat{\boldsymbol{r}}_i, \beta_i(\delta)^2 \tilde{M}_i^{-1}\right). \tag{B.9}$$

Letting $z_i \sim \mathcal{N}(\mathbf{0}, I)$ be independent for each i, the sample $\tilde{\boldsymbol{r}}_{i1}$ (and similarly, $\tilde{\boldsymbol{r}}_{i2}$) can equivalently be expressed as:

$$\tilde{\boldsymbol{r}}_{i1} = \hat{\boldsymbol{r}}_i + \beta_i(\delta)\tilde{M}_i^{-\frac{1}{2}} z_i, \tag{B.10}$$

since the random variable in Eq. (B.10) has the same distribution as that in Eq. (B.9). The quantity $||\tilde{\boldsymbol{r}}_{i1} - \hat{\boldsymbol{r}}_i||_{\tilde{M}_i}$ can be rewritten as:

$$||\tilde{\boldsymbol{r}}_{i1} - \hat{\boldsymbol{r}}_i||_{\tilde{M}_i} = \left\|\beta_i(\delta)\tilde{M}_i^{-\frac{1}{2}} z_i\right\|_{\tilde{M}_i} = \beta_i(\delta)\sqrt{z_i^T \tilde{M}_i^{-\frac{1}{2}} \tilde{M}_i \tilde{M}_i^{-\frac{1}{2}} z_i} = \beta_i(\delta)||z_i||_2.$$

Because the probability distribution of $||z_i||_2$ is fixed, there exists some fixed $a > 0$ such that with probability at least $1 - \delta$, $||z_i||_2 \leq a$. So, for each i, with probability at least $1 - \delta$,

$$||\tilde{\boldsymbol{r}}_{i1} - \hat{\boldsymbol{r}}_i||_{\tilde{M}_i} = \beta_i(\delta)||z_i||_2 \leq \beta_i(\delta)a.$$

The previous statement can be combined with the high-probability inequality $||\hat{\boldsymbol{r}}_i - \overline{\boldsymbol{r}}||_{\tilde{M}_i} \leq m_{\max}\beta_i(\delta)$ to obtain that for each i, with probability at least $1 - \delta$,

$$||\tilde{\boldsymbol{r}}_{i1} - \overline{\boldsymbol{r}}||_{\tilde{M}_i} \leq ||\tilde{\boldsymbol{r}}_{i1} - \hat{\boldsymbol{r}}_i||_{\tilde{M}_i} + ||\hat{\boldsymbol{r}}_i - \overline{\boldsymbol{r}}||_{\tilde{M}_i} \leq (a + m_{\max})\beta_i(\delta).$$

Taking squares and dividing by $\beta_i(\delta)$ yields that for each i, with probability at least $1 - \delta$,

$$\frac{1}{\beta_i(\delta)^2}(\tilde{\boldsymbol{r}}_{i1} - \overline{\boldsymbol{r}})^T \tilde{M}_i(\tilde{\boldsymbol{r}}_{i1} - \overline{\boldsymbol{r}}) \leq (a + m_{\max})^2.$$

By assumption, $\frac{\lambda_d^{(i)}}{\beta_i(\delta)^2} \xrightarrow{D} \infty$ as $i \longrightarrow \infty$. Recall from Definition 7 that $\boldsymbol{v}_j^{(i)}$, $j \in \{1, \ldots, d\}$, represent the eigenvectors of $\tilde{M}_i$ corresponding to the eigenvalues $\lambda_j^{(i)}$. Then, with probability at least $1 - \delta$ for each i:

$$\begin{aligned}\frac{1}{\beta_i(\delta)^2}(\tilde{\boldsymbol{r}}_{i1} - \overline{\boldsymbol{r}})^T \tilde{M}_i(\tilde{\boldsymbol{r}}_{i1} - \overline{\boldsymbol{r}}) &= \frac{1}{\beta_i(\delta)^2}(\tilde{\boldsymbol{r}}_{i1} - \overline{\boldsymbol{r}})^T \left(\sum_{j=1}^{d} \lambda_j^{(i)} \boldsymbol{v}_j^{(i)} \boldsymbol{v}_j^{(i)T}\right)(\tilde{\boldsymbol{r}}_{i1} - \overline{\boldsymbol{r}}) \\ &= \frac{1}{\beta_i(\delta)^2} \sum_{j=1}^{d} \lambda_j^{(i)} \left((\tilde{\boldsymbol{r}}_{i1} - \overline{\boldsymbol{r}})^T \boldsymbol{v}_j^{(i)}\right)^2 \leq (a + m_{\max})^2.\end{aligned} \tag{B.11}$$

Since $\frac{\lambda_j^{(i)}}{\beta_i(\delta)^2} \longrightarrow \infty$ as $i \longrightarrow \infty$ for each j, and $\boldsymbol{v}_j^{(i)}$ is an orthonormal basis, the constant bound of $(a+m_{\max})^2$ in Eq. (B.11) is violated if we do not have $\tilde{\boldsymbol{r}}_{i1} - \overline{\boldsymbol{r}} \xrightarrow{D} \boldsymbol{0}$. Eq. (B.11) must hold with probability at least $1 - \delta$ independently for each iteration i, with the $(1 - \delta)$-probability due entirely to randomness in the posterior sampling distribution, given by Eq. (B.9). Therefore, it follows that $\tilde{\boldsymbol{r}}_{i1} \xrightarrow{D} \overline{\boldsymbol{r}}$. The proof that $\tilde{\boldsymbol{r}}_{i2} \xrightarrow{D} \overline{\boldsymbol{r}}$ with high probability is identical.

□

The analysis will show convergence in distribution of the reward samples, $\tilde{\boldsymbol{r}}_{i1}, \tilde{\boldsymbol{r}}_{i2} \xrightarrow{D} \overline{\boldsymbol{r}}$, by applying Lemma 7 and demonstrating that $\frac{1}{\beta_i(\delta)^2}\lambda_d^{(i)} \longrightarrow \infty$ as $i \longrightarrow \infty$. This result is proven by contradiction: intuitively, if $\frac{1}{\beta_i(\delta)^2}\lambda_d^{(i)}$ is upper-bounded, then

DPS has a lower-bounded probability of selecting policies that increase $\lambda_d^{(i)}$. First, Lemma 8 demonstrates that under the contradiction hypothesis, there is a non-decaying probability of sampling rewards $\tilde{\boldsymbol{r}}_{i1}, \tilde{\boldsymbol{r}}_{i2}$ that are highly-aligned with the eigenvector $\boldsymbol{v}_d^{(i)}$ of $\tilde{M}_i^{-1}$ corresponding to its largest eigenvalue, $(\lambda_d^{(i)})^{-1}$.

Lemma 8. *Assume that for a given iteration i, $\beta_i(\delta)^2 \left(\lambda_d^{(i)}\right)^{-1} \geq \alpha$. Then for any $a > 0$, the reward samples $\tilde{\boldsymbol{r}}_{i1}, \tilde{\boldsymbol{r}}_{i2}$ satisfy:*

$$P\left(\tilde{\boldsymbol{r}}_{i1}^T \boldsymbol{v}_d^{(i)} \geq a \max_{j<d} |\tilde{\boldsymbol{r}}_{i1}^T \boldsymbol{v}_j^{(i)}|\right) \geq c(a) > 0, \tag{B.12}$$

$$P\left(\tilde{\boldsymbol{r}}_{i2}^T \boldsymbol{v}_d^{(i)} \leq -a \max_{j<d} |\tilde{\boldsymbol{r}}_{i2}^T \boldsymbol{v}_j^{(i)}|\right) \geq c(a) > 0, \tag{B.13}$$

where $c : \mathbb{R}_+ \longrightarrow \mathbb{R}_+$ is a continuous, monotonically-decreasing function.

Proof. Recall that the reward samples $\tilde{\boldsymbol{r}}_{i1}, \tilde{\boldsymbol{r}}_{i2}$ are drawn according to the distribution in Eq. (B.9). The proof begins by demonstrating that the reward samples can equivalently be expressed as:

$$\tilde{\boldsymbol{r}}_{i1} = \hat{\boldsymbol{r}}_i + \beta_i(\delta) \sum_{j=1}^{d} \left(\lambda_j^{(i)}\right)^{-\frac{1}{2}} z_{ij} \boldsymbol{v}_j^{(i)}, z_{ij} \sim \mathcal{N}(0,1) \text{ i.i.d.}, \tag{B.14}$$

and similarly for $\tilde{\boldsymbol{r}}_{i2}$. Similarly to Eq. (B.9), the expression in Eq. (B.14) has a multivariate Gaussian distribution. One can take the expectation and covariance of Eq. (B.14) with respect to the variables $\{z_{ij}\}$ to show that they match the expressions in Eq. (B.9):

$$\mathbb{E}\left[\hat{\boldsymbol{r}}_i + \beta_i(\delta) \sum_{j=1}^{d} \left(\lambda_j^{(i)}\right)^{-\frac{1}{2}} z_{ij} \boldsymbol{v}_j^{(i)}\right] = \hat{\boldsymbol{r}}_i + \beta_i(\delta) \sum_{j=1}^{d} \left(\lambda_j^{(i)}\right)^{-\frac{1}{2}} \mathbb{E}[z_{ij}] \boldsymbol{v}_j^{(i)} = \hat{\boldsymbol{r}}_i,$$

$$\begin{aligned}
\text{Cov}\left[\hat{\boldsymbol{r}}_i + \beta_i(\delta) \sum_{j=1}^{d} \left(\lambda_j^{(i)}\right)^{-\frac{1}{2}} z_{ij} \boldsymbol{v}_j^{(i)}\right] &\overset{(a)}{=} \mathbb{E}\left[\left(\beta_i(\delta) \sum_{j=1}^{d} \left(\lambda_j^{(i)}\right)^{-\frac{1}{2}} z_{ij} \boldsymbol{v}_j^{(i)}\right)\left(\beta_i(\delta) \sum_{k=1}^{d} \left(\lambda_k^{(i)}\right)^{-\frac{1}{2}} z_{ik} \boldsymbol{v}_k^{(i)}\right)^T\right] \\
&= \beta_i(\delta)^2 \sum_{j=1}^{d} \sum_{k=1}^{d} \left(\lambda_j^{(i)}\right)^{-\frac{1}{2}} \left(\lambda_k^{(i)}\right)^{-\frac{1}{2}} \boldsymbol{v}_j^{(i)} \boldsymbol{v}_k^{(i)T} \mathbb{E}[z_{ij} z_{ik}] \\
&\overset{(b)}{=} \beta_i(\delta)^2 \sum_{j=1}^{d} \left(\lambda_j^{(i)}\right)^{-1} \boldsymbol{v}_j^{(i)} \boldsymbol{v}_j^{(i)T} = \beta_i(\delta)^2 \tilde{M}_i^{-1},
\end{aligned}$$

which match the expectation and covariance in Eq. (B.9). In the above, (a) applies the definition $\text{Cov}[\boldsymbol{x}] = \mathbb{E}[(\boldsymbol{x} - \mathbb{E}[\boldsymbol{x}])(\boldsymbol{x} - \mathbb{E}[\boldsymbol{x}])^T]$, and (b) holds because $\mathbb{E}[z_{ij} z_{ik}] = \text{Cov}[z_{ij} z_{ik}] = \delta_{jk}$, where δ_{jk} is the Kronecker delta function.

Next, it is shown that the probability that $\tilde{\boldsymbol{r}}_{i1}$ is arbitrarily-aligned with $\boldsymbol{v}_d^{(i)}$ is lower-bounded above zero: that is, there exists $c : \mathbb{R}_+ \longrightarrow \mathbb{R}_+$ such that for any $a > 0$, $P(\tilde{\boldsymbol{r}}_{i1}^T \boldsymbol{v}_d^{(i)} \geq a \max_{j<d} |\tilde{\boldsymbol{r}}_{i1}^T \boldsymbol{v}_j^{(i)}|) \geq c(a) > 0$. This can be shown by bounding the terms $|\tilde{\boldsymbol{r}}_{i1}^T \boldsymbol{v}_j^{(i)}|$, $j < d$, and $\tilde{\boldsymbol{r}}_{i1}^T \boldsymbol{v}_d^{(i)}$. Firstly, the term $|\tilde{\boldsymbol{r}}_{i1}^T \boldsymbol{v}_j^{(i)}|$, $j < d$, can be upper-bounded:

$$
\begin{aligned}
|\tilde{\boldsymbol{r}}_{i1}^T \boldsymbol{v}_j^{(i)}| &\overset{(a)}{=} \left| \hat{\boldsymbol{r}}_i^T \boldsymbol{v}_j^{(i)} + \beta_i(\delta) \sum_{k=1}^{d} \left(\lambda_k^{(i)}\right)^{-\frac{1}{2}} z_{ik} \boldsymbol{v}_k^{(i)T} \boldsymbol{v}_j^{(i)} \right| \overset{(b)}{=} \left| \hat{\boldsymbol{r}}_i^T \boldsymbol{v}_j^{(i)} + \beta_i(\delta) \left(\lambda_j^{(i)}\right)^{-\frac{1}{2}} z_{ij} \right| \\
&\leq |\hat{\boldsymbol{r}}_i^T \boldsymbol{v}_j^{(i)}| + \beta_i(\delta) \left(\lambda_j^{(i)}\right)^{-\frac{1}{2}} |z_{ij}| \overset{(c)}{\leq} ||\hat{\boldsymbol{r}}_i||_2 ||\boldsymbol{v}_j^{(i)}||_2 + \beta_i(\delta) \left(\lambda_j^{(i)}\right)^{-\frac{1}{2}} |z_{ij}| \\
&\overset{(d)}{\leq} b + \beta_i(\delta) \left(\lambda_j^{(i)}\right)^{-\frac{1}{2}} |z_{ij}|,
\end{aligned}
$$

where (a) applies Eq. (B.14), (b) follows from orthonormality of the eigenbasis, (c) follows from the Cauchy-Schwarz inequality, and (d) uses Lemma 4, which states that $||\hat{\boldsymbol{r}}_i||_2 \leq b$. Similarly, $\tilde{\boldsymbol{r}}_{i1}^T \boldsymbol{v}_d^{(i)}$ can be lower-bounded:

$$
\begin{aligned}
\tilde{\boldsymbol{r}}_{i1}^T \boldsymbol{v}_d^{(i)} &\overset{(a)}{=} \hat{\boldsymbol{r}}_i^T \boldsymbol{v}_d^{(i)} + \beta_i(\delta) \sum_{j=1}^{d} \left(\lambda_j^{(i)}\right)^{-\frac{1}{2}} z_{ij} \boldsymbol{v}_j^{(i)T} \boldsymbol{v}_d^{(i)} \overset{(b)}{=} \hat{\boldsymbol{r}}_i^T \boldsymbol{v}_d^{(i)} + \beta_i(\delta) \left(\lambda_d^{(i)}\right)^{-\frac{1}{2}} z_{i1} \\
&\geq -|\hat{\boldsymbol{r}}_i^T \boldsymbol{v}_d^{(i)}| + \beta_i(\delta) \left(\lambda_d^{(i)}\right)^{-\frac{1}{2}} z_{i1} \overset{(c)}{\geq} -||\hat{\boldsymbol{r}}_i||_2 ||\boldsymbol{v}_d^{(i)}||_2 + \beta_i(\delta) \left(\lambda_d^{(i)}\right)^{-\frac{1}{2}} z_{i1} \\
&\overset{(d)}{\geq} -b + \beta_i(\delta) \left(\lambda_d^{(i)}\right)^{-\frac{1}{2}} z_{i1},
\end{aligned}
$$

where as before, (a) applies Eq. (B.14), (b) follows from orthonormality of the eigenbasis, (c) follows from the Cauchy-Schwarz inequality, and (d) holds via Lemma 4. Given these upper and lower bounds, the probability $P\left(\tilde{\boldsymbol{r}}_{i1}^T \boldsymbol{v}_d^{(i)} \geq a \max_{j<d} |\tilde{\boldsymbol{r}}_{i1}^T \boldsymbol{v}_j^{(i)}|\right)$ can be lower-bounded:

$$
\begin{aligned}
P\left(\tilde{\boldsymbol{r}}_{i1}^T \boldsymbol{v}_d^{(i)} \geq a \max_{j<d} |\tilde{\boldsymbol{r}}_{i1}^T \boldsymbol{v}_j^{(i)}|\right) &\overset{(a)}{\geq} P\left(-b + \beta_i(\delta) \left(\lambda_d^{(i)}\right)^{-\frac{1}{2}} z_{i1} \geq a \max_{j<d} \left[b + \beta_i(\delta) \left(\lambda_j^{(i)}\right)^{-\frac{1}{2}} |z_{ij}| \right]\right) \\
&= P\left(z_{i1} \geq \frac{b\sqrt{\lambda_d^{(i)}}}{\beta_i(\delta)} + a \max_{j<d} \left[\frac{b\sqrt{\lambda_d^{(i)}}}{\beta_i(\delta)} + \sqrt{\frac{\lambda_d^{(i)}}{\lambda_j^{(i)}}} |z_{ij}| \right]\right) \\
&\overset{(b)}{\geq} P\left(z_{i1} \geq \frac{b}{\sqrt{\alpha}} + a \max_{j<d} \left[\frac{b}{\sqrt{\alpha}} + |z_{ij}| \right]\right) \\
&= P\left(z_{i1} \geq \frac{b(1+a)}{\sqrt{\alpha}} + a \max_{j<d} |z_{ij}| \right) := c(a) > 0,
\end{aligned}
$$

where (a) results from the upper and lower bounds derived above, and (b) follows because $\frac{\lambda_d^{(i)}}{\lambda_j^{(i)}} \leq 1$ and $\beta_i(\delta) \left(\lambda_d^{(i)}\right)^{-\frac{1}{2}} \geq \sqrt{\alpha}$ by assumption. The function $c(a) > 0$ is continuous and decreasing in a.

