SPECTRAL RANKING REGRESSION

İLKAY YILDIZ*, BioSensics LLC, USA
JENNIFER DY, ECE Department, Northeastern University, USA
DENİZ ERDOĞMUŞ, ECE Department, Northeastern University, USA
SUSAN OSTMO, Casey Eye Institute, Oregon Health and Science University, USA
J. PETER CAMPBELL, Casey Eye Institute, Oregon Health and Science University, USA
MICHAEL F. CHIANG, National Eye Institute, National Institutes of Health, USA
STRATIS IOANNIDIS, ECE Department, Northeastern University, USA

We study the problem of ranking regression, in which a dataset of rankings is used to learn Plackett-Luce scores as functions of sample features. We propose a novel spectral algorithm to accelerate learning in ranking regression. Our main technical contribution is to show that the Plackett-Luce negative log-likelihood *augmented with a proximal penalty* has stationary points that satisfy the balance equations of a Markov Chain. This allows us to tackle the ranking regression problem via an efficient spectral algorithm by using the Alternating Directions Method of Multipliers (ADMM). ADMM separates the learning of scores and model parameters, and in turn, enables us to devise fast spectral algorithms for ranking regression via both shallow and deep neural network (DNN) models. For shallow models, our algorithms are up to 579 times faster than the Newton's method. For DNN models, we extend the standard ADMM via a Kullback-Leibler proximal penalty and show that this is still amenable to fast inference via a spectral approach. Compared to a state-of-the-art siamese network, our resulting algorithms are up to 175 times faster and attain better predictions by up to 26% Top-1 Accuracy and 6% Kendall-Tau correlation over five real-life ranking datasets.

CCS Concepts: • **Computing methodologies → Neural networks**; **Ranking**; **Learning to rank**; **Supervised learning by regression**; **Spectral methods**.

Additional Key Words and Phrases: Plackett-Luce, ranking, spectral methods, Markov Chain, ADMM

ACM Reference Format:
İlkay Yıldız, Jennifer Dy, Deniz Erdoğmuş, Susan Ostmo, J. Peter Campbell, Michael F. Chiang, and Stratis Ioannidis. 2022. SPECTRAL RANKING REGRESSION. *J. ACM* , , Article 170.2244 (2022), 39 pages. https://doi.org/10.1145/3530693

Authors' addresses: İlkay Yıldız, ilkyyldz95@gmail.com, BioSensics LLC, 57 Chapel St, Newton, MA, USA, 02458. Work performed at Northeastern University; Jennifer Dy, jdy@ece.neu.edu, ECE Department, Northeastern University, 409 Dana Research Center, 360 Huntington Avenue, Boston, MA, USA, 02115; Deniz Erdoğmuş, erdogmus@ece.neu.edu, ECE Department, Northeastern University, 409 Dana Research Center, 360 Huntington Avenue, Boston, MA, USA, 02115; Susan Ostmo, ostmo@ohsu.edu, Casey Eye Institute, Oregon Health and Science University, 3303 S Bond Ave Building 1, 11th Floor, Portland, OR, USA, 97239; J. Peter Campbell, campbelp@ohsu.edu, Casey Eye Institute, Oregon Health and Science University, 3303 S Bond Ave Building 1, 11th Floor, Portland, OR, USA, 97239; Michael F. Chiang, michael.chiang@nih.gov, National Eye Institute, National Institutes of Health, 10 Center Dr, Bethesda, MD, USA, 20814; Stratis Ioannidis, ioannidis@ece.neu.edu, ECE Department, Northeastern University, 409 Dana Research Center, 360 Huntington Avenue, Boston, MA, USA, 02115.

0004-5411/2022/-ART170.2244 $15.00
https://doi.org/10.1145/3530693

1 INTRODUCTION

Learning from ranking observations arises in many domains, including, e.g., econometrics [McFadden 1973; Ryzin and Mahajan 1999], psychometrics [Bradley and Terry 1952; Thurstone 1927], sports [Elo 1978], and medicine [Tian et al. 2019], to name a few. Ranking observations are (potentially noisy and incomplete) orderings of subsets of samples. Given a dataset of such ranking observations, probabilistic inference typically assumes the existence of an underlying total ordering and aims to recover it. The Plackett Luce model [Plackett 1975] is a prominent parametric model in this setting; it postulates that (a) each sample is parametrized by a score and (b) the probability that a sample is ranked higher than a set of alternatives is proportional to this score.

Plackett-Luce scores are traditionally learned from ranking observations via Maximum Likelihood Estimation (MLE) [Dykstra 1960; Hajek et al. 2014; Hunter 2004; Negahban et al. 2018]; under a reparametrization, the negative log-likelihood is convex and Plackett-Luce scores can be estimated via, e.g., Newton's method [Nocedal and Wright 2006]. Nevertheless, for large datasets, Newton's method can be prohibitively slow [Hunter 2004]. Recently, Maystre and Grossglauser [2015] proposed a highly efficient iterative spectral method, termed Iterative Luce Spectral Ranking (ILSR), that estimates Plackett-Luce scores significantly faster than state-of-the-art methods. ILSR relies on the fact that ML estimates of Plackett-Luce scores constitute the stationary distribution of a Markov Chain with transition rates defined by ranking observations. In turn, this stationary distribution can be found very efficiently via spectral techniques, e.g., the power method [Lei et al. 2016].

The above approaches learn Plackett-Luce scores in the absence of sample features, which precludes rank predictions on samples outside the training set. A natural variant of the above setting is *ranking regression*. In ranking regression, Plackett-Luce scores are assumed to be a parametric function of sample features; the parameters of this function are then regressed from ranking observations. Ranking regression has received considerable attention in the literature, via both shallow [Joachims 2002; Pahikkala et al. 2009; Tian et al. 2019] and deep models [Burges et al. 2005; Chang et al. 2016; Dubey et al. 2016; Han 2018; Yıldız et al. 2019]. Particularly in deep neural network (DNN) regression of Plackett-Luce scores, siamese neural networks [Bromley et al. 1994] perform extremely well: for example, Yıldız et al. [2019] train a 5.9M parameter siamese neural network from pairwise comparisons *on just* 80 *images*, attaining 0.92 AUC.

Virtually all existing work on ranking regression relies on classic optimization methods for parameter inference. To the best of our knowledge, the opportunity to accelerate learning in ranking regression via spectral methods such as ILSR [Maystre and Grossglauser 2015] has not yet been explored. One reason is that this is technically challenging: in contrast to the feature-less setting, it is not a priori evident how to devise a spectral method akin to ILSR for ranking regression via Plackett-Luce. Particularly, Plackett-Luce scores regressed from sample features *cannot* be directly expressed as stationary distributions of a Markov Chain.

Our main technical contribution is to show that the Plackett-Luce negative log-likelihood *augmented with a proximal penalty* has stationary points that satisfy the balance equations of a Markov Chain (c.f. Theorem 3.2). This allows us to tackle the ranking regression problem via a spectral method akin to ILSR by using Alternating Directions Method of Multipliers (ADMM) [Boyd et al. 2011]. Intuitively, ADMM decouples the optimization of Plackett-Luce scores from regression model parameters, encapsulating them in a proximal penalty. In turn, this can be solved highly efficiently via an ILSR-like spectral algorithm.

Overall, we make the following contributions.

- We propose the first approach to accelerate learning in ranking regression via spectral methods. As stated above, our main technical contribution is to show that the Plackett-Luce negative log-likelihood augmented with a proximal penalty has stationary points that satisfy the balance

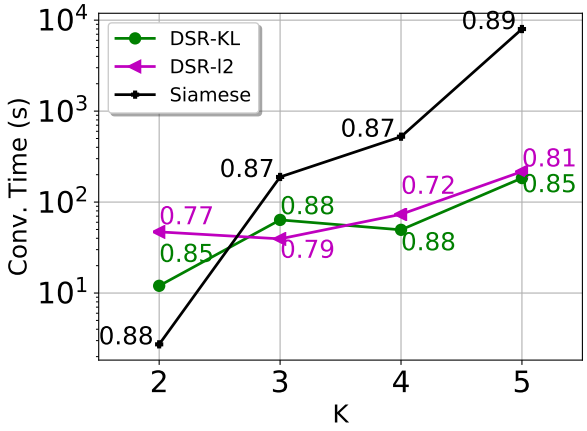


Fig. 1. Training time and Top-1 prediction accuracy (indicated next to each marker) vs. query size (K) on the Movehub-Cost dataset partitioned w.r.t. rank partitioning (c.f. Section 4.2). We compare Deep Spectral Ranking with a KL penalty (DSR-KL), Deep Spectral Ranking with an ℓ_2 penalty (DSR-l2), and a traditional siamese network. Siamese network training grows exponentially with K; both our spectral algorithms (DSR-KL and DSR-l2) scale more gracefully w.r.t. K. Moreover, DSR-KL has a considerably higher accuracy, comparable to (or better than) than the one attained by the less efficient siamese network.

equations of a Markov chain (c.f. Theorem 3.2). We further show that using ADMM to solve the ranking regression problem reduces it to the minimization of such an objective, which is thus amenable to a highly efficient spectral solution.

- We employ our spectral algorithm to regress Plackett-Luce scores via shallow regression functions of sample features, focusing on standard ADMM with ℓ_2-norm proximal penalty. Although the ranking regression problem is non-convex, we establish conditions that yield convergence guarantees for our spectral algorithm in the case of affine regression (c.f. Theorem 3.4).
- We also extend the application of our spectral algorithm to ranking regression via deep neural networks. We show experimentally that replacing the standard ℓ_2-norm proximal penalty in ADMM with Kullback-Leibler (KL) divergence [Kullback and Leibler 1951] has a better performance in this setting.

Our proposed algorithm yields significant performance improvements across the board. In the case of shallow ranking regression, our spectral algorithm is up to 579 times faster than traditional shallow ranking regression methods, and outperforms feature-less methods (including ILSR) by 13% accuracy. In the case of deep ranking regression, our algorithm significantly outperforms standard deep siamese networks, while the KL-divergence penalty attains better accuracy than the one attained by the ℓ_2-penalty ADMM. As shown in Figure 1, siamese network training grows exponentially with ranking size K; both KL-divergence (DSR-KL) and ℓ_2-penalty (DSR-l2) ADMM scale more gracefully w.r.t. K. However, DSR-KL has a considerably higher accuracy, comparable to the one attained by the less efficient siamese network.

The remainder of this paper is organized as follows. We review related literature in Sec. 2. We formulate our main contributions and proposed algorithms in Sec. 3. We present our experiments in Sec. 4 and conclude with future work in Sec. 5.

2 RELATED WORK AND TECHNICAL PRELIMINARY

2.1 Related Work

Rank Aggregation. The problem of *rank aggregation* [Dwork et al. 2001], in which a total ordering of samples is regressed from ranking observations, is classic; literature on the subject is vast—see, e.g., the surveys by Fligner and Verducci [1993], Cattelan [2012] and Marden [2014]. Probabilistic inference in this setting typically assumes (a) that a "true" total ordering of samples exists, and (b) that ranking observations exhibit a *stochastic transitivity property* [Agarwal 2016]: a sample is more likely to be ranked higher than another when this event is consistent with the underlying total ordering. The noisy permutation model is a non-parametric model for this setting: pairwise comparisons consistent with the underlying total ordering are observed under i.i.d. Bernoulli noise. Maximum likelihood estimation (MLE) is NP-hard in this setting. A polytime algorithm by Braverman and Mossel [2008] recovers the underlying ordering in $\Theta(n \log n)$ comparisons, w.h.p.; this is tightened by several recent works [Mao et al. 2018a,b; Wauthier et al. 2013]. The Mallows model [Mallows 1957] assumes that the probability of a ranking observation is a decreasing function of its distance from the underlying total ordering, under appropriate notions of distance (e.g., Kendall-Tau); MLE can be approached, e.g., via EM [Lu and Boutilier 2011]. Shah et al. [2016b] learn the full matrix of pairwise comparison probabilities via a minimax optimal estimator. Rajkumar and Agarwal [2016] learn the matrix via matrix completion, requiring $\Theta(nr \log n)$ comparisons, where $r \ll n$ is the rank. Ammar and Shah [2011] assume that comparisons are sampled from an unknown distribution over total orderings and propose an entropy maximization algorithm requiring $\Theta(n^2)$ comparisons.

Parametric Rank Aggregation. We focus on parametric models, as they are more natural in the context of regressing rankings from sample features. In both Plackett-Luce [Plackett 1975] and Thurstone [Thurstone 1927] each sample is parametrized by a score. In the Thurstone model, observations result from comparing scores after the addition of Gaussian noise. Vojnovic and Yun [2016] and Shah et al. [2016a] estimate Thurstone scores via MLE and provide sample complexity bounds that are inversely proportional to the smallest non-zero eigenvalue of the Laplacian of a graph modeling comparisons. In Plackett-Luce, the probability that a sample is chosen over a set of alternatives is proportional to its score. Hunter [2004] proposes a Minorization-Maximization (MM) approach to estimate Plackett-Luce scores via MLE, earlier used by Dykstra [1960] on pairwise comparisons (i.e., on the Bradley-Terry (BT) setting). Hajek et al. [2014] provide an upper bound on the error in estimating the Plackett-Luce scores via MLE and show that the latter is minimax-optimal. Negahban et al. [2018] propose a latent factor model, estimating parameters via a convex relaxation of the corresponding rank penalty and providing sample complexity guarantees. Vojnovic et al. [2020] show that gradient-descent and MM algorithms that solve the MLE problem converge with linear rate.

Plackett-Luce Inference via Spectral Methods. Our focus on Plackett-Luce is due to the recent emergence of spectral algorithms for inference in this setting. Negahban et al. [2012] propose the Rank Centrality (RC) algorithm for the BT setting and derive a minimax error bound. Chen and Suh [2015] propose a spectral MLE algorithm extending RC with an additional stage that cyclically performs MLE for each score. Soufiani et al. [2013] and Jang et al. [2017] extend RC to rankings of two or more samples by breaking rankings into independent comparisons. Improved bounds, applying also to broader noise settings, are provided by Rajkumar and Agarwal [2014]. Khetan and Oh [2016] generalize the work by Soufiani et al. [2013] by breaking rankings into independent shorter rankings, and building a hierarchy of tractable and consistent estimators. Blanchet et al. [2016] model sequential choices by state transitions in a Markov chain (MC), where transitions are functions of choice probabilities.

Bridging the above approaches with MLE, Maystre and Grossglauser [2015] show that the MLE of Plackett-Luce scores can be expressed as the stationary distribution of an MC. Their proposed Iterative Luce Spectral Ranking (ILSR) algorithm estimates the Plackett-Luce scores faster than traditional optimization methods, such as, e.g., Hunter's [Hunter 2004] and Newton's method [Nocedal and Wright 2006], and more accurately than prior spectral rank aggregation methods. More recently, Ragain and Ugander [2016] showed that a spectral approach applies even after relaxing the assumption that the relative order of any two samples is independent of the alternatives. Agarwal et al. [2018] proposed another spectral method called accelerated spectral ranking by departing from the exact equivalence between MLE and MC approximation and demonstrate faster convergence than ILSR.

Ranking Regression via Plackett-Luce. We depart from all aforementioned methods by regressing ranked choices from sample features. Ranking regression via the Plackett-Luce (PL) [Plackett 1975] model has a long history, mostly focusing on pairwise comparisons, i.e., restricted to $K = 2$; this case is also known as the Bradley-Terry (BT) model [Bradley and Terry 1952]. The resulting algorithms have been employed in a broad array of domains including medicine [Tian et al. 2019], information retrieval on search engines [Burges et al. 2005; Joachims 2002], image-based aesthetic recommendations [Chang et al. 2016; Sun et al. 2017], click prediction for online advertisements [Sculley 2010], and image retrieval [Chen et al. 2015], to name a few.

Among shallow regressors, RankSVM [Joachims 2002] learns a target ranking from features via a linear Support Vector Machine (SVM), with constraints imposed by all possible comparisons. Pahikkala et al. [2009] propose a regularized least-squares based algorithm for learning to rank from comparisons. Several works [Guo et al. 2018; Joachims 2002; Pahikkala et al. 2009; Tian et al. 2019] regress comparisons via maximum-likelihood estimation (MLE) over BT [Bradley and Terry 1952] models.

Deeper models for regressing comparisons have also been extensively explored in the BT setting [Burges et al. 2005; Chang et al. 2016; Doughty et al. 2018; Dubey et al. 2016; Yıldız et al. 2019]. Burges et al. [2005] estimate comparisons via a fully connected neural network called RankNet, using the BT model to construct a cross-entropy loss. Several works [Chang et al. 2016; Doughty et al. 2018; Dubey et al. 2016; Yıldız et al. 2019] learn from comparisons via siamese networks [Bromley et al. 1994]. Chang et al. [2016] and Yıldız et al. [2019] learn from comparisons using MLE on the BT model. Dubey et al. [2016] predict image comparisons, via a loss function combining cross-entropy and hinge loss. Doughty et al. [2018] learn similarities and comparisons of videos via hinge loss. All of the above methods generalize to rankings under the PL model, but suffer from the complexity issues outlined in the introduction.

Cao et al. [2007]; Xia et al. [2008] and Ma et al. [2020] focus on learning from star/relevance ratings rather than rankings; they train a neural network called ListNet that treats ratings as Plackett-Luce scores. Particularly, given a dataset of n ratings, ListNet constructs all $\frac{n!}{(n-K)!}$ possible K-sized rankings; it is then trained by minimizing the cross entropy between the Plackett-Luce probabilities that a given K-ranking comprises the top samples among all n, with scores being (a) the predicted scores and (b) the ground-truth ratings, respectively. Hence, although ListNet solves a very different problem than ranking regression, it is related to the siamese methods mentioned above through its (exponential in K) objective.

ADMM Applications in DNN Training. When training deep neural networks, ADMM is traditionally used to enforce sparsity. One application is compressed sensing, in which a signal is recovered from undersampled measurements via a DNN transformation [Li et al. 2018; Liu et al. 2018; Ma et al. 2019; Sun et al. 2016; Yang et al. 2020]. The training objective combines a standard regression loss with a norm penalty on measurements, including, e.g., ℓ_1, ℓ_2, Frobenius, or nuclear

norm. Another application is model pruning [Ye et al. 2019, 2018; Zhao and Liao 2019]: ADMM is used to enforce weight sparsity by replacing norm penalties with (possibly non-convex) low-cardinality set constraints. Zhao et al. [2018] train a DNN classifier to recover samples distorted by additive adversarial noise, using an objective that combines a classification loss with an ℓ_0, ℓ_1, ℓ_2, or ℓ_∞ norm penalty.