By identical arguments, $P(\tilde{\boldsymbol{r}}_{i2}^T \boldsymbol{v}_d^{(i)} \leq -a \max_{j<d} |\tilde{\boldsymbol{r}}_{i2}^T \boldsymbol{v}_j^{(i)}|) \geq c(a)$. Thus, for any $a > 0$ and set of eigenvectors $\boldsymbol{v}_j^{(i)}$:

$$P(\tilde{\boldsymbol{r}}_{i1}^T \boldsymbol{v}_d^{(i)} \geq a \max_{j<d} |\tilde{\boldsymbol{r}}_{i1}^T \boldsymbol{v}_j^{(i)}|) \geq c(a) > 0,$$
$$P(\tilde{\boldsymbol{r}}_{i2}^T \boldsymbol{v}_d^{(i)} \leq -a \max_{j<d} |\tilde{\boldsymbol{r}}_{i2}^T \boldsymbol{v}_j^{(i)}|) \geq c(a) > 0.$$

□

This alignment between sampled rewards $\tilde{\boldsymbol{r}}_{i1}$, $\tilde{\boldsymbol{r}}_{i2}$ and $\boldsymbol{v}_d^{(i)}$ can also be expressed as follows by taking a high enough:

Lemma 9. *Assume that for a given iteration i, $\beta_i(\delta)^2 \left(\lambda_d^{(i)}\right)^{-1} \geq \alpha$. Then, for any $\varepsilon > 0$, the following events have nonzero, uniformly-lower-bounded probability: $\left\| \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2} - \boldsymbol{v}_d^{(i)} \right\|_2 < \varepsilon$ and $\left\| \frac{\tilde{\boldsymbol{r}}_{i2}}{||\tilde{\boldsymbol{r}}_{i2}||_2} - (-\boldsymbol{v}_d^{(i)}) \right\|_2 < \varepsilon$.*

Proof. By Lemma 8, Eq.s (B.12) and (B.13) both hold. The events in Eq.s (B.12) and (B.13), $\{\tilde{\boldsymbol{r}}_{i1}^T \boldsymbol{v}_d^{(i)} \geq a \max_{j<d} |\tilde{\boldsymbol{r}}_{i1}^T \boldsymbol{v}_j^{(i)}|\}$ and $\{\tilde{\boldsymbol{r}}_{i2}^T \boldsymbol{v}_d^{(i)} \leq -a \max_{j<d} |\tilde{\boldsymbol{r}}_{i2}^T \boldsymbol{v}_j^{(i)}|\}$, will be referred to as events $A(a)$ and $B(a)$, respectively. From Lemma 8, $A(a)$ and $B(a)$ have positive, lower-bounded probability for any a. The proof argues that regardless of the specific eigenbasis, the values of a required for the result to hold have some fixed upper bound.

First, note that under events $A(a)$ and $B(a)$, as $a \longrightarrow \infty$, $\frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2} \longrightarrow \boldsymbol{v}_d^{(i)}$ and $\frac{\tilde{\boldsymbol{r}}_{i2}}{||\tilde{\boldsymbol{r}}_{i2}||_2} \longrightarrow -\boldsymbol{v}_d^{(i)}$. Let $\varepsilon > 0$. Under event $A(a)$ for sufficiently-large a, $\left\| \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2} - \boldsymbol{v}_d^{(i)} \right\|_2 < \varepsilon$. Define $a_{\min,1}(\varepsilon, \boldsymbol{v}_1^{(i)}, \ldots, \boldsymbol{v}_d^{(i)})$ as the minimum value of a such that $A(a)$ implies $\left\| \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2} - \boldsymbol{v}_d^{(i)} \right\|_2 \leq \frac{\varepsilon}{2}$, given the eigenbasis $\{\boldsymbol{v}_1^{(i)}, \ldots, \boldsymbol{v}_d^{(i)}\}$. Because the inequality defining $A(a)$ is continuous in a, $\tilde{\boldsymbol{r}}_{i1}$, and the eigenbasis $\{\boldsymbol{v}_1^{(i)}, \ldots, \boldsymbol{v}_d^{(i)}\}$, the function $a_{\min,1}(\varepsilon, \boldsymbol{v}_1^{(i)}, \ldots, \boldsymbol{v}_d^{(i)})$ is also continuous in the eigenbasis $\{\boldsymbol{v}_1^{(i)}, \ldots, \boldsymbol{v}_d^{(i)}\}$. Because $a_{\min,1}(\varepsilon, \boldsymbol{v}_1^{(i)}, \ldots, \boldsymbol{v}_d^{(i)})$ is positive for all $\{\boldsymbol{v}_1^{(i)}, \ldots, \boldsymbol{v}_d^{(i)}\}$, and the set of all eigenbases $\{\boldsymbol{v}_1^{(i)}, \ldots, \boldsymbol{v}_d^{(i)}\}$ is compact, there exists $a_{\min,1}(\varepsilon)$ such that for any eigenbasis, if $A(a)$ holds for $a \geq a_{\min,1}(\varepsilon)$, then $\left\| \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2} - \boldsymbol{v}_d^{(i)} \right\|_2 < \varepsilon$.

By the same arguments, there exists $a_{\min,2}(\varepsilon)$ such that for any eigenbasis, if $B(a)$ holds for $a \geq a_{\min,2}(\varepsilon)$, then $\left\| \frac{\tilde{\boldsymbol{r}}_{i2}}{||\tilde{\boldsymbol{r}}_{i2}||_2} - (-\boldsymbol{v}_d^{(i)}) \right\|_2 < \varepsilon$. Taking $a_{\min}(\varepsilon) := \max\{a_{\min,1}(\varepsilon), a_{\min,2}(\varepsilon)\}$, then for any $a \geq a_{\min}(\varepsilon)$, under events $A(a)$ and $B(a)$, both $\left\| \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2} - \boldsymbol{v}_d^{(i)} \right\|_2 < \varepsilon$ and $\left\| \frac{\tilde{\boldsymbol{r}}_{i2}}{||\tilde{\boldsymbol{r}}_{i2}||_2} - (-\boldsymbol{v}_d^{(i)}) \right\|_2 < \varepsilon$ hold. □

The next step shows that given sampled rewards $\tilde{\boldsymbol{r}}_{i1}, \tilde{\boldsymbol{r}}_{i2}$ that are highly-aligned with the eigenvector $\boldsymbol{v}_d^{(i)}$ of $\tilde{M}_i$ as in Lemma 9, there is a lower-bounded probability of sampling actions that have non-zero projection onto this eigenvector.

Importantly, for each possible eigenvector $\boldsymbol{v}_d^{(i)}$, the analysis will assume that there exist two actions $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{A}$ such that:

$$|(\boldsymbol{x}_2 - \boldsymbol{x}_1)^T \boldsymbol{v}_d^{(i)}| \geq c > 0, \tag{B.15}$$

where the constant c is independent of the eigenvector $\boldsymbol{v}_d^{(i)}$.

If this is not satisfied, then $\boldsymbol{x}^T \boldsymbol{v}_d^{(i)}$ is the same (or arbitrarily-similar) for all actions $\boldsymbol{x} \in \mathcal{A}$. In this case, the eigenvector $\boldsymbol{v}_d^{(i)}$ lies along a dimension that is irrelevant for learning the utilities $\overline{\boldsymbol{r}}$. The corresponding eigenvalue $\lambda_d^{(i)}$ is necessarily 0, since all vectors $\boldsymbol{x}_i$ are orthogonal to $\boldsymbol{v}_d^{(i)}$. Thus, the learning problem would remain equivalent if $\mathcal{A}$ were reduced to a lower-dimensional subspace to which $\boldsymbol{v}_d^{(i)}$ is orthogonal. Therefore, without loss of generality, this proof assumes that no such eigenvectors exist, that is, Eq. (B.15) is satisfied for all eigenvectors $\boldsymbol{v}_d^{(i)}$ of $\tilde{M}_i$.

Lemma 10 (Generalized linear dueling bandit)**.** *Assume that there exists i_0 such that for $i > i_0$, $\beta_i(\delta)^2 \left(\lambda_d^{(i)}\right)^{-1} \geq \alpha$. Then, there exists a constant $c' > 0$ such that for $i > i_0$:*

$$\mathbb{E}\left[\left|\boldsymbol{x}_i^T \boldsymbol{v}_d^{(i)}\right|\right] \geq c' > 0, \tag{B.16}$$

where $c' > 0$ depends only on the true utility parameters $\overline{\boldsymbol{r}}$, so that in particular, Eq. (B.16) *holds for any eigenvector $\boldsymbol{v}_d^{(i)}$.*

Proof. By Lemma 9, for any $\varepsilon > 0$, the following events have nonzero, uniformly-lower-bounded probability: $\left\|\frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2} - \boldsymbol{v}_d^{(i)}\right\|_2 < \varepsilon$ and $\left\|\frac{\tilde{\boldsymbol{r}}_{i2}}{||\tilde{\boldsymbol{r}}_{i2}||_2} - (-\boldsymbol{v}_d^{(i)})\right\|_2 < \varepsilon$.

Recall that in the bandit setting, actions are selected as follows in each iteration i:

$$\boldsymbol{x}_{i1} = \operatorname*{argsup}_{\boldsymbol{x} \in \mathcal{A}} \boldsymbol{x}^T \tilde{\boldsymbol{r}}_{i1}, \text{ and } \boldsymbol{x}_{i2} = \operatorname*{argsup}_{\boldsymbol{x} \in \mathcal{A}} \boldsymbol{x}^T \tilde{\boldsymbol{r}}_{i2}. \tag{B.17}$$

Note that for any $\boldsymbol{x} \in \mathcal{A}$ and unit vector $\boldsymbol{r}$, $f_1(\boldsymbol{r}) := \boldsymbol{x}^T \boldsymbol{r}$ is uniformly-continuous in $\boldsymbol{r}$: if $||\boldsymbol{r}_1 - \boldsymbol{r}_2||_2 \leq \delta$, then,

$$|f_1(\boldsymbol{r}_1) - f_1(\boldsymbol{r}_2)| \leq |\boldsymbol{x}^T(\boldsymbol{r}_1 - \boldsymbol{r}_2)| \leq ||\boldsymbol{x}||_2 ||\boldsymbol{r}_1 - \boldsymbol{r}_2||_2 \leq L\delta,$$

where $\boldsymbol{x} \leq L$ for all $\boldsymbol{x} \in \mathcal{A}$. Therefore, for any $\boldsymbol{v}_d^{(i)}$ and $\varepsilon' > 0$, there exists ε_1 such that when $\varepsilon < \varepsilon_1$:

$$\left|\boldsymbol{x}_{i1}^T \boldsymbol{v}_d^{(i)} - \boldsymbol{x}_{i1}^T \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}\right| < \varepsilon', \text{ and } \left|\boldsymbol{x}_{i2}^T(-\boldsymbol{v}_d^{(i)}) - \boldsymbol{x}_{i2}^T \frac{\tilde{\boldsymbol{r}}_{i2}}{||\tilde{\boldsymbol{r}}_{i2}||_2}\right| < \varepsilon'. \tag{B.18}$$

Similarly, for any unit vector $\boldsymbol{r}$, $f_2(\boldsymbol{r}) := \sup_{\boldsymbol{x}\in\mathcal{A}} \boldsymbol{x}^T\boldsymbol{r}$ is uniformly-continuous in $\boldsymbol{r}$, since a continuous function over a compact set is uniformly-continuous, and the set of all unit vectors is compact. Therefore, for any $\boldsymbol{v}_d^{(i)}$ and $\varepsilon' > 0$, there exists ε_2 such that when $\varepsilon < \varepsilon_2$:

$$\begin{aligned} &\left|\sup_{\boldsymbol{x}\in\mathcal{A}} \boldsymbol{x}^T \boldsymbol{v}_d^{(i)} - \sup_{\boldsymbol{x}\in\mathcal{A}} \boldsymbol{x}^T \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}\right| < \varepsilon', \text{ and} \\ &\left|\sup_{\boldsymbol{x}\in\mathcal{A}} \boldsymbol{x}^T (-\boldsymbol{v}_d^{(i)}) - \sup_{\boldsymbol{x}\in\mathcal{A}} \boldsymbol{x}^T \frac{\tilde{\boldsymbol{r}}_{i2}}{||\tilde{\boldsymbol{r}}_{i2}||_2}\right| < \varepsilon'. \end{aligned} \tag{B.19}$$

Letting $\varepsilon \leq \min\{\varepsilon_1, \varepsilon_2\}$, $|\boldsymbol{x}_i^T \boldsymbol{v}_d^{(i)}|$ can be lower-bounded as follows, where c is defined as in Eq. (B.15):

$$\begin{aligned} 0 < c \leq \sup_{\boldsymbol{x}_1,\boldsymbol{x}_2\in\mathcal{A}} |(\boldsymbol{x}_2 - \boldsymbol{x}_1)^T \boldsymbol{v}_d^{(i)}| &= \left|\sup_{\boldsymbol{x}\in\mathcal{A}} \boldsymbol{x}^T \boldsymbol{v}_d^{(i)} - \inf_{\boldsymbol{x}\in\mathcal{A}} \boldsymbol{x}^T \boldsymbol{v}_d^{(i)}\right| \\ = \left|\sup_{\boldsymbol{x}\in\mathcal{A}} \boldsymbol{x}^T \boldsymbol{v}_d^{(i)} + \sup_{\boldsymbol{x}\in\mathcal{A}} \boldsymbol{x}^T (-\boldsymbol{v}_d^{(i)})\right| &\overset{(a)}{\leq} 2\varepsilon' + \left|\sup_{\boldsymbol{x}\in\mathcal{A}} \boldsymbol{x}^T \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2} + \sup_{\boldsymbol{x}\in\mathcal{A}} \boldsymbol{x}^T \frac{\tilde{\boldsymbol{r}}_{i2}}{||\tilde{\boldsymbol{r}}_{i2}||_2}\right| \\ \overset{(b)}{=} 2\varepsilon' + \left|\boldsymbol{x}_{i1}^T \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2} + \boldsymbol{x}_{i2}^T \frac{\tilde{\boldsymbol{r}}_{i2}}{||\tilde{\boldsymbol{r}}_{i2}||_2}\right| &\overset{(c)}{\leq} 4\varepsilon' + \left|\boldsymbol{x}_{i1}^T \boldsymbol{v}_d^{(i)} + \boldsymbol{x}_{i2}^T (-\boldsymbol{v}_d^{(i)})\right| = 4\varepsilon' + |\boldsymbol{x}_i^T \boldsymbol{v}_d^{(i)}|, \end{aligned}$$

where (a), (b), and (c) apply Eq.s (B.19), (B.17), and (B.18), respectively.

Let $\varepsilon' < \frac{c}{8}$. Then, $|\boldsymbol{x}_i^T \boldsymbol{v}_d^{(i)}| \geq \frac{c}{2} > 0$ when the events of Lemma 9 hold. Because the latter have a non-decaying probability of occurrence $\rho > 0$:

$$\mathbb{E}\left[\left|\boldsymbol{x}_i^T \boldsymbol{v}_d^{(i)}\right|\right] \geq \rho \frac{c}{8} = c' > 0.$$

□

With this result, one can finally prove the desired contradiction.

Lemma 11. *As* $i \longrightarrow \infty$, $\frac{\beta_i(\delta)^2}{\lambda_d^{(i)}} \xrightarrow{D} 0$, *where* $\lambda_d^{(i)}$ *is the minimum eigenvalue of* $\tilde{M}_i$.

Proof. First, one can show via contradiction that $\liminf_{i \longrightarrow \infty} \beta_i(\delta)^2 \left(\lambda_d^{(i)}\right)^{-1} = 0$. Assume that:

$$\liminf_{i \longrightarrow \infty} \beta_i(\delta)^2 \left(\lambda_d^{(i)}\right)^{-1} = 2\alpha > 0. \tag{B.20}$$

If Eq. (B.20) is true, then there exists i_0 such that for all $i \geq i_0$, $\beta_i(\delta)^2 \left(\lambda_d^{(i)}\right)^{-1} \geq \alpha$. The following assumes that $i \geq i_0$. Since $\beta_i(\delta)$ increases at most logarithmically in i, it suffices to show that $\lambda_d^{(i)}$ increases at least linearly on average to achieve a contradiction with Eq. (B.20).