Relationship to Previous Works. The present paper is a combination and extension of two of our earlier works [Yıldız et al. 2020, 2021], which dealt with shallow and deep model settings separately. We unify the key theoretical results of these works, and include all proofs omitted from these conference versions. We extend our experimental evaluations, and provide a more detailed technical preliminary.

2.2 Technical Preliminary

2.2.1 The Plackett-Luce Model. We introduce here the Plackett-Luce discriminative model [Plackett 1975] that we use in our analysis; before describing it for ranking observations, we first consider the simpler "maximal choice" setting. Consider a dataset of n samples, indexed by $i \in [n] \equiv \{1, \dots, n\}$. Every sample $i \in [n]$ has a corresponding d-dimensional feature vector $\boldsymbol{x}_i \in \mathbb{R}^d$. There exists an underlying total ordering of these n samples. A labeler of this dataset acts as a (possibly noisy) oracle revealing this total ordering: when presented with a query $A \subseteq [n]$, i.e., a set of alternatives, the noisy labeler chooses the *maximal* sample in A w.r.t. the underlying total ordering. Formally, our "labeled" dataset $\mathcal{D} = \{(c_\ell, A_\ell)\}_{\ell=1}^m$ consists of m *observations* (c_ℓ, A_ℓ), $\ell \in [m] \equiv \{1, ..., m\}$, where $A_\ell \subseteq [n]$ is the ℓ-th *query* submitted to the labeler and $c_\ell \in A_\ell$ is her respective ℓ-th *maximal choice* (i.e., the label). For every sample $i \in [n]$, we denote by $W_i = \{\ell \in [m] \mid i \in A_\ell, c_\ell = i\}$ the set of observations where sample $i \in [n]$ is chosen, and by $L_i = \{\ell \in [m] \mid i \in A_\ell, c_\ell \neq i\}$ the set of observations where sample $i \in [n]$ is not chosen.

The Plackett-Luce model [Plackett 1975] asserts that every sample $i \in [n]$ is associated with a non-negative deterministic score $\pi_i \in \mathbb{R}_+$. Given scores $\boldsymbol{\pi} = [\pi_i]_{i\in[n]} \in \mathbb{R}_+^n$, then (i) observations $(c_\ell, A_\ell), \ell \in [m]$ are independent, and (ii) for a query A_ℓ, the probability that sample $c_\ell \in A_\ell$ is chosen is:

$$\mathbf{P}(c_\ell \mid A_\ell, \boldsymbol{\pi}) = \pi_{c_\ell} / \textstyle\sum_{j\in A_\ell} \pi_j = \pi_\ell / \sum_{j\in A_\ell} \pi_j, \tag{1}$$

where, in the last equation, we abuse notation to write the score of the chosen sample as $\pi_\ell \equiv \pi_{c_\ell}$. Note that $\mathbf{P}(c_\ell \mid A_\ell, \boldsymbol{\pi}) = \mathbf{P}(c_\ell \mid A_\ell, s\boldsymbol{\pi})$, for all $s > 0$; thus, w.l.o.g., we assume (or enforce via rescaling) that Plackett-Luce scores satisfy $\mathbf{1}^\top \boldsymbol{\pi} = 1$, i.e., $\boldsymbol{\pi}$ is a distribution over $[n]$.

Ranking Observations. Plackett-Luce readily extends to datasets of (partial) *ranking observations*. In this setting, when presented with a query $A_\ell \subseteq [n]$ of $K = |A_\ell|$ samples, the labeler ranks the samples in A_ℓ into an ordered sequence $\alpha_1^\ell \succ \alpha_2^\ell \succ \cdots \succ \alpha_K^\ell$. Under the Plackett-Luce model, this ranking is expressed as $K-1$ maximal choice queries: α_1^ℓ over A_ℓ, α_2^ℓ over $A_\ell \setminus \{\alpha_1^\ell\}$, etc., so that:

$$\mathbf{P}(\alpha_1^\ell \succ \alpha_2^\ell \succ \cdots \succ \alpha_K^\ell \mid A_\ell, \boldsymbol{\pi}) = \textstyle\prod_{t=1}^{K-1} \left(\pi_{\alpha_t^\ell} / \sum_{s=t}^{K} \pi_{\alpha_s^\ell}\right). \tag{2}$$

The product form of (2) implies that a ranking observation in response to a query A_ℓ can be converted to $K-1$ *independent* maximal-choice observations (again, α_1^ℓ over A_ℓ, α_2^ℓ over $A_\ell \setminus \{\alpha_1^\ell\}$, and so forth), each governed by (1), that yield the same joint probability: $(\alpha_1^\ell \succ \cdots \succ \alpha_K^\ell)$ can be seen as the outcome of α_1^ℓ being chosen as the top within the query set A_ℓ, α_2^ℓ being the top among $A_\ell \setminus \{\alpha_1^\ell\}$, and so on. Maximum-Likelihood Estimation (MLE) over a dataset of ranking observations thus reduces to MLE over a dataset of maximal choice observations. For notational simplicity, we present our analysis over maximal-choice datasets, keeping the above reduction in mind.

Parameter Inference. Given the dataset of observations $\mathcal{D}$, MLE of the Plackett-Luce scores $\boldsymbol{\pi} \in \mathbb{R}_+^n$ amounts to minimizing the negative log-likelihood:

$$\min_{\boldsymbol{\pi} \in \mathbb{R}_+^n} \mathcal{L}(\mathcal{D} \mid \boldsymbol{\pi}) \equiv \sum_{\ell=1}^{m} \left(\log \sum_{j \in A_\ell} \pi_j - \log \pi_\ell \right). \tag{3}$$

Reparametrizing the scores as $\pi_i = e^{\theta_i}$, $i \in [n]$ makes the negative log-likelihood $\mathcal{L}$ convex in $\boldsymbol{\theta} = [\theta_i]_{i \in [n]}$. Under this transform, a global optimum of Eq. (3) can be computed via, e.g., Newton's method.

2.2.2 Iterative Luce Spectral Ranking (ILSR). As described in more detail in Section 2.2.1, MLE of Plackett-Luce scores is convex under the reparametrization $\pi_i = e^{\theta_i}$, $i \in [n]$, which in turn enables computing the Plackett-Luce scores via Newton's method. Nevertheless, Newton's method can be prohibitively slow for large n and m [Hunter 2004]. Maystre and Grossglauser [2015] proposed a novel spectral algorithm that is significantly faster than traditional optimization methods, such as, e.g., Hunter's [Hunter 2004] and Newton's method, and more accurate than prior spectral rank aggregation methods. They show that the MLE of Plackett-Luce scores can be expressed as the stationary distribution of an MC. Their algorithm relies on the following theorem which, for completeness, we re-prove in Appendix B:

Theorem 2.1 (Maystre and Grossglauser [2015]). *An optimal solution $\boldsymbol{\pi} \in \mathbb{R}_+^n$ to (3) satisfies:*

$$\sum_{j \neq i} \pi_j \lambda_{ji}(\boldsymbol{\pi}) = \sum_{j \neq i} \pi_i \lambda_{ij}(\boldsymbol{\pi}), \quad \text{for all } i \in [n], \tag{4}$$

where, for all $i, j \in [n]$, with $i \neq j$,

$$\lambda_{ji}(\boldsymbol{\pi}) = \sum_{\ell \in W_i \cap L_j} \left(\sum_{t \in A_\ell} \pi_t \right)^{-1} \geq 0, \tag{5}$$

for $W_i = \{\ell \mid i \in A_\ell, c_\ell = i\}$ the observations where sample $i \in [n]$ is chosen and $L_i = \{\ell \mid i \in A_\ell, c_\ell \neq i\}$ the observations where sample $i \in [n]$ is not chosen.

Eq. (4) are the *balance equations* of a continuous-time Markov Chain (MC) with transition rates:

$$\boldsymbol{\Lambda}(\boldsymbol{\pi}) = [\lambda_{ji}(\boldsymbol{\pi})]_{i,j \in [n]}, \tag{6}$$

where $\lambda_{ji}(\boldsymbol{\pi})$ are given by Eq. (5). Hence, $\boldsymbol{\pi}$ is the stationary distribution of the MC defined by transition rates $\boldsymbol{\Lambda}(\boldsymbol{\pi})$ (c.f. Appendix A).

Let $\texttt{ssd}(\boldsymbol{\Lambda})$ be the stationary distribution of an MC with transition rates $\boldsymbol{\Lambda}$. When matrix $\boldsymbol{\Lambda}$ is fixed (i.e., the transition rates are known), the vector $\texttt{ssd}(\boldsymbol{\Lambda})$ is a solution to the linear system defined by the balance equations (4) and $\mathbf{1}^\top \boldsymbol{\pi} = 1$, as it is a distribution. However, the transition matrix $\boldsymbol{\Lambda} = \boldsymbol{\Lambda}(\boldsymbol{\pi})$ in Theorem 2.1 *is itself a function of $\boldsymbol{\pi}$, and is therefore a priori unknown.* Maystre and Grossglauser [2015] find $\boldsymbol{\pi}$ through an iterative algorithm. Starting from the uniform distribution $\boldsymbol{\pi}^0 = \frac{1}{n}\mathbf{1}$, they compute:

$$\boldsymbol{\pi}^{l+1} = \texttt{ssd}\left(\boldsymbol{\Lambda}(\boldsymbol{\pi}^l)\right), \quad \text{for } l \in \mathbb{N}, \tag{7}$$

where $\boldsymbol{\Lambda}(\cdot)$ is given by (5), (6).

Maystre and Grossglauser [2015] refer to Eq. (7) as the *Iterative Luce Spectral Ranking* (ILSR) algorithm. They also establish that (7) converges to an optimal solution for MLE of Plackett-Luce scores under mild assumptions. Most importantly, as mentioned above, ILSR significantly outperforms state-of-the-art MLE algorithms in computational efficiency. We note that the dataset $\mathcal{D}$ appears in the computation of the rate matrix $\boldsymbol{\Lambda}$, via sets W_i and L_j; the computation of these weights can be easily parallelized.

2.2.3 Alternating Directions Method of Multipliers (ADMM). Virtually all shallow and deep models for ranking regression rely on classic optimization methods for direct parameter inference, resulting in prohibitively slow learning for large rankings. We solve the ranking regression problem via the Alternating Directions Method of Multipliers (ADMM) [Boyd et al. 2011] and attain spectral algorithms that significantly accelerate learning.

ADMM is a primal-dual algorithm designed for affine-constrained optimization problems with decoupled objectives, i.e., objectives that can be written as a sum of functions where each function depends on only one of the optimized variables. ADMM combines the traditional dual decomposition [Nocedal and Wright 2006] and augmented Lagrangian [Nocedal and Wright 2006] optimization algorithms; it has also been shown to be a special case of the Douglas-Rachford splitting [Nocedal and Wright 2006] algorithm. ADMM became particularly popular for being well-suited for distributed convex optimization, a problem that commonly arises in applied statistics and machine learning [Boyd et al. 2011]. We follow the notation by Boyd et al. [2011] in our exposition below.

Standard ADMM. Formally, the ADMM algorithm solves problems in the form:

$$\begin{aligned} &\text{minimize} && f(\boldsymbol{x}) + g(\boldsymbol{z}) \\ &\text{subject to} && \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z} = \boldsymbol{c}, \end{aligned} \tag{8}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ and $\boldsymbol{z} \in \mathbb{R}^m$ are the optimized variables, with $\boldsymbol{A} \in \mathbb{R}^{p \times n}$, $\boldsymbol{B} \in \mathbb{R}^{p \times m}$, and $\boldsymbol{c} \in \mathbb{R}^p$. f and g are typically assumed to be convex functions that depend on decoupled variables $\boldsymbol{x}$ and $\boldsymbol{z}$, respectively.

ADMM solves a constrained optimization problem by minimizing the *augmented Lagrangian* instantiated below, rather than the standard Lagrangian:

$$L_\rho(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y}) = f(\boldsymbol{x}) + g(\boldsymbol{z}) + \boldsymbol{y}^\top (\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z} - \boldsymbol{c}) + \frac{\rho}{2} \parallel \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z} - \boldsymbol{c} \parallel_2^2, \tag{9}$$

where $\boldsymbol{y} \in \mathbb{R}^p$ is the Lagrangian dual variable corresponding to the equality constraint and $\rho \geq 0$ is a penalty parameter. Evidently, the difference of augmented Lagrangian from the standard Lagrangian is the additional quadratic *proximal* penalty on the equality constraint. This additional penalty is shown to greatly improve convergence properties of the algorithm [Boyd et al. 2011], compared to, e.g. dual ascent.

ADMM iteratively optimizes the augmented Lagrangian w.r.t. the primal variables $\boldsymbol{x}$ and $\boldsymbol{z}$, and dual variables $\boldsymbol{y}$ as follows:

$$\begin{aligned} \boldsymbol{x}^{k+1} &= \arg\min_{\boldsymbol{x}} L_\rho(\boldsymbol{x}, \boldsymbol{z}^k, \boldsymbol{y}^k), \\ \boldsymbol{z}^{k+1} &= \arg\min_{\boldsymbol{z}} L_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{z}, \boldsymbol{y}^k), \\ \boldsymbol{y}^{k+1} &= \boldsymbol{y}^k + \rho(\boldsymbol{A}\boldsymbol{x}^{k+1} + \boldsymbol{B}\boldsymbol{z}^{k+1} - \boldsymbol{c}). \end{aligned} \tag{10}$$

For convex problems, there are well-established convergence properties for ADMM. If f and g are closed, proper, and convex, and the augmented Lagrangian without the quadratic penalty has a saddle point, the ADMM iterations are guaranteed to converge to a point where (a) the equality constraint is satisfied, and (b) objectives and dual variable attain optimal values. Even though the convergence has linear rate, in many applications, ADMM has been shown to converge to a modest accuracy in a few tens of iterations [Boyd et al. 2011]. ADMM has also been extensively used to solve non-convex problems similar to the one we study here [Chartrand and Wohlberg 2013; Guo et al. 2017; Hong 2018; Wang et al. 2019]. Focusing on specific instances of non-convex problems, Guo et al. [2017]; Magnússon et al. [2015] and Hong [2018] establish that ADMM converges to a stationary point of the augmented Lagrangian. In general, ADMM is not guaranteed to converge for non-convex problems, and even if it does, it may not converge a global optimum.

Generalization to Bregman Divergence Proximal Penalty. ADMM has extensions where the quadratic proximal penalty is generalized to, e.g., Bregman divergences. Wang and Banerjee [2014] employ ADMM with a Bregman divergence proximal penalty, where KL is a special case, to solve a problem of the form (8). They minimize the augmented Lagrangian:

$$L_\rho(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y}) = f(\boldsymbol{x}) + g(\boldsymbol{z}) + \boldsymbol{y}^\top(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z} - \boldsymbol{c}) + \rho D_p(\boldsymbol{B}\boldsymbol{z}, \boldsymbol{c} - \boldsymbol{A}\boldsymbol{x}), \tag{11}$$

where D_p is a Bregman divergence [Bregman 1967]. A Bregman divergence is associated with a stricly-convex and contionously-differentiable function $h : \Omega \to \mathbb{R}$, where Ω is a closed convex set. Given such a function h and any two points $x, y \in \Omega$, the Bregman divergence is the difference between the value of h at point x and the value of the first-order Taylor expansion of h around point y evaluated at point x [Bregman 1967]:

$$D_p(x, y) = h(x) - h(y) - \nabla h(y)^\top (x - y). \tag{12}$$

Having adjusted the augmented Lagrangian as (11), Bregman-ADMM again iteratively optimizes the augmented Lagrangian w.r.t. the primal variables $\boldsymbol{x}$ and $\boldsymbol{z}$, and dual variables $\boldsymbol{y}$ via (10). Wang and Banerjee [2014] show that ADMM with a Bregman divergence proximal penalty converges with linear rate for convex objectives. Wang et al. [2018] extend this convergence guarantee to local optima of non-convex problems. Several works [Shi et al. 2017; Yu and Açıkmeşe 2019; Yu et al. 2018] employ a Bregman ADMM algorithm for distributed optimization of separable problems.

3 SPECTRAL RANKING REGRESSION

3.1 Problem Formulation: Ranking Regression

As introduced in Section 2.2.1, MLE of the Plackett-Luce scores $\boldsymbol{\pi} \in \mathbb{R}^n_+$ amounts to minimizing the negative log-likelihood:

$$\min_{\boldsymbol{\pi} \in \mathbb{R}^n_+} \mathcal{L}(\mathcal{D} \mid \boldsymbol{\pi}) \equiv \sum_{\ell=1}^m \left(\log \sum_{j \in A_\ell} \pi_j - \log\, \pi_\ell \right). \tag{13}$$

To regress scores $\boldsymbol{\pi}$ from sample features $\boldsymbol{X} = [\boldsymbol{x}_1, .., \boldsymbol{x}_n]^T \in \mathbb{R}^{n \times d}$, we assume that there exists a function $\tilde{\pi} : \mathbb{R}^d \to [0, 1]$, parametrized by $\boldsymbol{w} \in \mathbb{R}^{d'}$, such that:

$$\pi_i = \tilde{\pi}(\boldsymbol{x}_i; \boldsymbol{w}), \quad \text{for all } i \in [n]\,. \tag{14}$$

Then, MLE amounts to minimizing the following objective w.r.t. $\boldsymbol{w} \in \mathbb{R}^{d'}$:

$$\mathcal{L}\big(\mathcal{D} \mid \boldsymbol{w}\big) \equiv \sum_{\ell=1}^m \Big(\log \sum_{j \in A_\ell} \tilde{\pi}(\boldsymbol{x}_j; \boldsymbol{w}) - \log \tilde{\pi}(\boldsymbol{x}_\ell; \boldsymbol{w}) \Big). \tag{15}$$

We tackle the regression problem via the MLE objective (15) w.r.t. three regression functions. Our notation is summarized in Table 1.