Under the contradiction hypothesis, Lemmas 8 and 10 both hold. Due to Lemma 10, DPS will infinitely often, and at a non-decaying rate, sample pairs $\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}$ such that $|\boldsymbol{x}_i^T \boldsymbol{v}_d^{(i)}|$ is lower-bounded away from zero. At iteration n, this proof analyzes the effect of this guarantee upon $\lambda_d^{(n)}$. Using that $\lambda_d^{(n)}$ corresponds to the eigenvector $\boldsymbol{v}_d^{(n)}$ of $\tilde{M}_n$:

$$\begin{aligned}\lambda_d^{(n)} = \boldsymbol{v}_d^{(n)T} \tilde{M}_n \boldsymbol{v}_d^{(n)} &\overset{(a)}{\geq} \boldsymbol{v}_d^{(n)T} \left(m_{\min} M_n\right)) \boldsymbol{v}_d^{(n)} \overset{(b)}{=} m_{\min} \boldsymbol{v}_d^{(n)T} \left(\lambda I + \sum_{i=1}^{n-1} \boldsymbol{x}_i \boldsymbol{x}_i^T\right) \boldsymbol{v}_d^{(n)} \\ &\geq m_{\min} \left[\lambda + \sum_{i=1}^{n-1} \left(\boldsymbol{x}_i^T \boldsymbol{v}_d^{(n)}\right)^2\right], \end{aligned} \tag{B.21}$$

where (a) follows from Lemma 5 and (b) from the definition of M_n. Note that while the right-hand side expression in Eq. (B.21) depends upon $\boldsymbol{x}_i^T \boldsymbol{v}_d^{(n)}$ for $i < n$, Eq. (B.16) depends upon $\boldsymbol{x}_i^T \boldsymbol{v}_d^{(i)}$. Clearly, if $\boldsymbol{v}_d^{(i)}$ were constant in i, then the combination of Eq.s (B.16) and (B.21) would suffice to prove that $\lambda_d^{(n)} \longrightarrow \infty$ with at least a linear average rate; however, $\boldsymbol{v}_d^{(i)}$ can vary with i over the space of unit vectors in $\mathbb{R}^d$.

This proof leverages that $\boldsymbol{v}_d^{(i)}$ is a unit vector, that the set of unit vectors in $\mathbb{R}^d$ is compact, and that any infinite cover of a compact set has a finite subcover. In particular, for any $\varepsilon > 0$, there exist sets $S_1, \ldots, S_K \subset \mathbb{R}^d$, $K < \infty$, such that:

1. For $\boldsymbol{v} \in \mathbb{R}^d$ such that $||\boldsymbol{v}||_2 = 1$, $\boldsymbol{v} \in S_k$ for some $k \in \{1, \ldots, K\}$, and
2. If $\boldsymbol{v}_1, \boldsymbol{v}_2 \in S_k$, then $||\boldsymbol{v}_1 - \boldsymbol{v}_2|| < \varepsilon$.

It will be shown that there exists a sequence $(n_i) \in \mathbb{N}$ such that $\boldsymbol{v}_d^{(n_i)} \in S_k$ for fixed $k \in \{1, \ldots, K\}$, with the events $\boldsymbol{v}_d^{(n_i)} \in S_k$ corresponding to the indices (n_i) occurring at some non-decaying frequency. Then, by appropriately choosing ε, the

proof will use Eq. (B.21) and the mutual proximity of the vectors $\boldsymbol{v}_d^{(n_i)}$ to show that $\lambda_d^{(n)}$ increases with an at-least linear rate.

Observe that for any number of total iterations N, there exists an integer $k \in \{1, \ldots, K\}$ such that $\boldsymbol{v}_d^{(i)} \in S_k$ during at least $\frac{N}{K}$ iterations. Because K is a constant and $\frac{N}{K}$ is linear in N, the number of iterations in which $\boldsymbol{v}_d^{(i)} \in S_k$ is at least linear in N for some k. The right-hand sum in Eq. (B.21) can then be divided according to the indices (n_i) and the remaining indices:

$$\begin{aligned}\lambda_d^{(n_j)} &\geq m_{\min}\left[\lambda + \sum_{i=1}^{n_j-1}\left(\boldsymbol{x}_i^T \boldsymbol{v}_d^{(n_i)}\right)^2\right] \\ &= m_{\min}\left[\lambda + \sum_{i=1}^{j-1}\left(\boldsymbol{x}_{n_i}^T \boldsymbol{v}_d^{(n_i)}\right)^2 + \sum_{j=1; j\notin\{n_1,n_2,\ldots,n_{j-1}\}}^{n_j-1}\left(\boldsymbol{x}_j^T \boldsymbol{v}_d^{(n_i)}\right)^2\right]. \end{aligned} \tag{B.22}$$

The latter sum in Eq. (B.22) is non-decreasing in n_j, as all of its terms are non-negative. In the former sum, $||\boldsymbol{v}_d^{(n_j)} - \boldsymbol{v}_d^{(n_i)}|| < \varepsilon$ for each $i \in \{1, \ldots, j-1\}$. Defining $\boldsymbol{\delta} := \boldsymbol{v}_d^{(n_j)} - \boldsymbol{v}_d^{(n_i)}$, so that $||\boldsymbol{\delta}||_2 \leq \varepsilon$:

$$\left(\boldsymbol{x}_{n_i}^T \boldsymbol{v}_d^{(n_i)}\right)^2 = \left(\boldsymbol{x}_{n_i}^T\left(\boldsymbol{v}_d^{(n_i)} + \boldsymbol{\delta}\right)\right)^2 = \left(\boldsymbol{x}_{n_i}^T \boldsymbol{v}_d^{(n_i)} + \boldsymbol{x}_{n_i}^T \boldsymbol{\delta}\right)^2 \geq \left(\left|\boldsymbol{x}_{n_i}^T \boldsymbol{v}_d^{(n_i)}\right| - \left|\boldsymbol{x}_{n_i}^T \boldsymbol{\delta}\right|\right)^2. \tag{B.23}$$

By the Cauchy-Schwarz inequality, $\left|\boldsymbol{x}_{n_i}^T \boldsymbol{\delta}\right| \leq ||\boldsymbol{\delta}||_2 * ||\boldsymbol{x}_{n_i}||_2 \leq 2\varepsilon h$, where h is the trajectory horizon. Because Eq. (B.16) requires that $\mathbb{E}\left[\left|\boldsymbol{x}_{n_i}^T \boldsymbol{v}_d^{(n_i)}\right|\right] \geq c' > 0$, one can choose ε small enough that $\mathbb{E}\left[\left|\boldsymbol{x}_{n_i}^T \boldsymbol{v}_d^{(n_i)}\right| - \left|\boldsymbol{x}_{n_i}^T \boldsymbol{\delta}\right|\right] \geq c' - 2\varepsilon h \geq c'' > 0$, and:

$$\mathbb{E}\left[\left(\boldsymbol{x}_{n_i}^T \boldsymbol{v}_d^{(n_i)}\right)^2\right] \overset{(a)}{\geq} \mathbb{E}\left[\left(\left|\boldsymbol{x}_{n_i}^T \boldsymbol{v}_d^{(n_i)}\right| - \left|\boldsymbol{x}_{n_i}^T \boldsymbol{\delta}\right|\right)^2\right] \overset{(b)}{\geq} \mathbb{E}\left[\left|\boldsymbol{x}_{n_i}^T \boldsymbol{v}_d^{(n_i)}\right| - \left|\boldsymbol{x}_{n_i}^T \boldsymbol{\delta}\right|\right]^2 \geq (c'')^2 > 0,$$

where (a) takes expectations of both sides of Eq. (B.23), and (b) follows from Jensen's inequality. Merging this result with Eq. (B.22) implies that $\lambda_d^{(n_j)}$ is expected to increase at least linearly on average, according to the positive constant c'', over the indices (n_i). Recall that there always exists an S_k such that the number of times when $\boldsymbol{v}_d^{(i)} \in S_k$ is at least linear in the total number of iterations N. Thus, the rate at which indices (n_i) occur is always (at least) linear in N on average, and $\lambda_d^{(n_j)}$ increases at least linearly in N in expectation.

One can now demonstrate that $\liminf_{i \longrightarrow \infty} \frac{\beta_i(\delta)^2}{\lambda_d^{(i)}} = 0$ holds: the numerator of $\frac{\beta_i(\delta)^2}{\lambda_d^{(i)}}$ is the square of a quantity that increases at most logarithmically in i, while the denominator

increases at least linearly in i on average. This contradicts the assumption in Eq. (B.20), and thereby proves that $\liminf_{i \longrightarrow \infty} \beta_i(\delta)^2 \left(\lambda_d^{(i)}\right)^{-1} = 0$ must hold.

Finally, one can leverage that $\liminf_{i \longrightarrow \infty} \beta_i(\delta)^2 \left(\lambda_d^{(i)}\right)^{-1} = 0$ to show that $\lim_{i \longrightarrow \infty} \beta_i(\delta)^2 \left(\lambda_d^{(i)}\right)^{-1} = 0$. Consider the following two possible cases: 1) $\beta_i(\delta)^2 \left(\lambda_d^{(i)}\right)^{-1}$ converges to zero in probability, and 2) $\beta_i(\delta)^2 \left(\lambda_d^{(i)}\right)^{-1}$ *does not* converge to zero in probability. In case 1), because convergence in probability implies convergence in distribution, the desired result holds.

In case 2), there exists some $\varepsilon > 0$ such that $P\left(\beta_i(\delta)^2 \left(\lambda_d^{(i)}\right)^{-1} \geq \varepsilon\right) \not\longrightarrow 0$. In this case, one can apply the same arguments used to show that $\liminf_{i \longrightarrow \infty} \beta_i(\delta)^2 \left(\lambda_d^{(i)}\right)^{-1} = 0$, but specifically over time indices where $\beta_i(\delta)^2 \left(\lambda_d^{(i)}\right)^{-1} \geq \varepsilon$. Due to the non-convergence in probability, these time indices must occur at some non-decaying rate; hence, the same analysis applies. Thus, $\lambda_d^{(i)}$ increases in i with at least a minimum linear average rate, while $\beta_i(\delta)$ increases at most logarithmically in i. This violates the non-convergence assumption of case 2), resulting in a contradiction. As a result, only case 1) can hold.

□

Combining Lemmas 7 and 11 leads to the desired result:

Proposition 4. *When running DPS in the generalized linear dueling bandit setting, with utilities given via either the linear or logistic link functions and with posterior sampling distributions given in Eq.* (4.13)*, then with probability* $1 - \delta$*, the sampled utilities* $\tilde{\boldsymbol{r}}_{i1}, \tilde{\boldsymbol{r}}_{i2}$ *converge in distribution to their true values,* $\tilde{\boldsymbol{r}}_{i1}, \tilde{\boldsymbol{r}}_{i2} \xrightarrow{D} \overline{\boldsymbol{r}}$*, as* $i \longrightarrow \infty$.

Asymptotic Consistency of the Utilities for Preference-Based RL

Proposition 4 can also be extended to the preference-based RL setting; in fact, several of the previous lemmas will be re-used in this analysis. The main complication is that policies are now selected rather than actions, and each policy corresponds to a distribution over possible trajectories.

The next result enables the proof to leverage convergence in distribution of the dynamics samples, $\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{p}}_{i2} \xrightarrow{D} \overline{\boldsymbol{p}}$ (as guaranteed by Proposition 3), in characterizing the impact of sampled policies upon convergence of the reward model.

Lemma 12. *Let $f : \mathbb{R}^{S^2A} \times \mathbb{R}^{SA} \longrightarrow \mathbb{R}$ be a function of transition dynamics $\boldsymbol{p} \in \mathbb{R}^{S^2A}$ and reward vector $\boldsymbol{r} \in \mathbb{R}^{SA}$, $f(\boldsymbol{p}, \boldsymbol{r})$, where f is continuous in $\boldsymbol{p}$ and uniformly-continuous in $\boldsymbol{r}$. Assume that Proposition 3 holds: $\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{p}}_{i2} \xrightarrow{D} \overline{\boldsymbol{p}}$. Then, for any $\delta, \varepsilon > 0$, there exists i' such that for $i > i'$, $|f(\overline{\boldsymbol{p}}, \boldsymbol{r}) - f(\tilde{\boldsymbol{p}}_{ij}, \boldsymbol{r})| < \varepsilon$ for any* unit vector $\boldsymbol{r}$ *and $j \in \{1, 2\}$ with probability at least $1 - \delta$.*

Proof. Without loss of generality, the result is shown for $j = 1$ (the steps are identical for $j = 1$ and $j = 2$). Applying Proposition 3, $\tilde{\boldsymbol{p}}_{i1} \xrightarrow{D} \overline{\boldsymbol{p}}$. By continuity of f, one can apply Fact 8 from Appendix B.1 to obtain that $f(\tilde{\boldsymbol{p}}_{i1}, \boldsymbol{r}) \xrightarrow{D} f(\overline{\boldsymbol{p}}, \boldsymbol{r})$ for any $\boldsymbol{r}$. Further applying Fact 9 from Appendix B.1, $f(\tilde{\boldsymbol{p}}_{i1}, \boldsymbol{r}) \xrightarrow{P} f(\overline{\boldsymbol{p}}, \boldsymbol{r})$ for any $\boldsymbol{r}$. By definition of convergence in probability, given δ, there exists $i_{\boldsymbol{r}}$ such that for $i \geq i_{\boldsymbol{r}}$:

$$|f(\tilde{\boldsymbol{p}}_{i1}, \boldsymbol{r}) - f(\overline{\boldsymbol{p}}, \boldsymbol{r})| < \frac{1}{3}\varepsilon \text{ with probability at least } 1 - \delta. \tag{B.24}$$

To obtain a high-probability bound that applies over all unit vectors $\boldsymbol{r}$, one can use compactness of the set of unit vectors. Any infinite cover of a compact set has a finite subcover; in particular, for any $\delta' > 0$, the set of unit vectors in $\mathbb{R}^d$ has a finite cover of the form $\{\mathcal{B}(\boldsymbol{r}_1, \delta'), \ldots, \mathcal{B}(\boldsymbol{r}_K, \delta')\}$, where $\{\boldsymbol{r}_1, \ldots, \boldsymbol{r}_K\}$ are unit vectors, and $\mathcal{B}(\boldsymbol{r}, \delta') := \{\boldsymbol{r}' \in \mathbb{R}^d \mid ||\boldsymbol{r}' - \boldsymbol{r}||_2 < \delta'\}$ is the d-dimensional sphere of radius δ' centered at $\boldsymbol{r}$. Thus, there exists a finite set of unit vectors $\mathcal{U} = \{\boldsymbol{r}_1, \ldots, \boldsymbol{r}_K\}$ such that for any unit vector $\boldsymbol{r}'$, $||\boldsymbol{r}_i - \boldsymbol{r}'||_2 < \delta'$ for some $i \in \{1, \ldots, K\}$. Because f is uniformly-continuous in $\boldsymbol{r}$, for any transition dynamics $\boldsymbol{p}$, there exists $\delta_{\boldsymbol{p}} > 0$ such that for any two unit vectors $\boldsymbol{r}, \boldsymbol{r}'$ such that $||\boldsymbol{r} - \boldsymbol{r}'||_2 < \delta_{\boldsymbol{p}}$:

$$|f(\boldsymbol{p}, \boldsymbol{r}) - f(\boldsymbol{p}, \boldsymbol{r}')| < \frac{1}{3}\varepsilon. \tag{B.25}$$

Without loss of generality, for each $\boldsymbol{p}$, define $\delta_{\boldsymbol{p}} := \sup x$ such that $||\boldsymbol{r} - \boldsymbol{r}'||_2 < x$ implies $|f(\boldsymbol{p}, \boldsymbol{r}) - f(\boldsymbol{p}, \boldsymbol{r}')| \leq \frac{1}{6}\varepsilon$. Then, because f is continuous in $\boldsymbol{p}$, $\delta_{\boldsymbol{p}}$ is also continuous in $\boldsymbol{p}$. Because the set of all possible transition probability vectors $\boldsymbol{p}$ is compact, and a continuous function over a compact set achieves its minimum value, there exists $\delta_{\min} > 0$ such that $\delta_{\boldsymbol{p}} \geq \delta_{\min} > 0$ over all $\boldsymbol{p}$. Let $\mathcal{U}$ be defined such that $\delta' \leq \delta_{\min}$; then, for any unit vector $\boldsymbol{r}'$, there exists $\boldsymbol{r} \in \mathcal{U}$ such that $||\boldsymbol{r} - \boldsymbol{r}'||_2 < \delta_{\min}$, and thus Eq. (B.25) holds for any $\boldsymbol{p}$.