Affine Regression. We assume that there exist $\boldsymbol{a} \in \mathbb{R}^d$ and $b \in \mathbb{R}$, such that $\boldsymbol{\pi} = \tilde{\pi}(\boldsymbol{X}; \boldsymbol{w}) \equiv \boldsymbol{X}\boldsymbol{a} + \boldsymbol{1}b$ for $\boldsymbol{w} = (\boldsymbol{a}, b) \in \mathbb{R}^{d+1}$. Then, MLE of parameters $\boldsymbol{w}$ amounts to solving:

$$\min_{\boldsymbol{w}=(\boldsymbol{a},b) \in \mathbb{R}^{d+1} : \tilde{\pi}(\boldsymbol{X}; \boldsymbol{w}) \geq \boldsymbol{0}} \mathcal{L}\left(\mathcal{D} \mid \tilde{\pi}(\boldsymbol{X}; \boldsymbol{w}) \equiv \boldsymbol{X}\boldsymbol{a} + \boldsymbol{1}b\right), \tag{16}$$

where $\mathcal{L}$ is given by (15). We note that Problem (16) is *not* convex, as the objective is not convex in $(\boldsymbol{a}, b) \in \mathbb{R}^{d+1}$.

Logistic Regression. In the logistic case, we assume that there exists $\boldsymbol{w} \in \mathbb{R}^d$ s.t. $\boldsymbol{\pi} = \tilde{\pi}(\boldsymbol{X}; \boldsymbol{w}) \equiv [e^{\boldsymbol{w}^\top \boldsymbol{x}_i}]_{i \in [n]}$. As $\tilde{\pi}(\boldsymbol{X}; \boldsymbol{w}) \geq 0$ by definition, MLE corresponds to:

$$\min_{\boldsymbol{w} \in \mathbb{R}^d} \mathcal{L}\left(\mathcal{D} \mid \tilde{\pi}(\boldsymbol{X}; \boldsymbol{w}) \equiv [e^{\boldsymbol{w}^\top \boldsymbol{x}_i}]_{i \in [n]}\right). \tag{17}$$

Table 1. Notation Summary

n	number of samples
m	number of choice / ranking observations
d	number of features of each sample
K	number of samples in each ranking query
$\boldsymbol{w} \in \mathbb{R}^{d'}$	regression model parameters
$\boldsymbol{x}_i \in \mathbb{R}^d$	feature vector of sample i
$\boldsymbol{X} \in \mathbb{R}^{n \times d}$	matrix of feature vectors of all samples
$\tilde{\boldsymbol{X}} = [\boldsymbol{X}\|\boldsymbol{1}] \in \mathbb{R}^{n \times (d+1)}$	extended matrix of feature vectors of all samples
c_ℓ	maximal choice for the ℓ-th observation
α_ℓ	ranking of samples in the ℓ-th observation
A_ℓ	set of alternative samples, i.e. , query set of the ℓ-th observation
$[n]$	set of all samples
$[m]$	set of all choice/ranking observations
$\mathcal{D}$	dataset of all query-observation pairs
W_i	set of observations in which sample i is chosen
L_i	set of observations in which sample i is not chosen
$\boldsymbol{\pi} \in \mathbb{R}^n_+$	scores of all samples
$\tilde{\boldsymbol{\pi}}(\cdot, \boldsymbol{w}) \in \mathbb{R}^n_+$	regression function of all scores
D_p	proximal penalty function of ADMM
$\mathcal{L}$	negative log-likelihood of the Plackett-Luce model
L	augmented Lagrangian of ADMM
$\rho > 0$	penalty parameter of ADMM
$\boldsymbol{y} \in \mathbb{R}^n$	dual variables of ADMM
$\lambda_{ji},\ i, j \in [n]$	transition rates of the ILSR Markov Chain (c.f. (5))
$\boldsymbol{\Lambda}$	transition matrix of the ILSR Markov Chain (c.f. (6))
$\mu_{ji},\ i, j \in [n]$	transition rates of the Markov Chain (c.f. (27))
$\boldsymbol{M}$	transition matrix of the Markov Chain (c.f. (28))
$\boldsymbol{\sigma} \in \mathbb{R}^n$	additional terms in the Markov Chain rates (c.f. Theorem 3.2)
$([n]_-, [n]_+)$	partition of $[n]$ s.t. $\sigma_i \geq 0$ for $i \in [n]_+$ and $\sigma_i < 0$ for $i \in [n]_-$
ssd	stationary distribution of a Markov Chain

The objective of (17) is convex in $\boldsymbol{w}$, so an optimal solution can be found via, e.g., Newton's method [Nocedal and Wright 2006]. The convexity of Problem (17) w.r.t. $\boldsymbol{w}$ follows by the facts that (a) the reparametrization $\pi_i = e^{\theta_i}$, $i \in [n]$ makes $\mathcal{L}$ convex, and (b) the composition of convex and affine is convex [Boyd and Vandenberghe 2004].

Deep Neural Network (DNN) Regression. Finally, we regress scores $\boldsymbol{\pi}$ via a generic DNN function $\tilde{\pi}(\boldsymbol{X}; \boldsymbol{w})$ by solving the following MLE problem:

$$\min_{\boldsymbol{w} \in \mathbb{R}^{d'}} \mathcal{L}(\mathcal{D} \mid \tilde{\boldsymbol{\pi}}(\boldsymbol{X}; \boldsymbol{w})), \tag{18}$$

where $\tilde{\boldsymbol{\pi}}(\boldsymbol{X}; \boldsymbol{w}) \in \mathbb{R}^n_+$ is the vector map whose coefficients $\tilde{\pi}_i \in \mathbb{R}_+$ are given by $\tilde{\pi}_i = \tilde{\pi}(\boldsymbol{x}_i; \boldsymbol{w})$, i.e., $\tilde{\boldsymbol{\pi}}(\boldsymbol{X}; \boldsymbol{w}) \equiv [\tilde{\pi}(\boldsymbol{x}_i; \boldsymbol{w})]_{i \in [n]}$. For instance, $\tilde{\pi}(\cdot, \boldsymbol{w})$ can be a fully-connected feed-forward neural network if $\boldsymbol{X}$ contains numerical features, or a convolutional neural network if $\boldsymbol{X}$ is an image.

As also discussed in Section 1, virtually all state-of-the-art ranking regression methods minimize Eq. (15) via classic gradient-based optimization [Burges et al. 2005; Chang et al. 2016; Doughty et al. 2018; Dubey et al. 2016; Guo et al. 2018; Joachims 2002; Pahikkala et al. 2009; Tian et al. 2019;

Yıldız et al. 2019]. Note that the optimization of objective (15) implies the following process: for each observation (c_ℓ, A_ℓ) in $\mathcal{D}$, the regression function $\tilde{\boldsymbol{\pi}}(\boldsymbol{X};\boldsymbol{w})$ needs to be evaluated and updated over *all samples in* A_ℓ. Overall, the ranking regression objective (15) implies that K samples need to be loaded in RAM to process a single ranking or maximal choice query. To make matters worse, the number m of possible rankings/maximal queries in $\mathcal{D}$ grows as $\binom{n}{K} = O(n^K)$. This means that the length of a training epoch of stochastic gradient descent (and hence, the number of expensive gradient computations) is $m = O(n^K)$, i.e., grows exponentially in K. Combined, these two effects can make the cost of a single optimization step over Eq. (15) prohibitive (c.f. Figures 1 and 5c). This motivates us to explore efficient spectral algorithms to solve regression problems (16)–(18), as we discuss next.

3.2 Key Technical Result: Decoupling Optimization and a Spectral Method

Given ILSR's significant computational benefits in Section 2, we wish to develop analogues in the regression setting. In contrast to the feature-less setting, it is not a priori evident how to solve Problems (16) - (18) via a spectral approach. Taking the affine case as an example, and momentarily ignoring issues of non-convexity, the stationary points of the optimization problem (16) *cannot be expressed via the balance equations of an MC*. Our main contribution is to circumvent this problem by using the Alternating Directions Method of Multipliers (ADMM) [Boyd et al. 2011]. Intuitively, ADMM allows us to decouple the optimization of scores $\boldsymbol{\pi}$ from model parameters $\boldsymbol{w}$, encapsulating them in a proximal penalty: the latter becomes amenable to a spectral approach after a series of manipulations that we outline below (see Theorem 3.2).