By Eq. (B.24), for each $\boldsymbol{r}_j \in \mathcal{U}$, there exists $i_{\boldsymbol{r}_j}$ such that for $i \geq i_{\boldsymbol{r}_j}$: $||f(\tilde{\boldsymbol{p}}_{i1}, \boldsymbol{r}) - f(\overline{\boldsymbol{p}}, \boldsymbol{r})||_2 < \frac{1}{3}\varepsilon$ with probability at least $1 - \delta$. Because $\mathcal{U}$ is a finite set, there exists $i' > \max\{i_{\boldsymbol{r}_1}, \ldots, i_{\boldsymbol{r}_K}\}$ such that for $\boldsymbol{r} \in \mathcal{U}$ and $i > i'$:

$$|f(\tilde{\boldsymbol{p}}_{i1}, \boldsymbol{r}) - f(\overline{\boldsymbol{p}}, \boldsymbol{r})| < \frac{1}{3}\varepsilon \text{ for each } \boldsymbol{r} \in \mathcal{U} \text{ with probability at least } 1 - \delta. \tag{B.26}$$

Therefore, for any unit vector $\boldsymbol{r}'$, there exists $\boldsymbol{r} \in \mathcal{U}$ such that $||\boldsymbol{r} - \boldsymbol{r}'||_2 < \delta' \leq \delta_{\min}$, and with probability at least $1 - \delta$ for $i > i'$:

$$\begin{aligned}
|f(\overline{\boldsymbol{p}}, \boldsymbol{r}') - f(\tilde{\boldsymbol{p}}_{i1}, \boldsymbol{r}')| &= |f(\overline{\boldsymbol{p}}, \boldsymbol{r}') - f(\overline{\boldsymbol{p}}, \boldsymbol{r}) + f(\overline{\boldsymbol{p}}, \boldsymbol{r}) - f(\tilde{\boldsymbol{p}}_{i1}, \boldsymbol{r}) + f(\tilde{\boldsymbol{p}}_{i1}, \boldsymbol{r}) - f(\tilde{\boldsymbol{p}}_{i1}, \boldsymbol{r}')| \\
&\overset{(a)}{\leq} |f(\overline{\boldsymbol{p}}, \boldsymbol{r}') - f(\overline{\boldsymbol{p}}, \boldsymbol{r})| + |f(\overline{\boldsymbol{p}}, \boldsymbol{r}) - f(\tilde{\boldsymbol{p}}_{i1}, \boldsymbol{r})| + |f(\tilde{\boldsymbol{p}}_{i1}, \boldsymbol{r}) - f(\tilde{\boldsymbol{p}}_{i1}, \boldsymbol{r}')| \\
&\overset{(b)}{\leq} \frac{1}{3}\varepsilon + \frac{1}{3}\varepsilon + \frac{1}{3}\varepsilon = \varepsilon,
\end{aligned}$$

where (a) holds due to the triangle inequality, and (b) holds via Eq.s (B.25) and (B.26), where the proof showed that there exists $\delta_{\min}$ such that $0 < \delta_{\min} \leq \delta_{\boldsymbol{p}}$ for all possible transition dynamics parameters $\boldsymbol{p}$. □

Applying Lemma 12 to the two functions $V(\boldsymbol{p}, \boldsymbol{r}, \pi_{vi}(\boldsymbol{p}, \boldsymbol{r})) = \max_{\pi'} V(\boldsymbol{p}, \boldsymbol{r}, \pi')$ and $V(\boldsymbol{p}, \boldsymbol{r}, \pi)$, for any fixed policy π, yields the following result.

Lemma 13. *For any $\varepsilon, \delta > 0$, any policy π, and any unit reward vector $\boldsymbol{r}$, both of the following hold with probability at least $1 - \delta$ for sufficiently-large i and $j \in \{1, 2\}$:*

$$\begin{aligned}
&|V(\overline{\boldsymbol{p}}, \boldsymbol{r}, \pi) - V(\tilde{\boldsymbol{p}}_{ij}, \boldsymbol{r}, \pi)| < \varepsilon \\
&|V(\overline{\boldsymbol{p}}, \boldsymbol{r}, \pi_{vi}(\overline{\boldsymbol{p}}, \boldsymbol{r})) - V(\tilde{\boldsymbol{p}}_{ij}, \boldsymbol{r}, \pi_{vi}(\tilde{\boldsymbol{p}}_{ij}, \boldsymbol{r}))| < \varepsilon.
\end{aligned}$$

Proof. Both statements follow by applying Lemma 12. First, consider the function $f_1(\boldsymbol{p}, \boldsymbol{r}) := V(\boldsymbol{p}, \boldsymbol{r}, \pi)$ for a fixed policy π. The value function $V(\boldsymbol{p}, \boldsymbol{r}, \pi)$ is continuous in both $\boldsymbol{p}$ and $\boldsymbol{r}$, and furthermore, it is linear in $\boldsymbol{r}$. Using these observations, the proof demonstrates that $V(\boldsymbol{p}, \boldsymbol{r}, \pi)$ is uniformly-continuous in $\boldsymbol{r}$. Indeed, for any linear function of the form $g(\boldsymbol{z}) = \boldsymbol{a}^T \boldsymbol{z}$, for any $\varepsilon' > 0$, for $\delta' := \frac{\varepsilon'}{||\boldsymbol{a}||}$, and for any $\boldsymbol{z}_1, \boldsymbol{z}_2$ such that $||\boldsymbol{z}_1 - \boldsymbol{z}_2|| < \delta'$:

$$|g(\boldsymbol{z}_1) - g(\boldsymbol{z}_2)| = |\boldsymbol{a}^T(\boldsymbol{z}_1 - \boldsymbol{z}_2)| \leq ||\boldsymbol{a}||_2 ||\boldsymbol{z}_1 - \boldsymbol{z}_2||_2 < ||\boldsymbol{a}||_2 \delta' = \varepsilon'.$$

Thus, f_1 satisfies the conditions of Lemma 12 for any fixed π, so that for $i > i_\pi$, $|V(\overline{\boldsymbol{p}}, \boldsymbol{r}, \pi) - V(\tilde{\boldsymbol{p}}_{ij}, \boldsymbol{r}, \pi)| < \varepsilon$ with probability at least $1 - \delta$. Because there are finitely-many deterministic policies π, one can set $i > \max_\pi i_\pi$, so that the statement holds jointly over all π.

Next, let $f_2(\boldsymbol{p}, \boldsymbol{r}) = \max_\pi V(\boldsymbol{p}, \boldsymbol{r}, \pi) = V(\boldsymbol{p}, \boldsymbol{r}, \pi_{vi}(\boldsymbol{p}, \boldsymbol{r}))$. A maximum over finitely-many continuous functions is continuous, and a maximum over finitely-many uniformly-continuous functions is uniformly-continuous. Therefore, f_2 also satisfies the conditions of Lemma 12. □

As in the bandit case, convergence in distribution of the reward samples, $\tilde{\boldsymbol{r}}_{i1}, \tilde{\boldsymbol{r}}_{i2} \xrightarrow{D} \overline{\boldsymbol{r}}$, will be shown by applying Lemma 7 and demonstrating that $\frac{1}{\beta_i(\delta)^2}\lambda_d^{(i)} \longrightarrow \infty$ as $i \longrightarrow \infty$. Similarly, this result is proven by contradiction: intuitively, if $\frac{1}{\beta_i(\delta)^2}\lambda_d^{(i)}$ is upper-bounded, then DPS has a lower-bounded probability of selecting policies that tend to increase $\lambda_d^{(i)}$.

Importantly, abbreviating $\lambda_d^{(i)}$'s eigenvector $\boldsymbol{v}_d^{(i)}$ as $\boldsymbol{v}$, this proof is contingent upon there existing a pair of policies π_1, π_2 such that:

$$\begin{aligned}\left|\mathbb{E}[\boldsymbol{x}_i^T\boldsymbol{v} \mid \pi_{i1} = \pi_1, \pi_{i2} = \pi_2]\right| &= \left|\mathbb{E}[(\boldsymbol{x}_{i2} - \boldsymbol{x}_{i1})^T\boldsymbol{v} \mid \pi_{i1} = \pi_1, \pi_{i2} = \pi_2]\right| \\ &\overset{(a)}{=} |V(\overline{\boldsymbol{p}}, \boldsymbol{v}, \pi_1) - V(\overline{\boldsymbol{p}}, \boldsymbol{v}, \pi_2)| > 0,\end{aligned}$$

where (a) holds because the value function $V(\overline{\boldsymbol{p}}, \boldsymbol{v}, \pi)$ gives the expected total reward of π under the reward vector $\boldsymbol{v}$. In other words, the proof will require,

$$\max_{\pi_1,\pi_2} \left|\mathbb{E}[\boldsymbol{x}_i^T\boldsymbol{v} \mid \pi_{i1} = \pi_1, \pi_{i2} = \pi_2]\right| = \max_{\pi_1,\pi_2} |V(\overline{\boldsymbol{p}}, \boldsymbol{v}, \pi_1) - V(\overline{\boldsymbol{p}}, \boldsymbol{v}, \pi_2)| > 0. \quad \text{(B.27)}$$

If this does not hold, then it is impossible to select a pair of policies under which the observation $\boldsymbol{x}_i$ is not expected to be orthogonal to the eigenvector $\boldsymbol{v}$.

This work argues that without loss of generality, Eq. (B.27) can be assumed to hold for all eigenvectors of $\tilde{M}_i$. Note that if Eq. (B.27) does not hold, then $\mathbb{E}[\boldsymbol{x}_{i1}^T\boldsymbol{v} \mid \pi_{i1} = \pi] = V(\overline{\boldsymbol{p}}, \boldsymbol{v}, \pi)$ is fixed for all π. Given $\overline{\boldsymbol{p}}$, by linearity of the value function V in the rewards, any $\boldsymbol{v}$-directed component of $\boldsymbol{r}$ does not affect policy selection: $V(\overline{\boldsymbol{p}}, \boldsymbol{r}, \pi) = V(\overline{\boldsymbol{p}}, \boldsymbol{r}^{\boldsymbol{v}} + \boldsymbol{r}^{\boldsymbol{v}\perp}, \pi) = V(\overline{\boldsymbol{p}}, \boldsymbol{r}^{\boldsymbol{v}}, \pi) + V(\overline{\boldsymbol{p}}, \boldsymbol{r}^{\boldsymbol{v}\perp}, \pi)$, where $\boldsymbol{r}^{\boldsymbol{v}}$ is the projection of $\boldsymbol{r}$ onto the $\boldsymbol{v}$-direction and $\boldsymbol{r}^{\boldsymbol{v}\perp}$ is its orthogonal complement in $\mathbb{R}^d$. Because $V(\overline{\boldsymbol{p}}, \boldsymbol{r}^{\boldsymbol{v}}, \pi)$ does not depend on π, $\pi_{vi}(\overline{\boldsymbol{p}}, \boldsymbol{r}) = \text{argmax}_\pi V(\overline{\boldsymbol{p}}, \boldsymbol{r}, \pi) = \text{argmax}_\pi V(\overline{\boldsymbol{p}}, \boldsymbol{r}^{\boldsymbol{v}\perp}, \pi)$.

This thesis calls any vector $\boldsymbol{v}$ which does *not* satisfy Eq. (B.27) an *irrelevant dimension* of the rewards: given $\overline{\boldsymbol{p}}$, removing the $\boldsymbol{v}$-directed component of $\boldsymbol{r}$ does not influence policy selection. The following lemma demonstrates that such vectors remain irrelevant towards policy selection even when $\overline{\boldsymbol{p}}$ is unknown, given posterior samples of the transition dynamics, $\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{p}}_{i2}$, that have sufficiently converged to the true values $\overline{\boldsymbol{p}}$ in distribution.

Lemma 14. *For any reward vector $\boldsymbol{r} \in \mathbb{R}^d$, let $\boldsymbol{r}^{\text{rel}}$ be the projection of $\boldsymbol{r}$ onto the relevant subspace (for which Eq.* (B.27) *holds), and $\boldsymbol{r}^{\perp}$ be its orthogonal complement in $\mathbb{R}^d$, such that $\boldsymbol{r} = \boldsymbol{r}^{\text{rel}} + \boldsymbol{r}^{\perp}$, and $\boldsymbol{r}^{\perp}$ belongs to the subspace of irrelevant dimensions (where Eq.* (B.27) *does* not *hold). The reward samples on iteration i are $\tilde{\boldsymbol{r}}_{ij}$, $j \in$*

$\{1, 2\}$. *Then, for any* $\varepsilon, \delta > 0$*, there exists* i_0 *such that for* $i > i_0$*, with probability at least* $1 - \delta$*:*

$$|V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{ij}, \pi_{ij}) - V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{ij}, \pi_{vi}(\tilde{\boldsymbol{p}}_{ij}, \tilde{\boldsymbol{r}}^{\text{rel}}_{ij}))| < \varepsilon.$$

In other words, with respect to the sampled rewards $\tilde{\boldsymbol{r}}_{ij}$*, the expected reward of the selected policy* $\pi_{ij} = \pi_{vi}(\tilde{\boldsymbol{p}}_{ij}, \tilde{\boldsymbol{r}}_{ij})$ *is close to the expected reward of the policy that would have been selected were* $\tilde{\boldsymbol{r}}_{ij}$ *replaced by* $\tilde{\boldsymbol{r}}^{\text{rel}}_{ij}$*.*

Proof. The result is proven for $j = 1$ (the proof is identical for $j = 2$). Because $V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}^{\perp}_{i1}, \pi)$ is constant for all π, the variable $w := V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}^{\perp}_{i1}, \pi)$ is defined for convenience. First, the proof shows that under the true transition dynamics $\overline{\boldsymbol{p}}$, the irrelevant dimensions of $\tilde{\boldsymbol{r}}_{i1}$ do not affect policy selection. For any π,

$$\begin{aligned} V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1}, \pi_{vi}(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1})) &= \max_{\pi} V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1}, \pi) \overset{(a)}{=} \max_{\pi} \left[V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1}, \pi) + V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}^{\perp}_{i1}, \pi) \right] \\ &\overset{(b)}{=} V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1}, \pi_{vi}(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1})) + V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}^{\perp}_{i1}, \pi_{vi}(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1})) \overset{(c)}{=} V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1}, \pi_{vi}(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1})), \end{aligned} \tag{B.28}$$

where (a) and (c) hold because $\tilde{\boldsymbol{r}}_{i1} = \tilde{\boldsymbol{r}}^{\text{rel}}_{i1} + \tilde{\boldsymbol{r}}^{\perp}_{i1}$ and the value function is linear in the rewards, and (b) holds because $V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}^{\perp}_{i1}, \pi) = w$ is constant across all policies π.

To upper-bound $|V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1}, \pi_{i1}) - V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1}, \pi_{vi}(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1}))|$:

$$\begin{aligned} &|V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1}, \pi_{i1}) - V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1}, \pi_{vi}(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1}))| \qquad \text{(B.29)} \\ &\overset{(a)}{=} |V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1}, \pi_{vi}(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}_{i1})) - (V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1}, \pi_{vi}(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1})) + w)| \\ &= |V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1}, \pi_{vi}(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}_{i1})) - (V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1}, \pi_{vi}(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1})) + w) \\ &\quad - V(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}_{i1}, \pi_{vi}(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}_{i1})) + V(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}_{i1}, \pi_{vi}(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}_{i1}))| \\ &\quad - V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1}, \pi_{vi}(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1})) + V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1}, \pi_{vi}(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1}))| \\ &\quad - (V(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1}, \pi_{vi}(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1})) + w) + (V(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1}, \pi_{vi}(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1})) + w)| \\ &\overset{(b)}{\leq} |V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1}, \pi_{vi}(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}_{i1})) - V(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}_{i1}, \pi_{vi}(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}_{i1})| \\ &\quad + |V(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}_{i1}, \pi_{vi}(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}_{i1})) - V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1}, \pi_{vi}(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1})|} \\ &\quad + |V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1}, \pi_{vi}(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1}) - (V(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1}, \pi_{vi}(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1})) + w)| \qquad \text{(B.30)} \\ &\quad + |(V(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1}, \pi_{vi}(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1})) + w) - (V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1}, \pi_{vi}(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1})) + w)| \\ &\overset{(c)}{\leq} |V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1}, \pi_{vi}(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}_{i1})) - V(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}_{i1}, \pi_{vi}(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}_{i1})| \\ &\quad + |V(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}_{i1}, \pi_{vi}(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}_{i1})) - V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1}, \pi_{vi}(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1})| \\ &\quad + |V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1}, \pi_{vi}(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1})) - V(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1}, \pi_{vi}(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1}))| \\ &\quad + |V(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1}, \pi_{vi}(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1})) - V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1}, \pi_{vi}(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}^{\text{rel}}_{i1}))|, \qquad \text{(B.31)} \end{aligned}$$

where (a) applies $\tilde{\boldsymbol{r}}_{i1} = \tilde{\boldsymbol{r}}_{i1}^{\text{rel}} + \tilde{\boldsymbol{r}}_{i1}^{\perp}$, linearity of the value function in the rewards, and the definition of w; (b) rearranges terms and uses the triangle inequality; and (c) applies Eq. (B.28) to line (B.30), that is, $V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1}, \pi_{vi}(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1})) = V(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1}^{\text{rel}}, \pi_{vi}(\overline{\boldsymbol{p}}, \tilde{\boldsymbol{r}}_{i1}^{\text{rel}})) + w$.

Each of the four terms in Eq. (B.31) can be upper-bounded with high probability using Lemma 13. In particular, for large enough i, each term is less than $\frac{1}{4}\varepsilon$ with probability at least $1 - \frac{1}{4}\delta$. Therefore, the desired result holds. □

Remark 7. *The reward dimensions could be irrelevant due to a number of reasons. For instance, in preference-based RL, because the elements of* $\boldsymbol{x}_i := \boldsymbol{x}_{i2} - \boldsymbol{x}_{i1}$ *must sum to zero, the vector* $[1, 1, \ldots, 1]^T$ *must always be orthogonal to every observation* $\boldsymbol{x}_i$*. Alternatively, the MDP's transition dynamics could constrain the expected number of visits to a particular state to be constant regardless of the policy.*

Such constraints result in a subspace of $\mathbb{R}^d$ *that is irrelevant to learning the optimal policy once the transition dynamics model has converged sufficiently. Therefore, Lemma 7 must only be satisfied for eigenvalues of* M_i *along relevant dimensions in order to asymptotically select the optimal policy. Thus, the analysis can assume that sampled reward vectors* $\tilde{\boldsymbol{r}}_{i1}, \tilde{\boldsymbol{r}}_{i2}$ *have been projected onto the relevant subspace of* $\mathbb{R}^d$*. As a result, in proving that* $\frac{\beta_i(\delta)^2}{\lambda_d^{(i)}} \xrightarrow{D} 0$ *as* $i \longrightarrow \infty$*, the analysis assumes that all eigenvectors of* $\tilde{M}_i$ *belong to the relevant subspace without loss of generality. More formally, without loss of generality, all eigenvectors* $\{\boldsymbol{v}_j^{(i)}\}$ *of* $\tilde{M}_i$ *are assumed to satisfy Eq.* (B.27)*.*

The analysis leverages Lemmas 8 and 9, which were proven in the bandit analysis, but also apply to the preference-based RL setting. Analogously to Lemma 10 in the bandit setting, the next step is to prove that $\mathbb{E}\left[\left|\boldsymbol{x}_i^T \boldsymbol{v}_d^{(i)}\right|\right] \geq c' > 0$; however, the proof is more involved in the RL setting, as it must account for the stochastic translation of policies to trajectories via the MDP transition dynamics. Intuitively, Lemmas 8 and 9 prove that under the contradiction hypothesis, there is a non-decaying probability of sampling rewards $\tilde{\boldsymbol{r}}_{i1}, \tilde{\boldsymbol{r}}_{i2}$ that are highly-aligned with the eigenvector $\boldsymbol{v}_d^{(i)}$ of $\tilde{M}_i$. Lemma 15, proven below, demonstrates that given such reward samples, there is a lower-bounded probability of sampling trajectories that have a non-zero projection onto this eigenvector.

Lemma 15 (Preference-based RL)**.** *Assume that there exists* i_0 *such that for* $i > i_0$*,* $\beta_i(\delta)^2 \left(\lambda_d^{(i)}\right)^{-1} \geq \alpha$*. Then, there exists* $i' \geq i_0$ *and a constant* $c' > 0$ *such that for*

$i > i'$:

$$\mathbb{E}\left[\left|\boldsymbol{x}_i^T \boldsymbol{v}_d^{(i)}\right|\right] \geq c' > 0, \tag{B.32}$$

where $c' > 0$ depends only on the MDP parameters $\overline{\boldsymbol{p}}$ and $\overline{\boldsymbol{r}}$, so that in particular, Eq. (B.32) *holds for any eigenvector $\boldsymbol{v}_d^{(i)}$.*

Proof. By Lemma 9, for any $\varepsilon > 0$, the following events have nonzero, uniformly-lower-bounded probability for all unit vectors $\boldsymbol{v}_d^{(i)}$:

$$\left\|\frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2} - \boldsymbol{v}_d^{(i)}\right\|_2 < \varepsilon \text{ and } \left\|\frac{\tilde{\boldsymbol{r}}_{i2}}{||\tilde{\boldsymbol{r}}_{i2}||_2} - (-\boldsymbol{v}_d^{(i)})\right\|_2 < \varepsilon.$$

Next, this proof shows that for small-enough ε, the event $\left\|\frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2} - \boldsymbol{v}_d^{(i)}\right\|_2 < \varepsilon$ implies that the expected reward accrued by π_{i1} with respect to $\boldsymbol{v}_d^{(i)}$—that is, $V(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_{i1})$—is close to the maximum possible expected reward with respect to $\boldsymbol{v}_d^{(i)}$: $\max_\pi V(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi) = V(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_{vi}(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}))$. (The same approach yields an equivalent result for $\tilde{\boldsymbol{r}}_{i2}$.)

Assume that $\left\|\frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2} - \boldsymbol{v}_d^{(i)}\right\|_2 < \varepsilon$, and let $\varepsilon' > 0$. For small-enough ε and when $\tilde{\boldsymbol{p}}_{i1}$ has sufficiently converged in distribution to $\overline{\boldsymbol{p}}$ (as is guaranteed to occur by Proposition 3), one can show that $|V(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_{vi}(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)})) - V(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_{i1})| < \varepsilon'$:

$$\begin{aligned}
&\left|V(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_{vi}(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)})) - V(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_{i1})\right| \qquad \text{(B.33)}\\
&\quad \overset{(a)}{=} \left|V\left(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_{vi}(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)})\right) - V\left(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_{vi}\left(\tilde{\boldsymbol{p}}_{i1}, \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}\right)\right)\right|\\
&\quad = \Bigg|V\left(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_{vi}(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)})\right) - V\left(\overline{\boldsymbol{p}}, \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}, \pi_{vi}\left(\overline{\boldsymbol{p}}, \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}\right)\right)\\
&\qquad + V\left(\overline{\boldsymbol{p}}, \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}, \pi_{vi}\left(\overline{\boldsymbol{p}}, \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}\right)\right) - V\left(\tilde{\boldsymbol{p}}_{i1}, \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}, \pi_{vi}\left(\tilde{\boldsymbol{p}}_{i1}, \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}\right)\right)\\
&\qquad + V\left(\tilde{\boldsymbol{p}}_{i1}, \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}, \pi_{vi}\left(\tilde{\boldsymbol{p}}_{i1}, \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}\right)\right) - V\left(\overline{\boldsymbol{p}}, \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}, \pi_{vi}\left(\tilde{\boldsymbol{p}}_{i1}, \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}\right)\right)\\
&\qquad + V\left(\overline{\boldsymbol{p}}, \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}, \pi_{vi}\left(\tilde{\boldsymbol{p}}_{i1}, \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}\right)\right) - V\left(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_{vi}\left(\tilde{\boldsymbol{p}}_{i1}, \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}\right)\right)\Bigg|\\
&\quad \overset{(b)}{\leq} \left|V\left(\overline{\boldsymbol{p}}, \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}, \pi_{vi}\left(\tilde{\boldsymbol{p}}_{i1}, \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}\right)\right) - V\left(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_{vi}\left(\tilde{\boldsymbol{p}}_{i1}, \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}\right)\right)\right| \qquad \text{(B.34)}
\end{aligned}$$

$$+\left|V\left(\overline{\boldsymbol{p}},\boldsymbol{v}_d^{(i)},\pi_{vi}(\overline{\boldsymbol{p}},\boldsymbol{v}_d^{(i)})\right)-V\left(\overline{\boldsymbol{p}},\frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2},\pi_{vi}\left(\overline{\boldsymbol{p}},\frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}\right)\right)\right| \tag{B.35}$$

$$+\left|V\left(\overline{\boldsymbol{p}},\frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2},\pi_{vi}\left(\overline{\boldsymbol{p}},\frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}\right)\right)-V\left(\tilde{\boldsymbol{p}}_{i1},\frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2},\pi_{vi}\left(\tilde{\boldsymbol{p}}_{i1},\frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}\right)\right)\right| \tag{B.36}$$

$$+\left|V\left(\tilde{\boldsymbol{p}}_{i1},\frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2},\pi_{vi}\left(\tilde{\boldsymbol{p}}_{i1},\frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}\right)\right)-V\left(\overline{\boldsymbol{p}},\frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2},\pi_{vi}\left(\tilde{\boldsymbol{p}}_{i1},\frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}\right)\right)\right|, \tag{B.37}$$

where (a) uses that $\pi_{i1} = \pi_{vi}(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}_{i1})$ by definition, and also that positive scaling of the reward argument of $\pi_{vi}(\boldsymbol{p}, \boldsymbol{r})$ does not affect its output; and (b) applies the triangle inequality. Next, the proof will show that each of Eq.s (B.34)-(B.37) can be upper-bounded by $\frac{1}{4}\varepsilon'$ (with high probability for Eq.s (B.36) and (B.37)) by an appropriate choice of ε and by utilizing that $\tilde{\boldsymbol{p}}_{i1} \xrightarrow{D} \overline{\boldsymbol{p}}$ (Proposition 3).

Beginning with line (B.34), because $V(\boldsymbol{p}, \boldsymbol{r}, \pi)$ is linear in $\boldsymbol{r}$, it is uniformly continuous in $\boldsymbol{r}$ for fixed transition dynamics $\boldsymbol{p}$ and policy π. So, for fixed dynamics $\overline{\boldsymbol{p}}$ and policy π and for any reward vector $\boldsymbol{r}$, there exists ε_π such that if $||\boldsymbol{r} - \boldsymbol{r}'|| < \varepsilon_\pi$, then $|V(\boldsymbol{p}, \boldsymbol{r}, \pi) - V(\boldsymbol{p}, \boldsymbol{r}', \pi)| < \frac{1}{4}\varepsilon'$. Because there are finitely-many deterministic policies, there exists $\varepsilon_1 > 0$ such that $\varepsilon_1 \leq \varepsilon_\pi$ for all π. Therefore, for any policy π, if $||\boldsymbol{r} - \boldsymbol{r}'||_2 < \varepsilon_1$, then $|V(\overline{\boldsymbol{p}}, \boldsymbol{r}, \pi) - V(\overline{\boldsymbol{p}}, \boldsymbol{r}', \pi)| < \frac{1}{4}\varepsilon'$. The expression in line (B.34) is thus upper-bounded by $\frac{1}{4}\varepsilon'$ if $\varepsilon < \varepsilon_1$.

To upper-bound line (B.35), observe that $V(\boldsymbol{p}, \boldsymbol{r}, \pi_{vi}(\boldsymbol{p}, \boldsymbol{r})) = \max_\pi V(\boldsymbol{p}, \boldsymbol{r}, \pi)$ is also uniformly continuous in $\boldsymbol{r}$ for fixed transition dynamics $\boldsymbol{p}$: the maximum over finitely-many uniformly continuous functions is also uniformly continuous. Thus, there exists $\varepsilon_2 > 0$ such that if $||\boldsymbol{r} - \boldsymbol{r}'||_2 < \varepsilon_2$, then:

$$|V(\overline{\boldsymbol{p}}, \boldsymbol{r}, \pi_{vi}(\overline{\boldsymbol{p}}, \boldsymbol{r})) - V(\overline{\boldsymbol{p}}, \boldsymbol{r}', \pi_{vi}(\overline{\boldsymbol{p}}, \boldsymbol{r}'))| < \frac{1}{4}\varepsilon'.$$

The expression in line (B.35) is thus upper-bounded by $\frac{1}{4}\varepsilon'$ if $\varepsilon < \varepsilon_2$.

To obtain high-probability upper bounds for lines (B.36) and (B.37), one can apply Lemma 13. For sufficiently-high i, each of the following holds with probability at least $1 - \frac{1}{2}\delta'$:

$$\left|V\left(\overline{\boldsymbol{p}},\frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2},\pi_{vi}\left(\overline{\boldsymbol{p}},\frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}\right)\right)-V\left(\tilde{\boldsymbol{p}}_{i1},\frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2},\pi_{vi}\left(\tilde{\boldsymbol{p}}_{i1},\frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}\right)\right)\right| < \frac{1}{4}\varepsilon',$$

$$\left|V\left(\tilde{\boldsymbol{p}}_{i1},\frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2},\pi\right)-V\left(\overline{\boldsymbol{p}},\frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2},\pi\right)\right| < \frac{1}{4}\varepsilon',$$

where the second statement holds for any policy π, and in particular for $\pi = \pi_{vi}\left(\tilde{\boldsymbol{p}}_{i1}, \frac{\tilde{\boldsymbol{r}}_{i1}}{||\tilde{\boldsymbol{r}}_{i1}||_2}\right)$.

Next, combine the bounds for lines (B.34)-(B.37) while setting $\varepsilon < \min\{\varepsilon_1, \varepsilon_2\}$ and $i > i'$. Then, for any $\varepsilon', \delta' > 0$, the above analysis has shown that by setting ε small enough and taking $i > i'$:

$$\left|V(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_{vi}(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)})) - V(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_{i1})\right| < \varepsilon' \text{ with probability at least } 1 - \delta'.$$

Combining with the analogous result for π_{i2} and $-\boldsymbol{v}_d^{(i)}$ yields that for any $\varepsilon', \delta' > 0$, there exists sufficiently-small ε and large-enough i' such that for $i > i'$:

$$\begin{aligned}&\left|V(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_{vi}(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)})) - V(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_{i1})\right| < \varepsilon' \text{ with probability at least } 1 - \delta', \text{ and} \\ &\left|V(\overline{\boldsymbol{p}}, -\boldsymbol{v}_d^{(i)}, \pi_{vi}(\overline{\boldsymbol{p}}, -\boldsymbol{v}_d^{(i)})) - V(\overline{\boldsymbol{p}}, -\boldsymbol{v}_d^{(i)}, \pi_{i2})\right| < \varepsilon' \text{ with probability at least } 1 - \delta'.\end{aligned} \tag{B.38}$$

Next, the proof will set ε' to a small enough number to achieve $\left|\mathbb{E}[\boldsymbol{x}_i^T \boldsymbol{v}_d^{(i)}]\right| > \varepsilon' > 0$. Firstly, note that $|\mathbb{E}[\boldsymbol{x}_i^T \boldsymbol{v}_d^{(i)}]|$ is maximized when setting $\pi_{i1} = \pi_{vi}(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)})$ and $\pi_{i2} = \pi_{vi}(\overline{\boldsymbol{p}}, -\boldsymbol{v}_d^{(i)})$:

$$\begin{aligned}&\max_{\pi_1,\pi_2}\left|\mathbb{E}[\boldsymbol{x}_i^T \boldsymbol{v}_d^{(i)}] \mid \pi_{i1} = \pi_1, \pi_{i2} = \pi_2\right| = \max_{\pi_1,\pi_2}\left|\mathbb{E}[\boldsymbol{x}_{i1}^T \boldsymbol{v}_d^{(i)} - \boldsymbol{x}_{i2}^T \boldsymbol{v}_d^{(i)}] \mid \pi_{i1} = \pi_1, \pi_{i2} = \pi_2\right| \\ &\quad = \left|\mathbb{E}[\boldsymbol{x}_{i1}^T \boldsymbol{v}_d^{(i)} - \boldsymbol{x}_{i2}^T \boldsymbol{v}_d^{(i)} \mid \pi_{i1} = \operatorname{argmax}_\pi \mathbb{E}[\boldsymbol{x}_{i1}^T \boldsymbol{v}_d^{(i)}], \pi_{i2} = \operatorname{argmin}_\pi \mathbb{E}[\boldsymbol{x}_{i2}^T \boldsymbol{v}_d^{(i)}]]\right| \\ &\quad = \left|\mathbb{E}[\boldsymbol{x}_{i1}^T \boldsymbol{v}_d^{(i)} - \boldsymbol{x}_{i2}^T \boldsymbol{v}_d^{(i)} \mid \pi_{i1} = \operatorname{argmax}_\pi \mathbb{E}[\boldsymbol{x}_{i1}^T \boldsymbol{v}_d^{(i)}], \pi_{i2} = \operatorname{argmax}_\pi \mathbb{E}[\boldsymbol{x}_{i2}^T (-\boldsymbol{v}_d^{(i)})]]\right| \\ &\quad = \left|\mathbb{E}[\boldsymbol{x}_{i1}^T \boldsymbol{v}_d^{(i)} - \boldsymbol{x}_{i2}^T \boldsymbol{v}_d^{(i)} \mid \pi_{i1} = \operatorname{argmax}_\pi V(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi), \pi_{i2} = \operatorname{argmax}_\pi V(\overline{\boldsymbol{p}}, -\boldsymbol{v}_d^{(i)}, \pi)]\right| \\ &\quad = \left|\mathbb{E}[\boldsymbol{x}_{i1}^T \boldsymbol{v}_d^{(i)} - \boldsymbol{x}_{i2}^T \boldsymbol{v}_d^{(i)} \mid \pi_{i1} = \pi_{vi}(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}), \pi_{i2} = \pi_{vi}(\overline{\boldsymbol{p}}, -\boldsymbol{v}_d^{(i)})]\right|.\end{aligned}$$

From Lemma 14 (also, see Remark 7), one can assume without loss of generality that for all $\boldsymbol{v}_d^{(i)}$, $\max_{\pi_1,\pi_2}\left|\mathbb{E}[\boldsymbol{x}_i^T \boldsymbol{v}_d^{(i)} \mid \pi_{i1} = \pi_1, \pi_{i2} = \pi_2]\right| > 0$.