We wish to devise an efficient algorithm to minimize the regression objective (15). To do so, we extend the standard ADMM [Boyd et al. 2011] introduced in Section 2. To this end, we rewrite the minimization of (15) as:

$$\underset{\boldsymbol{\pi},\boldsymbol{w}}{\text{Minimize}} \quad \mathcal{L}(\mathcal{D} \mid \boldsymbol{\pi}) \equiv \sum_{\ell=1}^{m} \left(\log \sum_{j \in A_\ell} \pi_j - \log\, \pi_\ell \right) \tag{19a}$$

$$\text{subject to:} \quad \boldsymbol{\pi} = \tilde{\boldsymbol{\pi}}(\boldsymbol{X};\boldsymbol{w}), \quad \boldsymbol{\pi} \geq \mathbf{0}. \tag{19b}$$

To solve (19) via ADMM, consider the following *augmented Lagrangian* (see Section 2):

$$L_\rho(\boldsymbol{\pi},\boldsymbol{w},\boldsymbol{y}) = \mathcal{L}(\mathcal{D} \mid \boldsymbol{\pi}) + \boldsymbol{y}^\top(\boldsymbol{\pi} - \tilde{\boldsymbol{\pi}}(\boldsymbol{X};\boldsymbol{w})) + \rho \cdot D_p(\boldsymbol{\pi}||\tilde{\boldsymbol{\pi}}(\boldsymbol{X};\boldsymbol{w})), \tag{20}$$

where $\boldsymbol{y} \in \mathbb{R}^n$ is the Lagrangian dual variable corresponding to the equality constraint in (19b), $\rho \geq 0$ is a penalty parameter, and $D_p(\cdot||\cdot)$ is a proximal penalty term. This term satisfies: (i) $D_p(\boldsymbol{\pi}||\tilde{\boldsymbol{\pi}}) \geq 0$ for all $\boldsymbol{\pi}, \tilde{\boldsymbol{\pi}} \in \mathbb{R}^n_+$, and (ii) $D_p(\boldsymbol{\pi}||\tilde{\boldsymbol{\pi}}) = 0$ if and only if $\tilde{\boldsymbol{\pi}} = \boldsymbol{\pi}$. Classic ADMM involves using the usual ℓ_2 proximal penalty, i.e., $D_p(\boldsymbol{\pi}||\tilde{\boldsymbol{\pi}}) = \frac{1}{2} \| \boldsymbol{\pi} - \tilde{\boldsymbol{\pi}} \|_2^2$. We depart from this by considering more generic D_p; as we discuss in Section 3.5, using Kullback-Leibler (KL) divergence instead, i.e., $D_p(\boldsymbol{\pi}||\tilde{\boldsymbol{\pi}}) = \sum_{i=1}^n \pi_i \log \frac{\pi_i}{\tilde{\pi}_i}$ has significant advantages in our setting.

In its general form, ADMM alternates between optimizing $\boldsymbol{\pi}$ and $\boldsymbol{w}$ until convergence via the following primal-dual algorithm on the augmented Lagrangian (c.f. Section 2):

$$\boldsymbol{\pi}^{k+1} = \underset{\boldsymbol{\pi} \in \mathbb{R}^n_+}{\arg\min}\ L_\rho(\boldsymbol{\pi}, \boldsymbol{w}^k, \boldsymbol{y}^k) \tag{21a}$$

$$\boldsymbol{w}^{k+1} = \underset{\boldsymbol{w} \in \mathbb{R}^{d'}}{\arg\min}\ L_\rho(\boldsymbol{\pi}^{k+1}, \boldsymbol{w}, \boldsymbol{y}^k) \tag{21b}$$

$$\boldsymbol{y}^{k+1} = \boldsymbol{y}^k + \rho(\boldsymbol{\pi}^{k+1} - \tilde{\boldsymbol{\pi}}(\boldsymbol{X};\boldsymbol{w}^{k+1})). \tag{21c}$$

This has the following immediate computational advantages. First, step (21b) is equivalent to:

$$\boldsymbol{w}^{k+1} = \underset{\boldsymbol{w}\in\mathbb{R}^{d'}}{\arg\min}\ \rho D_p\big(\boldsymbol{\pi}^{k+1}||\tilde{\boldsymbol{\pi}}(\boldsymbol{X};\boldsymbol{w})\big) - \boldsymbol{y}^{k\top}\tilde{\boldsymbol{\pi}}(\boldsymbol{X};\boldsymbol{w}). \tag{22}$$

Note that this operation *does not* depend on dataset $\mathcal{D}$: the model $\tilde{\boldsymbol{\pi}}(\boldsymbol{X};\boldsymbol{w})$ is regressed directly from the present score estimates $\boldsymbol{\pi}^{k+1}$ via penalty D_p, with the additional linear dual term. In particular, as discussed below, D_p leads to a least squares regression in the case of the ℓ_2-proximal penalty, and a max-entropy penalty in the case of KL-divergence; both can be solved efficiently. Crucially, an optimization method that solves (22) iterates over the $O(n)$ samples, instead of the $O(n^K)$ ranking observations, which would be the case when Eq. (15) is minimized directly without ADMM decoupling.

Most importantly, step (21a)–which *does* depend on $\mathcal{D}$–admits a highly efficient spectral implementation for a proximal penalty D_p; ADMM (21) therefore delegates solving the "expensive" portion of the problem to a highly efficient algorithm. To establish this, we begin with the following lemma, which we prove in Appendix C.

Lemma 3.1. *Given $\tilde{\boldsymbol{\pi}}^k \equiv \tilde{\boldsymbol{\pi}}(\boldsymbol{X};\boldsymbol{w}^k) \in \mathbb{R}^n_+$ and $\boldsymbol{y}^k \in \mathbb{R}^n$, let $\boldsymbol{\pi} \in \mathbb{R}^n_+$ be such that:*

$$\nabla_{\boldsymbol{\pi}} L_\rho(\boldsymbol{\pi}, \boldsymbol{w}^k, \boldsymbol{y}^k) = \boldsymbol{0}. \tag{23}$$

Let $\boldsymbol{\sigma}(\boldsymbol{\pi}) = [\sigma_i(\boldsymbol{\pi})]_{i\in[n]} \in \mathbb{R}^n$, where

$$\sigma_i(\boldsymbol{\pi}) \equiv \rho\frac{\partial D_p(\boldsymbol{\pi}||\tilde{\boldsymbol{\pi}}^k)}{\partial \pi_i} + y_i^k, \quad \text{for } i \in [n]\,, \tag{24}$$

and

$$\lambda_{ji}(\boldsymbol{\pi}) \equiv \textstyle\sum_{\ell\in W_i\cap L_j}\left(\sum_{t\in A_\ell}\pi_t\right)^{-1} \geq 0, \quad \text{for } i,j \in [n] \ \text{ with } i \neq j, \tag{25}$$

with $W_i = \{\ell\,|i \in A_\ell, c_\ell = i\}$ denoting the observations where sample $i \in [n]$ is chosen and $L_i = \{\ell\,|i \in A_\ell, c_\ell \neq i\}$ denoting the observations where sample $i \in [n]$ is not chosen. Then, (23) *is equivalent to:*

$$\textstyle\sum_{j\neq i}\pi_j\lambda_{ji}(\boldsymbol{\pi}) - \sum_{j\neq i}\pi_i\lambda_{ij}(\boldsymbol{\pi}) = \pi_i\sigma_i(\boldsymbol{\pi}), \quad \text{for all } i \in [n]\,. \tag{26}$$

Contrasting Eq. (26) to Eq. (4), it is not immediately evident that the former corresponds to the balance equations of an MC as, in general, $\boldsymbol{\sigma}(\boldsymbol{\pi}) = [\sigma_i(\boldsymbol{\pi})]_{i\in[n]} \neq \boldsymbol{0}$. Nevertheless, for an appropriate definition of transition rates, we prove that this is indeed the case:

Theorem 3.2. *Equations* (26) *are the balance equations of a continuous-time Markov Chain (MC) with transition rates:*

$$\mu_{ji}(\boldsymbol{\pi}) = \begin{cases} \lambda_{ji}(\boldsymbol{\pi}) + \frac{2\pi_i\sigma_i(\boldsymbol{\pi})\sigma_j(\boldsymbol{\pi})}{\sum_{t\in[n]_-}\pi_t\sigma_t(\boldsymbol{\pi}) - \sum_{t\in[n]_+}\pi_t\sigma_t(\boldsymbol{\pi})}, & \text{if } j \in [n]_+ \ \text{ and } \ i \in [n]_-\,, \\ \lambda_{ji}(\boldsymbol{\pi}) & \text{otherwise,} \end{cases} \tag{27}$$

where $([n]_+, [n]_-)$ denotes a partition of $[n]$ such that $\sigma_i(\boldsymbol{\pi}) \geq 0$ for all $i \in [n]_+$ and $\sigma_i(\boldsymbol{\pi}) < 0$ for all $i \in [n]_-$.

The proof is in Appendix D. By Lemma 3.1 and Theorem 3.2, we conclude that a stationary $\boldsymbol{\pi} \in \mathbb{R}^n_+$ satisfying (23) is also the stationary distribution of the continuous-time MC with transition rates:

$$\boldsymbol{M}(\boldsymbol{\pi}) = [\mu_{ji}(\boldsymbol{\pi})]_{i,j\in[n]}, \tag{28}$$

where $\mu_{ji}(\boldsymbol{\pi})$ are given by Eq. (27). Motivated by these observations, and mirroring ILSR (Eq. (7)), we compute a solution to (21a) via:

$$\boldsymbol{\pi}^{l+1} = \mathsf{ssd}\left(\boldsymbol{M}(\boldsymbol{\pi}^l)\right), \quad \text{for } l \in \mathbb{N}, \tag{29}$$