Because $\left|\mathbb{E}[\boldsymbol{x}_i^T \boldsymbol{v}_d^{(i)} \mid \pi_{i1} = \pi_1, \pi_{i2} = \pi_2]\right|$ is continuous in $\boldsymbol{v}_d^{(i)}$ for fixed π_1, π_2, and a maximum over finitely-many continuous functions is continuous:

$$\max_{\pi_1,\pi_2}\left|\mathbb{E}[\boldsymbol{x}_i^T \boldsymbol{v}_d^{(i)} \mid \pi_{i1} = \pi_1, \pi_{i2} = \pi_2]\right| \text{ is also continuous in } \boldsymbol{v}_d^{(i)}.$$

Because $\boldsymbol{v}_d^{(i)}$ belongs to the compact set of unit vectors, the expression achieves a minimum positive value on the set of possible $\boldsymbol{v}_d^{(i)}$, and thus, there exists $\eta > 0$ such

that $\max_{\pi_1,\pi_2} \left|\mathbb{E}[\boldsymbol{x}_i^T \boldsymbol{v}_d^{(i)} \mid \pi_{i1} = \pi_1, \pi_{i2} = \pi_2]\right| \geq \eta > 0$. Setting $\varepsilon' := \frac{\eta}{3}$:

$$\begin{aligned}
0 < 3\varepsilon' = \eta &\leq \max_{\pi_1,\pi_2} \left|\mathbb{E}[\boldsymbol{x}_i^T \boldsymbol{v}_d^{(i)} \mid \pi_{i1} = \pi_1, \pi_{i2} = \pi_2]\right| \\
&= \left|\max_{\pi_1} \mathbb{E}[\boldsymbol{x}_{i1}^T \boldsymbol{v}_d^{(i)} \mid \pi_{i1} = \pi_1] - \min_{\pi_2} \mathbb{E}[\boldsymbol{x}_{i2}^T \boldsymbol{v}_d^{(i)} \mid \pi_{i2} = \pi_2]\right| \\
&= \left|\max_{\pi_1} V(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_1) - \min_{\pi_2} V(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_2)\right| \\
&= \left|\max_{\pi_1} V(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_1) + \max_{\pi_2} [-V(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_2)]\right| \\
&= \left|\max_{\pi_1} V(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_1) + \max_{\pi_2} V(\overline{\boldsymbol{p}}, -\boldsymbol{v}_d^{(i)}, \pi_2)\right| \\
&= \left|V(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_{vi}(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)})) + V(\overline{\boldsymbol{p}}, -\boldsymbol{v}_d^{(i)}, \pi_{vi}(\overline{\boldsymbol{p}}, -\boldsymbol{v}_d^{(i)}))\right| \\
&\overset{(a)}{\leq} \left|V(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_{i1}) + V(\overline{\boldsymbol{p}}, -\boldsymbol{v}_d^{(i)}, \pi_{i2})\right| + 2\varepsilon',
\end{aligned}$$

where (a) holds with probability at least $1 - 2\delta'$ by Eq. (B.38). Subtracting $2\varepsilon'$ from the right-hand side of inequality (a) and from $\eta = 3\varepsilon'$ yields that with probability at least $1 - 2\delta'$,

$$\varepsilon' < \left|V(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_{i1}) + V(\overline{\boldsymbol{p}}, -\boldsymbol{v}_d^{(i)}, \pi_{i2})\right| = \left|V(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_{i1}) - V(\overline{\boldsymbol{p}}, \boldsymbol{v}_d^{(i)}, \pi_{i2})\right| = \left|\mathbb{E}[\boldsymbol{x}_i^T \boldsymbol{v}_d^{(i)}]\right|.$$

This implies that:

$$\mathbb{E}\left[\left|\boldsymbol{x}_i^T \boldsymbol{v}_d^{(i)}\right|\right] \overset{(a)}{\geq} \left|\mathbb{E}\left[\boldsymbol{x}_i^T \boldsymbol{v}_d^{(i)}\right]\right| \geq c' > 0 \text{ for some positive } c' \text{ and for all } i > i' \text{ and } \boldsymbol{v}_d^{(i)},$$

where (a) holds via Jensen's inequality and $c' := \varepsilon'$. □

To finish proving that in preference-based RL, the utility model is asymptotically consistent, Lemma 11 can be applied exactly as in the bandit case. The proof of Lemma 11 is the same in the RL setting as in the bandit case, except that Lemma 10 must be replaced by Lemma 15, and Eq. (B.16) must be replaced by Eq. (B.32).

Combining Lemmas 7 and 11 leads to the final result:

Proposition 5. *When running DPS in the preference-based RL setting, with utilities given via either the linear or logistic link functions and with posterior sampling distributions given in Eq.* (4.13)*, then with probability* $1 - \delta$*, the sampled utilities* $\tilde{\boldsymbol{r}}_{i1}, \tilde{\boldsymbol{r}}_{i2}$ *converge in distribution to their true values,* $\tilde{\boldsymbol{r}}_{i1}, \tilde{\boldsymbol{r}}_{i2} \xrightarrow{D} \overline{\boldsymbol{r}}$*, as* $i \longrightarrow \infty$*.*

B.4 Asymptotic Consistency of the Selected Policies in DPS

From the asymptotic consistency of the dynamics and reward samples, one can show that the sampled policies converge in distribution to the optimal policy:

Theorem 1. *Assume that DPS is executed in the preference-based RL setting, with utilities given via either the linear or logistic link functions and with posterior sampling distributions given in Eq.* (4.13)*.*

With probability $1-\delta$*, the sampled policies* π_{i1}, π_{i2} *converge in distribution to the optimal policy,* π^**, as* $i \longrightarrow \infty$*. That is,* $P(\pi_{i1} = \pi^*) \longrightarrow 1$ *and* $P(\pi_{i2} = \pi^*) \longrightarrow 1$ *as* $i \longrightarrow \infty$*.*

Under the same conditions, with probability $1-\delta$*, in the generalized linear dueling bandit setting with a finite action space* $\mathcal{A}$*, the sampled actions converge in distribution to the optimal actions:* $P(\boldsymbol{x}_{i1} = \boldsymbol{x}^*) \longrightarrow 1$ *and* $P(\boldsymbol{x}_{i2} = \boldsymbol{x}^*) \longrightarrow 1$ *as* $i \longrightarrow \infty$*.*

Proof. First, consider the preference-based RL case.

It suffices to show that $P(\pi_{i1} = \pi^*) \longrightarrow 1$ as $i \longrightarrow \infty$, as the proof is identical for π_{i2}. From Propositions 3 and 5, respectively, $\tilde{\boldsymbol{p}}_{i1} \xrightarrow{D} \overline{\boldsymbol{p}}$ and $\tilde{\boldsymbol{r}}_{i1} \xrightarrow{D} \overline{\boldsymbol{r}}$ with probability $1-\delta$. The proof proceeds under the assumption that $\tilde{\boldsymbol{r}}_{i1} \xrightarrow{D} \overline{\boldsymbol{r}}$, i.e., that the probability-$(1-\delta)$ event occurs.

By Fact 8 in Appendix B.1, for each fixed π, $V(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}_{i1}, \pi) \xrightarrow{D} V(\overline{\boldsymbol{p}}, \overline{\boldsymbol{r}}, \pi)$, as value functions are continuous in the dynamics and reward parameters. Applying Fact 9 in Appendix B.1, for each fixed π and $\varepsilon > 0$:

$$P(|V(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}_{i1}, \pi) - V(\overline{\boldsymbol{p}}, \overline{\boldsymbol{r}}, \pi)| > \varepsilon) \longrightarrow 0 \text{ as } i \longrightarrow \infty. \tag{B.39}$$

Next, the value of ε is set to less than half of the smallest gap between the value of the optimal policy and the value of any suboptimal policy:

$$\varepsilon < \frac{1}{2}\left[\max_{\pi} V(\overline{\boldsymbol{p}}, \overline{\boldsymbol{r}}, \pi) - \max_{\pi \text{ s.t. } V(\overline{\boldsymbol{p}}, \overline{\boldsymbol{r}}, \pi) < \max_{\pi'} V(\overline{\boldsymbol{p}}, \overline{\boldsymbol{r}}, \pi')} V(\overline{\boldsymbol{p}}, \overline{\boldsymbol{r}}, \pi)\right].$$

Then, the probability of selecting a non-optimal policy can be upper-bounded by a quantity that decays with i:

$$\begin{aligned} P(\pi_{i1} \neq \pi^*) &\overset{(a)}{\leq} P\left(\bigcup_{\pi} \{|V(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}_{i1}, \pi) - V(\overline{\boldsymbol{p}}, \overline{\boldsymbol{r}}, \pi)| > \varepsilon\}\right) \\ &\overset{(b)}{\leq} \sum_{\pi} P\left(|V(\tilde{\boldsymbol{p}}_{i1}, \tilde{\boldsymbol{r}}_{i1}, \pi) - V(\overline{\boldsymbol{p}}, \overline{\boldsymbol{r}}, \pi)| > \varepsilon\right) \xrightarrow{(c)} 0 \text{ as } i \longrightarrow \infty, \end{aligned}$$

where (a) follows from the definition of ε, (b) follows from the union bound, and (c) holds due to Eq. (B.39).

In the generalized linear dueling bandit setting with a finite action space, the proof follows along identical lines, with the following differences: policies π are replaced by actions $\boldsymbol{x} \in \mathcal{A}$, there are no transition dynamics, and the bandit value function is defined as $V_b(\boldsymbol{r}, \boldsymbol{x}) := \boldsymbol{r}^T \boldsymbol{x}$ for $\boldsymbol{x} \in \mathcal{A}$ and reward vector $\boldsymbol{r}$. Proposition 4 is also used in place of Proposition 5. With these substitutions, the steps of the proof are the same as those above.

□

Appendix C

ADDITIONAL DETAILS ABOUT THE DUELING POSTERIOR SAMPLING EXPERIMENTS IN THE LINEAR AND LOGISTIC DUELING BANDIT SETTINGS

This appendix provides additional information about the DPS simulations in the linear and logistic dueling bandit settings, discussed in Chapter 4. In particular, it describes the two baseline algorithms compared against DPS and gives hyperparameter optimization details for both DPS and the baselines algorithms.

C.1 Baselines: Sparring with Upper Confidence Bound (UCB) Algorithms

Sparring, introduced in Ailon, Karnin, and Joachims (2014), is a framework for extending standard bandit algorithms—which expect to receive absolute, numerical reward feedback—to the dueling bandit setting. Sparring instantiates two standard bandit algorithms, called Player 1 and Player 2, which learn by directly competing against each other. In each learning iteration i, Player 1 selects an action $\boldsymbol{x}_{i1} \in \mathcal{A}$, while Player 2 selects an action $\boldsymbol{x}_{i2} \in \mathcal{A}$. These two actions are dueled against one another, and each of the players learns whether its action was preferred or dominated.

The two baseline algorithms couple Sparring with the linear upper confidence bound (UCB) algorithm (Abbasi-Yadkori, Pál, and Szepesvári, 2011) and the logistic UCB algorithm (Filippi et al., 2010), respectively. The linear and logistic UCB algorithms can be viewed as subroutines that fit within the Sparring framework proposed in Ailon, Karnin, and Joachims (2014). The linear and logistic UCB Sparring approaches are each described in further detail below.

Sparring with Linear UCB

This section describes the procedure for Sparring with linear UCB (Abbasi-Yadkori, Pál, and Szepesvári, 2011), outlined in Algorithm 15. Without loss of generality, consider Player 2 in the Sparring framework. In the i^{th} learning iteration, Player 2 selects action $\boldsymbol{x}_{i2}$ and receives $y_i \in \left\{-\frac{1}{2}, \frac{1}{2}\right\}$ as feedback, where $y_i = \frac{1}{2}$ if $\boldsymbol{x}_{i2} \succ \boldsymbol{x}_{i1}$, and $y_i = -\frac{1}{2}$ if $\boldsymbol{x}_{i1} \succ \boldsymbol{x}_{i2}$. (Analogously, Player 1 selects action $\boldsymbol{x}_{i1}$ and receives $-y_i$ as reward feedback.)

On each iteration, Player 2 uses Linear UCB to select an action. First, Player 2

Algorithm 15 Sparring with Linear UCB

$\mathcal{H}_0^{(1)} = \emptyset$ ▷ Initialize Player 1's history
$\mathcal{H}_0^{(2)} = \emptyset$ ▷ Initialize Player 2's history
for $i = 1, 2, \ldots$ **do**
  for $j \in \{1, 2\}$ **do**
    Player j calculates $M_i^{(j)}$ via Eq. (C.2) and $\mathcal{H}_i^{(j)}$
    Player j calculates $\hat{\boldsymbol{r}}_i^{(j)}$ via Eq. (C.1) and $\mathcal{H}_i^{(j)}$
    Player j selects $\boldsymbol{x}_{ij}$ via Eq. (C.4)
  end for
  Execute actions $\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}$ and observe feedback $y_i \in \left\{-\frac{1}{2}, \frac{1}{2}\right\}$
  $\mathcal{H}_i^{(1)} = \mathcal{H}_{i-1}^{(1)} \cup (\boldsymbol{x}_{i1}, -y_i)$ ▷ Update Player 1's history
  $\mathcal{H}_i^{(2)} = \mathcal{H}_{i-1}^{(2)} \cup (\boldsymbol{x}_{i2}, y_i)$ ▷ Update Player 2's history
end for

calculates a MAP estimate of the utilities $\overline{\boldsymbol{r}}$, analogously to Eq. (4.2). When Player 2's history contains $n-1$ trials, this is given by:

$$\hat{\boldsymbol{r}}_n^{(2)} := \left(M_n^{(2)}\right)^{-1} \sum_{i=1}^{n-1} y_i \boldsymbol{x}_{i2}, \text{ where} \tag{C.1}$$

$$M_n^{(2)} := \lambda I + \sum_{i=1}^{n-1} \boldsymbol{x}_{i2} \boldsymbol{x}_{i2}^T \text{ for } \lambda \geq 1. \tag{C.2}$$

Player 1 calculates a MAP estimate $\hat{\boldsymbol{r}}_n^{(1)}$ analogously, except that each iteration $i \in \{1, \ldots, n-1\}$, $\boldsymbol{x}_{i1}$ and $-y_i$ are substituted for $\boldsymbol{x}_{i2}$ and y_i in Eq.s (C.1) and (C.2).

In this work, the implementation of linear UCB adds an intercept parameter to the utility vector. This is accomplished by appending the element 1 to each action in $\mathcal{A}$ in the utility inference step, to obtain a $(d+1)$-dimensional action space; then, the inferred utility vector $\overline{\boldsymbol{r}}$ belongs to $\mathbb{R}^{d+1}$, and its $(d+1)^{\text{th}}$ element is the intercept. Including the intercept significantly improves the resulting algorithm's performance.

Player 2 then estimates a confidence region centered at $\hat{\boldsymbol{r}}_n^{(2)}$, and intuitively, chooses an action which optimistically maximizes the expected reward within that confidence set. More formally, let $\mathcal{R}_{n2}$ be Player 2's confidence set in iteration n. Then, the action $\boldsymbol{x}_{n2}$ is selected by solving the following optimization problem:

$$\boldsymbol{x}_{n2} = \underset{\boldsymbol{x} \in \mathcal{A}}{\operatorname{argmax}} \max_{\boldsymbol{r} \in \mathcal{R}_{n2}} \boldsymbol{r}^T \boldsymbol{x}. \tag{C.3}$$

The approach in Abbasi-Yadkori, Pál, and Szepesvári (2011) utilizes ellipsoidal confidence sets, of the form:

$$\mathcal{R}_{n2} = \left\{ \boldsymbol{r} \,\middle|\, ||\hat{\boldsymbol{r}}_n^{(2)} - \boldsymbol{r}||_{M_n^{(2)}} \leq \beta_n(\delta) \right\},$$

where $\beta_n(\delta)$ is a function that increases slowly in n according to $\sqrt{\log n}$, and whose definition is given in Eq. (4.4) (where in Eq. (4.4), M_n is now given by Eq. (C.2)).

It can be shown (Filippi et al., 2010) that the optimization in Eq. (C.3) is equivalent to:

$$\boldsymbol{x}_{n2} = \underset{\boldsymbol{x} \in \mathcal{A}}{\operatorname{argmax}} \left[\boldsymbol{x}^T \hat{\boldsymbol{r}}_n^{(2)} + \beta_n(\delta) ||\boldsymbol{x}||_{(M_n^{(2)})^{-1}} \right]. \tag{C.4}$$

This optimization problem is analogous for Player 1, except that Player 1 uses $\hat{\boldsymbol{r}}_n^{(1)}$ and $M_n^{(1)}$ instead of $\hat{\boldsymbol{r}}_n^{(2)}$ and $M_n^{(2)}$. In practice, it is straightforward to solve the optimization in Eq. (C.4) over a finite action space, since the objective can simply be computed over all possible actions.

Importantly, the values $\beta_n(\delta)$ in Eq. (C.4) enable Abbasi-Yadkori, Pál, and Szepesvári (2011) to prove worst-case regret guarantees, but yield relatively-conservative performance in practice. In fact, the performance can be significantly improved by replacing $\beta_n(\delta)$ by a constant $\beta > 0$, and then treating β as a tunable hyperparameter. This approach is taken in the current implementation, which optimizes the performance of Sparring with Linear UCB over two hyperparameters: β and λ.

In Algorithm 15, lines 5-7 summarize Linear UCB's procedure for selecting actions.

Sparring with Logistic UCB

Algorithm 16 outlines the procedure for Sparring with logistic UCB (Filippi et al., 2010), which extends linear UCB to model feedback generation via a sigmoidal link function. As described for linear UCB above, an intercept parameter is added to the utility vector. Within the Sparring framework, Player 2 calculates a utility estimate $\hat{\boldsymbol{r}}_n^{(2)}$ on the n^{th} learning iteration as follows:

$$\hat{\boldsymbol{r}}_n^{(2)} = \underset{\boldsymbol{r} \in \Theta}{\operatorname{argmin}} \left\| \sum_{i=1}^{n-1} \left(g_{\log}(\boldsymbol{x}_{i2}^T \boldsymbol{r}) \boldsymbol{x}_{i2} - \left(y_i + \frac{1}{2} \right) \boldsymbol{x}_{i2} \right) \right\|_{\left(M_n^{(2)}\right)^{-1}}, \tag{C.5}$$

where $g_{\log}$ is the sigmoidal link function, i.e. $g_{\log}(x) = (1 + e^{-x})^{-1}$, and $M_n^{(2)}$ is calculated using Eq. (C.2) as in the linear UCB case. The additive $\frac{1}{2}$ translates the preference outcomes to 0/1 labels. Lastly, Θ is a compact set of allowable utility vector estimates $\boldsymbol{r}$, where $\Theta \in \mathbb{R}^{d+1}$ due to the intercept term added to the utility model. Projecting the utility estimate onto the compact set Θ ensures that $\hat{\boldsymbol{r}}_n^{(2)}$ cannot have an arbitrarily-large magnitude.

Algorithm 16 Sparring with Logistic UCB

$\mathcal{H}_0^{(1)} = \emptyset$ ▷ Initialize Player 1's history
$\mathcal{H}_0^{(2)} = \emptyset$ ▷ Initialize Player 2's history
for $i = 1, 2, \ldots$ **do**
  for $j \in \{1, 2\}$ **do**
    Player j calculates $M_i^{(j)}$ via Eq. (C.2) and $\mathcal{H}_i^{(j)}$
    Player j calculates $\hat{\boldsymbol{r}}_n^{(j)\text{MAP}}$ via Eq. (C.7) and $\mathcal{H}_i^{(j)}$
    Player j calculates $\hat{\boldsymbol{r}}_n^{(j)}$ via Eq. (C.6) and $\mathcal{H}_i^{(j)}$
    Player j selects $\boldsymbol{x}_{ij}$ via Eq. (C.8)
  end for
  Execute actions $\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}$ and observe feedback $y_i \in \left\{-\frac{1}{2}, \frac{1}{2}\right\}$
  $\mathcal{H}_i^{(1)} = \mathcal{H}_{i-1}^{(1)} \cup (\boldsymbol{x}_{i1}, -y_i)$ ▷ Update Player 1's history
  $\mathcal{H}_i^{(2)} = \mathcal{H}_{i-1}^{(2)} \cup (\boldsymbol{x}_{i2}, y_i)$ ▷ Update Player 2's history
end for

Because solving the optimization problem in Eq. (C.5) is challenging in practice, the simulations use a computationally-simpler estimate of the utilities:

$$\hat{\boldsymbol{r}}_n^{(2)} = \underset{\boldsymbol{r}\in\Theta}{\operatorname{argmin}} \left\| \hat{\boldsymbol{r}}_n^{\text{MAP}} - \boldsymbol{r} \right\|_2, \tag{C.6}$$

where $\hat{\boldsymbol{r}}_n^{(2)\text{MAP}}$ is the MAP estimate of the utilities. Letting $\mathcal{D}_n^{(2)}$ be Player 2's dataset after $n-1$ preferences, such that $\mathcal{D}_n^{(2)} := \{\boldsymbol{x}_{12}, y_1, \boldsymbol{x}_{22}, y_2, \ldots, \boldsymbol{x}_{(n-1)2}, y_{n-1}\}$, the MAP estimate is given as follows (analogously to Eq. (4.7) for DPS):

$$\begin{aligned}
\hat{\boldsymbol{r}}_n^{(2)\text{MAP}} &= \underset{\boldsymbol{r}}{\operatorname{argmin}} \left[-\log p\left(\boldsymbol{r} \mid \mathcal{D}_n^{(2)}\right) \right] = \underset{\boldsymbol{r}}{\operatorname{argmin}} \left[-\log p(\boldsymbol{r}) - \log p\left(\mathcal{D}_n^{(2)} \mid \boldsymbol{r}\right) \right] \\
&\overset{(a)}{=} \underset{\boldsymbol{r}}{\operatorname{argmin}} \left\{ \lambda^{-1} ||\boldsymbol{r}||_2^2 - \log \left[\prod_{i=1}^{n-1} \frac{1}{1 + \exp(-2y_i \boldsymbol{x}_{i2}^T \boldsymbol{r})} \right] \right\} \\
&= \underset{\boldsymbol{r}}{\operatorname{argmin}} \left\{ \lambda^{-1} ||\boldsymbol{r}||_2^2 + \sum_{i=1}^{n-1} \log(1 + \exp(-2y_i \boldsymbol{x}_{i2}^T \boldsymbol{r})) \right\},
\end{aligned} \tag{C.7}$$

where step (a) applies the Gaussian prior $p(\boldsymbol{r}) \sim \mathcal{N}(\boldsymbol{0}, \lambda I)$ for some prior hyperparameter $\lambda > 0$. This optimization procedure is identical for Player 1, except that $\boldsymbol{x}_{i1}$ and $-y_i$ are substituted for $\boldsymbol{x}_{i2}$ and y_i, respectively. The optimization problem in Eq. (C.7) is convex, and therefore can be solved via standard convex optimizers.

In the n^{th} iteration, given $\hat{\boldsymbol{r}}_n^{(2)\text{MAP}}$ and $M_n^{(2)}$, Player 2 selects an action by solving the following optimization problem:

$$\boldsymbol{x}_{n2} = \underset{\boldsymbol{x}\in\mathcal{A}}{\operatorname{argmax}} \left[g_{\log}(\boldsymbol{x}^T \hat{\boldsymbol{r}}_n^{(2)\text{MAP}}) + \beta_n(\delta) ||\boldsymbol{x}||_{(M_n^{(2)})^{-1}} \right], \tag{C.8}$$

where $\beta_n(\delta)$ is defined in Filippi et al. (2010) and given in Eq. (4.11), and increases in n according to $\sqrt{\log n}$. As in the linear UCB case, the $\beta_n(\delta)$ values allow Filippi et al. (2010) to prove worst-case regret guarantees, but yield conservative behavior in practice. Thus, faster convergence can be achieved by replacing $\beta_n(\delta)$ by a constant, tunable hyperparameter $\beta > 0$. In this work, the performance of Sparring with logistic UCB is optimized over two hyperparameters in total: β and λ.

In Algorithm 16, lines 5-8 summarize the action selection process for logistic UCB.

C.2 Hyperparameter Optimization

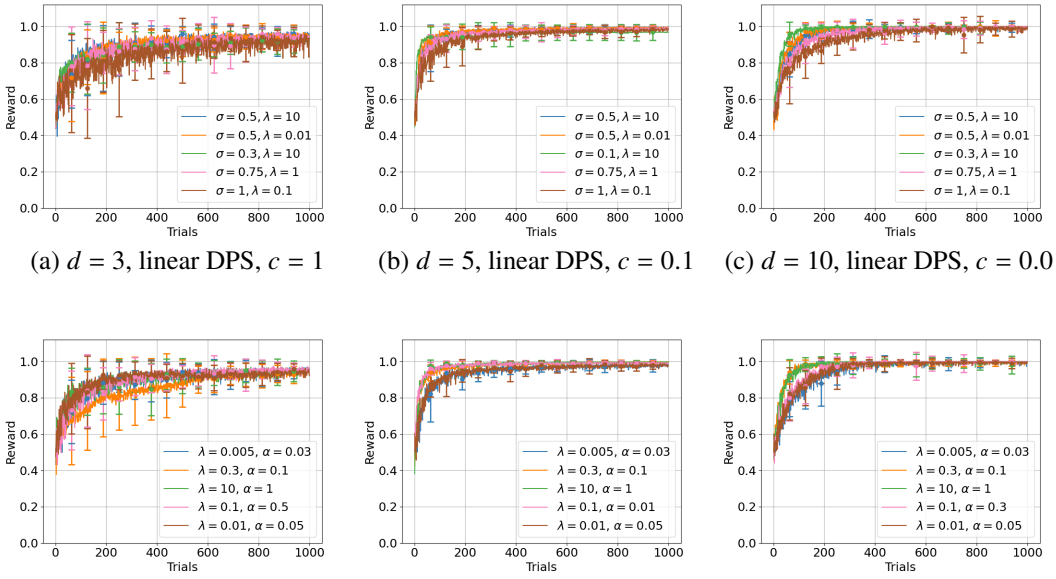
The linear and logistic dueling bandit simulations compare four algorithms: DPS with linear utility inference, DPS with logistic utility inference, Sparring with linear UCB, and Sparring with logistic UCB.

Table C.1 lists the hyperparameter ranges tested for these four algorithms, as well as the optimized hyperparameter values (these are the values used to generate the performance curve plots in Section 4.5). For linear and logistic UCB, the hyperparameters were optimized individually for each dimension d and preference noise level. In contrast, for each of the two versions of DPS, a single set of hyperparameters was identified that performed well across the different values of d and feedback noise levels considered. Note that for Sparring with logistic UCB, the simulations set Θ equal to a sphere of radius $10d$ centered at the origin. Hyperparameters were optimized by considering 20 simulation runs of each candidate set of values.

Finally, Figure C.1 compares the performance of both linear and logistic DPS across several of the different hyperparameter settings evaluated. Overall, DPS performs well and is robust to the choice of hyperparameter values to a certain degree.

Table C.1: Hyperparameters in the linear and logistic dueling bandit experiments. For the linear and logistic UCB algorithms, the hyperparameters were optimized individually for each dimension d and preference noise level. For each d, the best-performing hyperparameters are listed in the following order: logistic noise with $c = 0.01$, logistic noise with $c = 0.1$, logistic noise with $c = 1$, and linear noise with $c = 4$. For each of the two versions of DPS, a single set of hyperparameters was found that performed well across the different values of d and noise levels considered. Hyperparameters were optimized by performing 20 simulation runs of each candidate set of values.

Algorithm	Hyperparameter	Range Tested	Optimized Value
DPS (linear)	σ	[0.01, 5]	0.5
	λ	[0.01, 30]	10
DPS (logistic)	α	[0.01, 10]	0.03
	λ	[0.005, 10]	0.005
Sparring (linear UCB)	β	[0.001, 20]	$d = 3$: [0.5, 1, 1, 0.5]
			$d = 5$: [0.5, 1, 1, 5]
			$d = 10$: [1, 1, 1, 1]
	λ	[0.001, 20]	$d = 3$: [10, 5, 5, 0.05]
			$d = 5$: [0.01, 0.5, 20, 0.01]
			$d = 10$: [1, 1, 0.01, 0.001]
Sparring (linear UCB)	β	[0.001, 20]	$d = 3$: [0.5, 1, 0.01, 0.005]
			$d = 5$: [1, 1, 1, 0.005]
			$d = 10$: [0.5, 0.5, 0.5, 0.005]
	λ	[0.001, 20]	$d = 3$: [1, 5, 0.1, 0.005]
			$d = 5$: [1, 1, 0.5, 0.05]
			$d = 10$: [1, 1, 0.5, 0.001]



(a) $d = 3$, linear DPS, $c = 1$ (b) $d = 5$, linear DPS, $c = 0.1$ (c) $d = 10$, linear DPS, $c = 0.01$

(d) $d = 3$, logistic DPS, $c = 1$ (e) $d = 5$, logistic DPS, $c = 0.1$ (f) $d = 10$, logistic DPS, $c = 0.01$

Figure C.1: Hyperparameter sensitivity of DPS in the linear and logistic dueling bandit settings (mean ± std over 20 runs). The plots show DPS with utility inference via both Bayesian linear (a-c) and logistic (d-f) regression. The learning curves compare several sets of hyperparameters among those evaluated, for several different action space dimensionalities d and levels of logistic user feedback noise c. The plots normalize the rewards for each simulation run such that the best action in $\mathcal{A}$ has a reward of 1, while the worst action has a reward of 0. Overall, DPS performs well and is robust to the hyperparameter values to a certain degree.

Appendix D

ADDITIONAL DETAILS ABOUT THE DUELING POSTERIOR SAMPLING EXPERIMENTS IN THE PREFERENCE-BASED RL SETTING

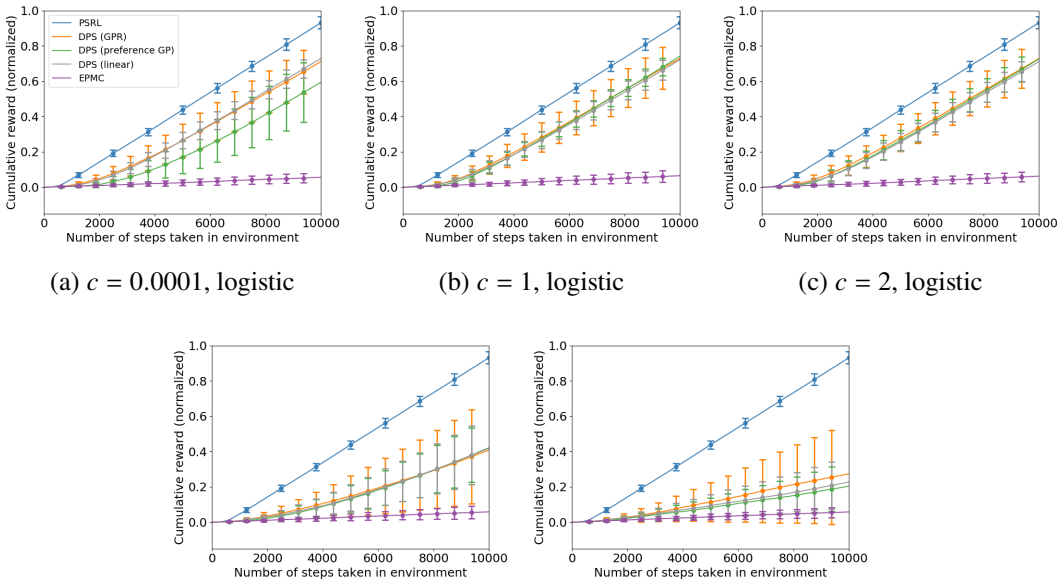
Python code for reproducing the DPS experiments in preference-based RL is located at: `https://github.com/ernovoseller/DuelingPosteriorSampling` (Novoseller et al., 2020a).

Experiments were conducted in three simulated environments, described in Section 4.5: RiverSwim and random MDPs (Osband, Russo, and Van Roy, 2013) and the simplified version of the Mountain Car problem described in Wirth, 2017. The first two cases used a fixed episode horizon of 50, while for the Mountain Car, episodes have a maximum length of 500, but terminate sooner if the agent reaches the goal state. Figures D.1, D.2, and D.3 display performance in the three environments for the five degrees of user preference noise evaluated. Experiments were run on an Ubuntu 16.04.3 machine with 32 GB of RAM and an Intel i7 processor. Some experiments were also run on an AWS server.

This section details the ranges of hyperparameter values tested for the different DPS credit assignment models, as well as the particular hyperparameters used in the displayed performance curves. Hyperparameters were tuned by considering mean performance over 30 experiment repetitions for each parameter setting considered. Only the least-noisy simulated user preference feedback (logistic noise, $c = 0.001$) was used to tune the hyperparameters; this value of c is sufficiently-small that the preferences are nearly-deterministic except for in tie cases, in which preferences are generated uniformly at random. For both Gaussian process-based credit assignment models, the squared exponential kernel was used:

$$k(\tilde{s}_i, \tilde{s}_j) = \sigma_f^2 \exp\left(-\frac{1}{2}\left(\frac{||f(\tilde{s}_i) - f(\tilde{s}_j)||}{l}\right)^2\right) + \sigma_n^2 \delta_{ij},$$

where σ_f^2 is the signal variance, l is the kernel lengthscale, σ_n^2 is the noise variance, δ_{ij} is the Kronecker delta function, and $f : \{1, \dots, S\} \times \{1, \dots, A\} \longrightarrow \mathbb{R}^m$ maps each state-action pair to an m-dimensional representation that encodes proximity between the state-action pairs. In the RiverSwim and Random MDP environments,



(a) $c = 0.0001$, logistic (b) $c = 1$, logistic (c) $c = 2$, logistic

(d) $c = 10$, logistic (e) $c = 100$, linear

Figure D.1: Empirical performance of DPS in the RiverSwim environment. Plots display mean +/- one standard deviation over 100 runs of each algorithm tested. Normalization is with respect to the total reward achieved by the optimal policy. Overall, DPS performs well and is robust to the choice of credit assignment model.

only lengthscales of $l = 0$ were considered, such that $k(\tilde{s}_i, \tilde{s}_j) = (\sigma_f^2 + \sigma_n^2)\delta_{ij}$; in these cases, f maps each state-action pair onto some unique index, and by convention, $\log \frac{0}{0} = 0$. For the Mountain Car, $f(\tilde{s}_i)$ mapped each state-action pair to a 3-dimensional vector of the form (position, velocity, action).

Please see Appendix A for definitions of the other hyperparameters in the credit assignment models. Tables D.2, D.3, and D.4 display both the tested ranges and optimized values (those appearing in the performance curves) for each case.

To model the transition dynamics, DPS uses a Dirichlet prior and posterior. To make the same prior assumptions with respect to each state-action pair, all parameters of the Dirichlet prior were set to the same value, $\alpha_0 > 0$: thus, the Dirichlet prior parameters are a length-d vector of the form, $\alpha_0 * [1, 1, \ldots, 1]^T$. For the RiverSwim and Random MDP environments, α_0 is set to 1, creating a uniform prior over all dynamics models. Because $\alpha_0 = 1$ is a reasonable choice when the numbers of states and actions are small, this work does not optimize performance over different α_0 values in the RiverSwim and Random MDP environments. For the Mountain Car problem, smaller prior values perform better because they favor sparse dynamics

distributions. For this environment, values of α_0 ranging from 0.0001 to 1 were tested, and 0.0005 was found to be the best-performing value among the ones tested.