Algorithm 1 Spectral Ranking

procedure ADMM($\boldsymbol{X}, \mathcal{D}, \rho$)
  Initialize $\boldsymbol{\pi} = \tilde{\boldsymbol{\pi}} \leftarrow \frac{1}{n}\mathbf{1}; \boldsymbol{y} \leftarrow \mathbf{0}$
  repeat
    $\boldsymbol{\pi} \leftarrow$ ILSRX($\mathcal{D}, \rho, \boldsymbol{\pi}, \tilde{\boldsymbol{\pi}}, \boldsymbol{y}$)
    $\boldsymbol{w} \leftarrow$ Solution to Eq. (22)
    $\tilde{\boldsymbol{\pi}} \leftarrow \tilde{\boldsymbol{\pi}}(\boldsymbol{X}; \boldsymbol{w})$
    $\boldsymbol{y} \leftarrow \boldsymbol{y} + \rho(\boldsymbol{\pi} - \tilde{\boldsymbol{\pi}})$
  until convergence
return $\boldsymbol{w}$, $\boldsymbol{\pi}$
end procedure

procedure ILSRX($\mathcal{D}, \rho, \boldsymbol{\pi}, \tilde{\boldsymbol{\pi}}, \boldsymbol{y}$)
  repeat
    Calculate $[\lambda_{ji}(\boldsymbol{\pi})]_{i,j\in[n]}$ via Eq. (25)
    Calculate $[\sigma_i(\boldsymbol{\pi})]_{i\in[n]}$ via Eq. (24)
    Calculate $\boldsymbol{M}(\boldsymbol{\pi}) = [\mu_{ji}(\boldsymbol{\pi})]_{i,j\in[n]}$ via Eq. (27)
    $\boldsymbol{\pi} \leftarrow \mathtt{ssd}(\boldsymbol{M}(\boldsymbol{\pi}))$
  until convergence
return $\boldsymbol{\pi}$
end procedure

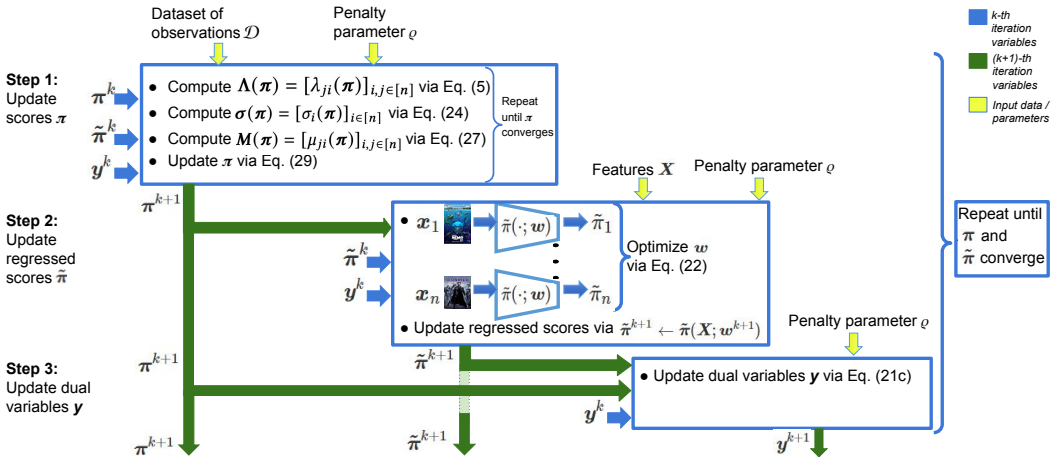


Fig. 2. Spectral Ranking Algorithm Diagram. We iteratively update $\boldsymbol{\pi}, \boldsymbol{w}$, and $\boldsymbol{y}$ via Eq. (21) until convergence. Each block corresponds to one of the three ADMM steps in Eq. (21). External inputs are provided on the left, in yellow, and optimized variables from the previous iteration or step are provided on the top. We indicate the variables provided by the previous (k-th) iteration in blue and the updated variables (i.e., the output at the $(k+1)$-th iteration) in green.

where $\boldsymbol{M}(\cdot)$ is given by Eq. (28) and $\mathtt{ssd}(\cdot)$ is an algorithm computing the steady state distribution. We refer to this procedure as ILSRX ("ILSR with features").

Our spectral algorithm solving Eq. (19) is summarized in Algorithm 1 and Figure 2. We initialize all samples with equal scores, i.e., $\boldsymbol{\pi} = \tilde{\boldsymbol{\pi}} = \frac{1}{n}\mathbf{1}$, and set the initial dual variable as $\boldsymbol{y}^0 = \mathbf{0}$. We iteratively update $\boldsymbol{\pi}, \boldsymbol{w}$, and $\boldsymbol{y}$ via Eq. (21) until convergence. At each ADMM iteration $k+1$, we

initialize $\boldsymbol{\pi}$ with $\boldsymbol{\pi}^k$, and update $\boldsymbol{\pi}$ via ILSRX given by Eq. (29). We elaborate on the resulting implementation specifics for each of the regression problems (16)–(18) below.

3.3 Spectral Algorithm for Affine Regression

In this section, we apply Algorithm 1 on affine regression of Plackett-Luce scores via:

$$\tilde{\boldsymbol{\pi}}(\boldsymbol{X};\boldsymbol{w}) = \boldsymbol{X}\boldsymbol{a} + \mathbf{1}b = \tilde{\boldsymbol{X}}\boldsymbol{w}; \tag{30}$$

we introduce $\boldsymbol{w} = (\boldsymbol{a}, b) \in \mathbb{R}^{d+1}$ and $\tilde{\boldsymbol{X}} = [\boldsymbol{X}|\mathbf{1}] \in \mathbb{R}^{n\times(d+1)}$ for notational simplicity. Recall from Section 3.1 that the resulting MLE problem is given by:

$$\min_{\boldsymbol{w}\in\mathbb{R}^{d+1}:\tilde{\pi}(\boldsymbol{X};\boldsymbol{w})\geq\mathbf{0}} \mathcal{L}\left(\mathcal{D} \mid \tilde{\pi}(\boldsymbol{X};\boldsymbol{w}) \equiv \tilde{\boldsymbol{X}}\boldsymbol{w}\right). \tag{31}$$

In this setting, we employ standard ADMM with quadratic proximal penalty:

$$D_p(\boldsymbol{\pi}||\tilde{\boldsymbol{\pi}}) = \frac{1}{2} \parallel \boldsymbol{\pi} - \tilde{\boldsymbol{\pi}} \parallel_2^2 . \tag{32}$$

This leads to additional theoretical guarantees explained in Section 3.7. Employing Algorithm 1 with quadratic $D_p(\boldsymbol{\pi}||\tilde{\boldsymbol{\pi}})$ and affine $\tilde{\boldsymbol{\pi}}(\boldsymbol{X};\boldsymbol{w})$, the regression step (21b) of ADMM becomes:

$$\begin{aligned}\boldsymbol{w}^{k+1} &= \arg\min_{\boldsymbol{w}\in\mathbb{R}^{d+1}} \boldsymbol{y}^{k^T}(\boldsymbol{\pi}^{k+1} - \tilde{\boldsymbol{X}}\boldsymbol{w}) + \frac{\rho}{2} \parallel \boldsymbol{\pi}^{k+1} - \tilde{\boldsymbol{X}}\boldsymbol{w} \parallel_2^2 \\ &= (\tilde{\boldsymbol{X}}^T\tilde{\boldsymbol{X}})^{-1}\tilde{\boldsymbol{X}}^T(\boldsymbol{\pi}^{k+1} + \frac{1}{\rho}\boldsymbol{y}^k).\end{aligned} \tag{33}$$

Eq. (33) follows from completing the square by combining the linear and quadratic terms into a single quadratic objective. This has the following immediate computational advantage: the regression step (33) is a quadratic minimization and admits a closed form solution.

Crucially, the score update step (21a) is amenable to the spectral solution established in Theorem 3.2, where:

$$\boldsymbol{\sigma} = \rho(\boldsymbol{\pi} - \tilde{\boldsymbol{X}}\boldsymbol{w}^k) + \boldsymbol{y}^k. \tag{34}$$

Having adjusted the transition matrix $\boldsymbol{M}(\boldsymbol{\pi})$ thusly, $\boldsymbol{\pi}$ can be obtained by repeated iterations of ILSRX (29), with $\tilde{\pi}(\boldsymbol{X};\boldsymbol{w})$ and D_p given by Eq. (30) and Eq. (32), respectively. We refer to this instance of Algorithm 1 as Spectral Ranking-l2 (SR-l2).

We note that, as Problem (19) is non-convex, selecting a good initialization point is important in practice. We discuss initialization below, leaving the additional theoretical guarantees to Section 3.7.

Initialization. We initialize $\boldsymbol{w}$ so that $\tilde{\boldsymbol{X}}\boldsymbol{w}$ is a good approximation of Plackett-Luce scores. We use a technique akin to Saha and Rajkumar [2018], applied to our affine setting. Given a distribution over queries $A \subseteq [n]$, let $P_{ij} = \mathbb{E}_A[c = i|\{i, j\} \subseteq A]$ be the probability that i is chosen given a query A that contains both i and j. By (1), for $i, j \in [n]$, $\frac{P_{ij}}{P_{ji}} = \frac{\pi_i}{\pi_j} = \frac{\boldsymbol{x}_i^\top\boldsymbol{a}+b}{\boldsymbol{x}_j^\top\boldsymbol{a}+b}$, or:

$$\delta_{ij}(\boldsymbol{w}) \equiv (P_{ij}\boldsymbol{x}_j - P_{ji}\boldsymbol{x}_i)^\top\boldsymbol{a} + (P_{ij} - P_{ji})b = 0. \tag{35}$$

Motivated by (35), we estimate P_{ij} empirically from $\mathcal{D}$, and obtain our initialization $\boldsymbol{w}^0 = (\boldsymbol{a}^0, b^0) \in \mathbb{R}^{d+1}$ by solving (35) in the least-square sense; that is,

$$\boldsymbol{w}^0 = \arg\min_{\boldsymbol{w}\in\mathbb{R}^{d+1}:\tilde{\boldsymbol{X}}\boldsymbol{w}\geq\mathbf{0}\wedge\mathbf{1}^\top\tilde{\boldsymbol{X}}\boldsymbol{w}=1} \sum_{i,j}\delta_{ij}^2(\boldsymbol{w}). \tag{36}$$

Note that this is a convex quadratic program. Given $\boldsymbol{w}^0$, we generate the initial Plackett-Luce scores via the affine parametrization $\boldsymbol{\pi}^0 = \tilde{\boldsymbol{X}}\boldsymbol{w}^0$.