The EMPC algorithm (Wirth and Fürnkranz, 2013a) has two hyperparameter values, α and η. Both of these were optimized jointly via a grid search over the values $(0.1, 0.2, \ldots, 0.9)$, with 100 repetitions of each pair of values. The best-performing hyperparameter values (i.e. those achieving the highest total reward) are displayed in Table D.1; these are the hyperparameter values depicted in the performance curve plots.

Finally, Figures D.4, D.5, and D.6 illustrate how DPS's performance varies as the hyperparameters are modified over a set of representative values from the tested ranges. These plots demonstrate that DPS is largely robust across many choices of model hyperparameters.

Table D.1: Hyperparameters for the EPMC baseline algorithm (Wirth and Fürnkranz, 2013a). Each table element shows the best-performing α/η values for the corresponding simulation environment and type of simulated user feedback (logistic or linear noise). For preference feedback with logistic noise, values of c are given in parentheses; larger values correspond to noisier preference feedback.

Noise:	Logistic (10)	Logistic (2)	Logistic (1)	Logistic (0.0001)	Linear
RiverSwim	0.1/0.8	0.3/0.7	0.1/0.2	0.8/0.8	0.3/0.1
Random MDPs	0.2/0.2	0.7/0.7	0.6/0.4	0.2/0.8	0.7/0.1

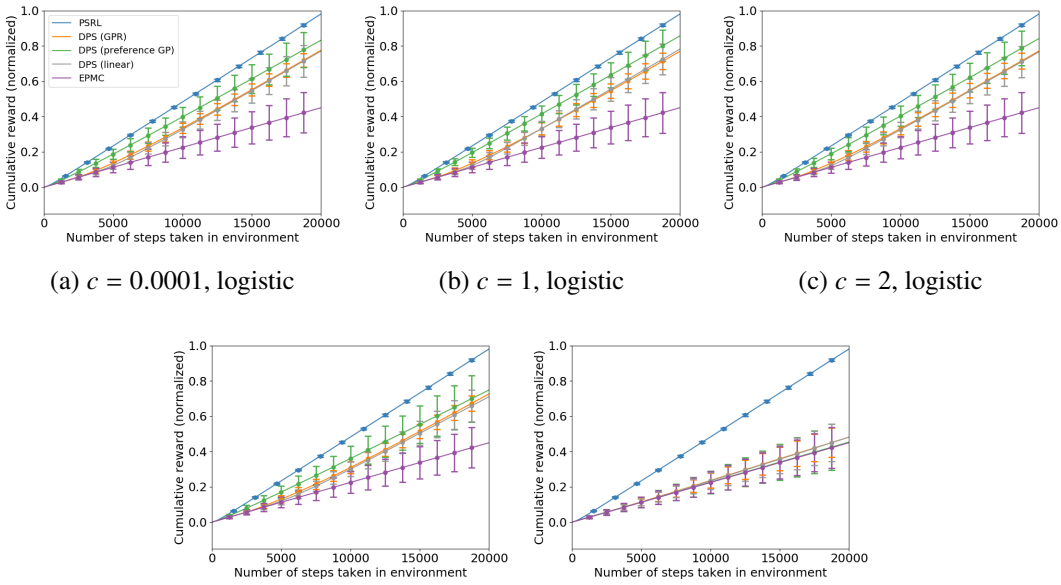
Noise:	Logistic (100)	Logistic (20)	Logistic (10)	Logistic (0.0001)	Linear
MountainCar	0.1/0.8	0.1/0.7	0.1/0.6	0.1/0.4	0.2/0.5

Table D.2: Credit assignment hyperparameters tested for the RiverSwim Environment.

Model	Hyperparameter	Range Tested	Optimized Value
Bayesian linear regression	σ	[0.05, 5]	0.5
	λ	[0.01, 10]	0.1
GP regression	σ_f^2	[0.001, 0.5]	0.1
	l	[0, 0] ([state, action])	0
	σ_n^2	[0.0001, 0.1]	0.001
GP preference (special case:	$\lambda = \sigma_f^2 + \sigma_n^2$	[0.1, 30]	1
Bayesian logistic regression)	α	[0.01, 1]	1
GP preference (varying c)	c	[1, 13]	N/A
	σ_f^2	[1]	
	l	[0, 0] ([state, action])	
	σ_n^2	[0.001]	
	α	[1]	

Table D.3: Credit assignment hyperparameters tested for the Random MDP Environment.

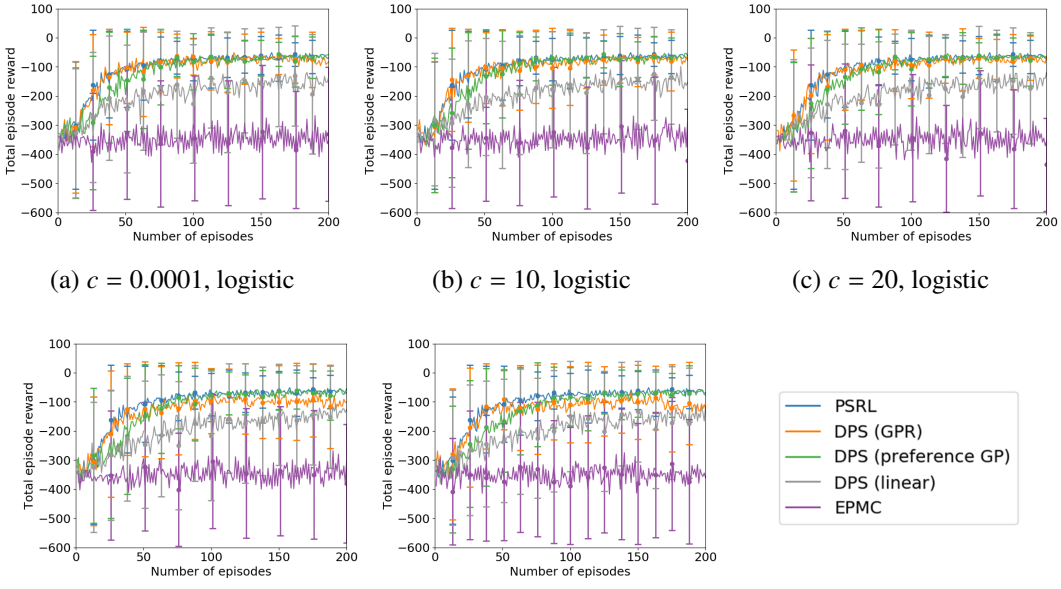
Model	Hyperparameter	Range Tested	Optimized Value
Bayesian linear regression	σ	[0.05, 5]	0.1
	λ	[0.01, 20]	10
GP regression	σ_f^2	[0.001, 1]	0.05
	l	[0, 0] ([state, action])	0
	σ_n^2	[0.0001, 0.1]	0.0005
GP preference (special case:	$\lambda = \sigma_f^2 + \sigma_n^2$	[1, 15]	0.1
Bayesian logistic regression)	α	[0.01, 1]	0.01
GP preference (varying c)	c	[0.0001, 1000]	N/A
	σ_f^2	[1]	
	l	[0, 0] ([state, action])	
	σ_n^2	[0.03]	
	α	[1]	



(a) $c = 0.0001$, logistic (b) $c = 1$, logistic (c) $c = 2$, logistic

(d) $c = 10$, logistic (e) c varies, linear

Figure D.2: Empirical performance of DPS in the Random MDP environment. Plots display mean +/- one standard deviation over 100 runs of each algorithm tested. Normalization is with respect to the total reward achieved by the optimal policy. Overall, DPS performs well and is robust to the choice of credit assignment model.



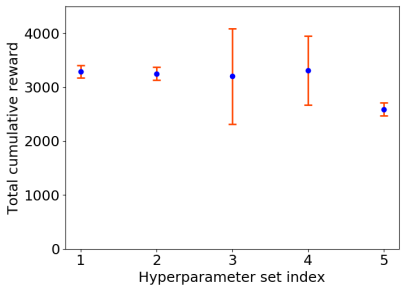
(a) $c = 0.0001$, logistic (b) $c = 10$, logistic (c) $c = 20$, logistic

(d) $c = 100$, logistic (e) $c = 1,000$, linear (f) Legend

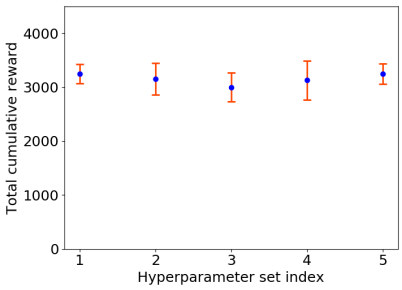
Figure D.3: Empirical performance of DPS in the Mountain Car environment. Plots display mean +/- one standard deviation over 100 runs of each algorithm tested. Overall, DPS performs well and is robust to the choice of credit assignment model.

Table D.4: Credit assignment hyperparameters tested for the Mountain Car Environment.

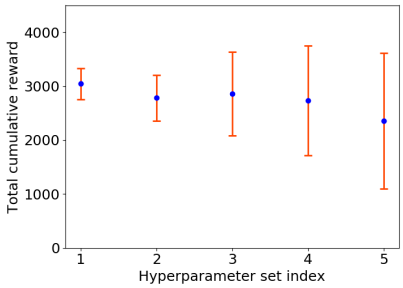
Model	Hyperparameter	Range Tested	Optimized Value
Bayesian linear regression	σ	[0.001, 30]	10
	λ	[0.001, 10]	1
GP regression	σ_f^2	[0.0001, 10]	0.01
	l	$[x, x, 0], x \in [1, 3]$ ([position, velocity, action])	$x = 2$
	σ_n^2	[1e-7, 0.01]	1e-5
GP preference (special case:	$\lambda = \sigma_f^2 + \sigma_n^2$	[0.0001, 10]	0.0001
Bayesian logistic regression)	α	[0.0001, 1]	0.01
GP preference (varying c)	c	[10, 10000]	N/A
	σ_f^2	[1]	
	l	[2, 2, 0] ([position, velocity, action])	
	σ_n^2	[0.001]	
	α	[1]	



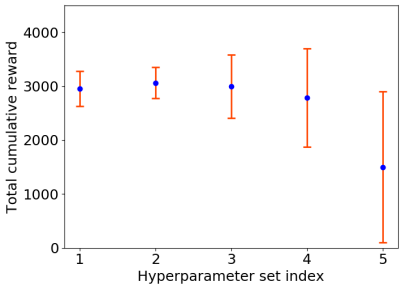
(a) Bayesian linear regression



(b) Gaussian process regression

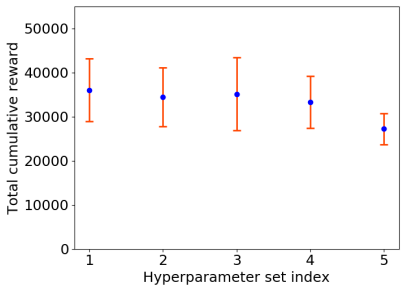


(c) Gaussian process preference model (special case: Bayesian logistic regression)

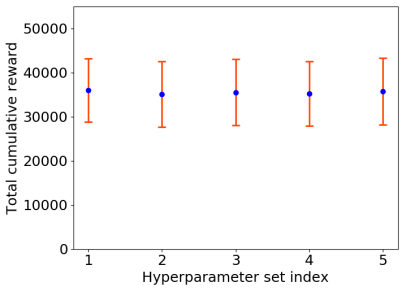


(d) Gaussian process preference model (varying c)

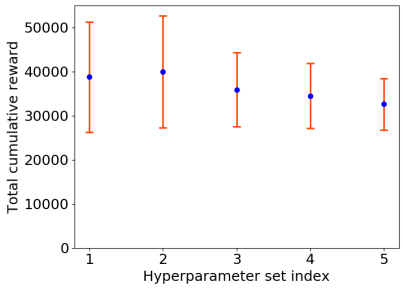
Figure D.4: Empirical performance of DPS in the RiverSwim environment for different hyperparameter combinations. Plots display mean +/- one standard deviation over 30 runs of each algorithm tested with logistic user noise and $c = 0.001$. Overall, DPS is robust to the choice of hyperparameters. The hyperparameter values depicted in each plot are (from left to right): for Bayesian linear regression, $(\sigma, \lambda) = \{(0.5, 0.1), (0.5, 10), (0.1, 0.1), (0.1, 10), (1, 0.1)\}$; for GP regression, $(\sigma_f^2, \sigma_n^2) = \{(0.1, 0.001), (0.1, 0.1), (0.01, 0.001), (0.001, 0.0001), (0.5, 0.1)\}$; for Bayesian logistic regression (special case of the GP preference model), $(\lambda, \alpha) = \{(1, 1), (30, 1), (20, 0.5), (1, 0.5), (30, 0.1)\}$; and additionally for the GP preference model, $c \in \{0.5, 1, 2, 5, 13\}$. See Table D.2 for the values of any hyperparameters not specifically mentioned here.



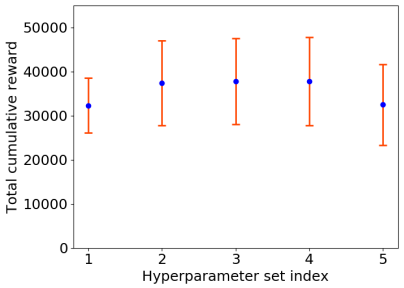
(a) Bayesian linear regression



(b) Gaussian process regression

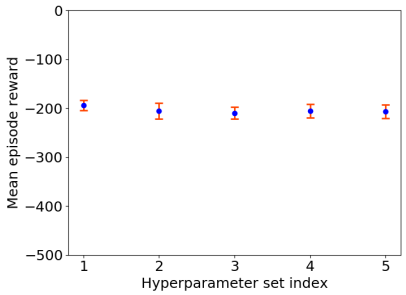


(c) Gaussian process preference model (special case: Bayesian logistic regression)

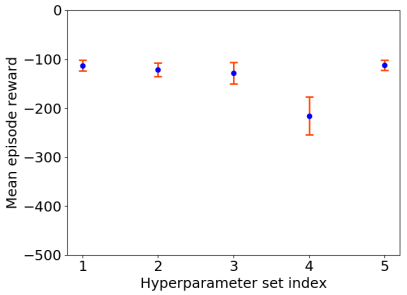


(d) Gaussian process preference model (varying c)

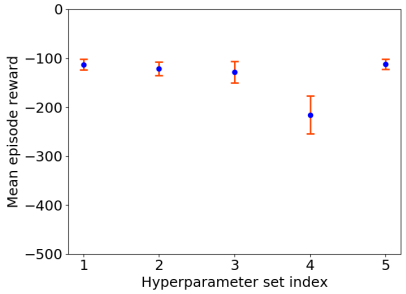
Figure D.5: Empirical performance of DPS in the Random MDP environment for different hyperparameter combinations. Plots display mean +/- one standard deviation over 30 runs of each algorithm tested with logistic user noise and $c = 0.001$. Overall, DPS is robust to the choice of hyperparameters. The hyperparameter values depicted in each plot are (from left to right): for Bayesian linear regression, $(\sigma, \lambda) = \{(0.1, 10), (0.1, 0.1), (0.05, 0.01), (0.5, 20), (1, 10)\}$; for GP regression, $(\sigma_f^2, \sigma_n^2) = \{(0.05, 0.0005), (0.001, 0.0001), (0.05, 0.1), (0.001, 0.0005), (1, 0.1)\}$; for Bayesian logistic regression (special case of the GP preference model), $(\lambda, \alpha) = \{(0.1, 0.01), (1, 0.01), (0.1, 1), (30, 0.1), (5, 0.5)\}$; and additionally for the GP preference model, $c \in \{1, 10, 15, 19, 100\}$. See Table D.3 for the values of any hyperparameters not specifically mentioned here.



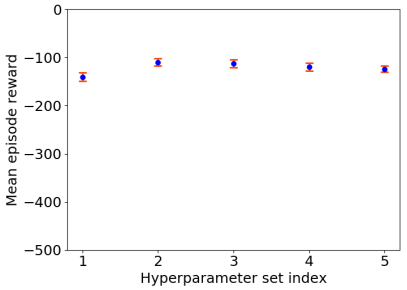
(a) Bayesian linear regression



(b) Gaussian process regression



(c) Gaussian process preference model (special case: Bayesian logistic regression)



(d) Gaussian process preference model (varying c)

Figure D.6: Empirical performance of DPS in the Mountain Car environment for different hyperparameter combinations. Plots display mean +/- one standard deviation over 30 runs of each algorithm tested with logistic user noise and $c = 0.001$. Overall, DPS is robust to the choice of hyperparameters. The hyperparameter values depicted in each plot are (from left to right): for Bayesian linear regression, $(\sigma, \lambda) = \{(10, 1), (10, 10), (30, 0.001), (0.001, 10), (0.1, 0.1)\}$; for GP regression, $(\sigma_f^2, l, \sigma_n^2) = \{(0.01, 2, 10^{-5}), (0.01, 1, 10^{-5}), (0.1, 2, 0.01), (1, 2, 0.001), (0.001, 3, 10^{-6})\}$; for Bayesian logistic regression (special case of the GP preference model), $(\lambda, \alpha) = \{(0.0001, 0.01), (0.1, 0.01), (0.0001, 0.0001), (0.001, 0.0001), (0.001, 0.01)\}$; and additionally for the GP preference model, $c \in \{10, 300, 400, 700, 1000\}$. See Table D.4 for the values of any hyperparameters not specifically mentioned here.