3.4 Spectral Algorithm for Logistic Regression

We describe here how to apply Algorithm 1 on logistic regression of Plackett-Luce scores via:

$$\log \tilde{\boldsymbol{\pi}}(\boldsymbol{X};\boldsymbol{w}) = \boldsymbol{X}\boldsymbol{w}, \tag{37}$$

where $\log \boldsymbol{\pi} = [\log \pi_i]_{i\in[n]}$ is the $\mathbb{R}^n$ vector generated by applying log to $\boldsymbol{\pi}$ element-wise. Recall from Section 3.1 that the resulting MLE problem is given by:

$$\min_{\boldsymbol{w}\in\mathbb{R}^d} \mathcal{L}\left(\mathcal{D} \mid \tilde{\pi}(\boldsymbol{X};\boldsymbol{w}) \equiv [e^{\boldsymbol{w}^\top \boldsymbol{x}_i}]_{i\in[n]}\right), \tag{38}$$

which is convex, and can thus be solved by Newton's method. Nevertheless, we would like to accelerate its computation via a spectral method akin to ILSR.

Mirroring the affine case, we employ Algorithm 1 with quadratic proximal penalty:

$$D_p(\boldsymbol{\pi}||\tilde{\boldsymbol{\pi}}) = \frac{1}{2} \| \boldsymbol{\pi} - \tilde{\boldsymbol{\pi}} \|_2^2 . \tag{39}$$

As a result, the regression step (21b) corresponds to:

$$\begin{aligned} \boldsymbol{w}^{k+1} &= \arg\min_{\boldsymbol{w}\in\mathbb{R}^{d+1}} \boldsymbol{y}^{k^T}(\log \boldsymbol{\pi}^{k+1} - \boldsymbol{X}\boldsymbol{w}) + \tfrac{\rho}{2} \| \log \boldsymbol{\pi}^{k+1} - \boldsymbol{X}\boldsymbol{w} \|_2^2 \\ &= (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T(\log \boldsymbol{\pi}^{k+1} + \frac{1}{\rho}\boldsymbol{y}^k), \end{aligned} \tag{40}$$

which is again a quadratic minimization and admits a closed form solution.

More importantly, the score update step (21a) is amenable to the spectral solution established in Theorem 3.2. *Mutatis mutandis*, following the same manipulations in Lemma 3.1, a stationary point of the objective in each step (21a) can be cast as the stationary distribution of the continuous-time MC with transition rates $\mu_{ji}(\boldsymbol{\pi})$, $i, j \in [n]$ (Eq. (27)), where vector $\boldsymbol{\sigma} = [\sigma_i]_{i\in[n]}$ is now given by:

$$\sigma_i = \frac{\rho(\log \pi_i - \boldsymbol{x}_i^T\boldsymbol{w}^k) + y_i^k}{\pi_i}, \quad i \in [n] . \tag{41}$$

Having adjusted the transition matrix $\boldsymbol{M}(\boldsymbol{\pi})$ thusly, $\boldsymbol{\pi}$ can again be obtained by repeated iterations of ILSRX (29), with $\tilde{\pi}(\boldsymbol{X};\boldsymbol{w})$ and D_p given by Eq. (37) and Eq. (39), respectively. We refer to this instance of Algorithm 1 as Spectral Ranking-l2-log (SR-l2-log).

Initialization. Similar to the initialization of SR-l2 (c.f. Eq. (36)), we initialize $\boldsymbol{w}$ so that the initial scores obey the Plackett-Luce model, mirroring the approach by Saha and Rajkumar [2018]. Defining P_{ij}, $i, j \in [n]$ the same way, and using the logistic parametrization in Eq. (17), we have that:

$$\frac{P_{ij}}{P_{ji}} = \frac{\pi_i}{\pi_j} = e^{\boldsymbol{w}^T(\boldsymbol{x}_i - \boldsymbol{x}_j)}. \tag{42}$$

Accordingly, we initialize $\boldsymbol{w}$ as:

$$\boldsymbol{w}^0 = \arg\min_{\boldsymbol{w}\in\mathbb{R}^d} \sum_{i,j} \left(\boldsymbol{w}^T(\boldsymbol{x}_i - \boldsymbol{x}_j) - \log\left(\frac{\hat{P}_{ij}}{\hat{P}_{ji}}\right)\right)^2, \tag{43}$$

where $\hat{P}_{ij}$, $i, j \in [n]$, are again empirical estimates obtained from dataset $\mathcal{D}$. Given $\boldsymbol{w}^0$, we generate the initial Plackett-Luce scores via the logistic parametrization $\boldsymbol{\pi}^0 = [e^{\boldsymbol{x}_i^T\boldsymbol{w}^0}]_{i\in[n]}$.

3.5 Spectral Algorithm for Deep Neural Network (DNN) Regression

In this section, we employ Algorithm 1 for deep neural network (DNN) regression of Plackett-Luce scores via $\tilde{\boldsymbol{\pi}}(\boldsymbol{X};\boldsymbol{w})$ (c.f. Problem (18)).

To regress scores via a generic DNN $\tilde{\boldsymbol{\pi}}(\boldsymbol{X};\boldsymbol{w})$, we generalize the traditional quadratic proximal penalty $D_p(\boldsymbol{\pi}||\tilde{\boldsymbol{\pi}}) = \frac{1}{2} \parallel \boldsymbol{\pi} - \tilde{\boldsymbol{\pi}} \parallel_2^2$ to KL-divergence, which is naturally suited to Problem (19): this is because the scores computed by ILSRX (29) form, by construction, a distribution over $[n]$. This generalization to KL penalty is still amenable to a spectral solution via ILSRX (29), as Theorem 3.2 holds for any proximal penalty D_p; in practice, this also leads to a significantly improved performance over an ℓ_2-proximal penalty (by up to 56% Top-1 accuracy and 25% Kendall-Tau correlation, as discussed in Sec. 4.9). As shown in Eq. (48) below, the KL-divergence penalty leads to training $\tilde{\boldsymbol{\pi}}(\boldsymbol{X};\boldsymbol{w})$ with a max-entropy loss and an additional linear term at each ADMM iteration. Compared to the ℓ_2-proximal penalty, max-entropy has been observed to converge faster and lead to better fitting [Bosman et al. 2020; Caruana and Niculescu-Mizil 2004; Golik et al. 2013]. Finally, ADMM with Bregman divergence proximal penalties, including KL divergence, has been shown to offer local convergence guarantees for non-convex problems (c.f. Section 2.2.3); this further motivates us to employ KL over other distance measures between probability distributions.

We implement DSR with the two proximal penalty functions $D_p(\boldsymbol{\pi}||\tilde{\boldsymbol{\pi}})$ mentioned above: Kullback-Leibler (KL) divergence [Kullback and Leibler 1951] and ℓ_2 norm. We describe implementation specifics for each of these cases below.

ℓ_2 proximal penalty. For ADMM with quadratic proximal penalty:

$$D_p(\boldsymbol{\pi}||\tilde{\boldsymbol{\pi}}) = \frac{1}{2} \parallel \boldsymbol{\pi} - \tilde{\boldsymbol{\pi}} \parallel_2^2, \tag{44}$$

the regression step (21b) of DSR becomes a least squares minimization of the form:

$$\boldsymbol{w}^{k+1} = \underset{\boldsymbol{w}\in\mathbb{R}^{d'}}{\arg\min} \;\; \parallel \tilde{\boldsymbol{\pi}}(\boldsymbol{X};\boldsymbol{w}) - (\boldsymbol{\pi}^{k+1} + \frac{1}{\rho}\boldsymbol{y}^k) \parallel_2^2 . \tag{45}$$

As a result, we train the DNN regressor $\tilde{\boldsymbol{\pi}}(\boldsymbol{X};\boldsymbol{w})$ via SGD optimization over the loss function (45).

Moreover, the score update step (21a) is amenable to the spectral approach established in Theorem 3.2 and solved via repeated iterations of ILSRX (29), where:

$$\sigma_i(\boldsymbol{\pi}) = \rho \frac{\partial D_p(\boldsymbol{\pi}||\tilde{\boldsymbol{\pi}}^k)}{\partial \pi_i} + y_i^k = \rho(\pi_i - \tilde{\pi}(\boldsymbol{x}_i, \boldsymbol{w}^k)) + y_i^k, \tag{46}$$

for all $i \in [n]$ in the transition rates (27). We refer to this instance of Algorithm 1 as Deep Spectral Ranking-l2 (DSR-l2).

KL proximal penalty. For ADMM with KL proximal penalty:

$$D_p(\boldsymbol{\pi}||\tilde{\boldsymbol{\pi}}) = \sum_{i=1}^{n} \pi_i \log \frac{\pi_i}{\tilde{\pi}_i}, \tag{47}$$

the optimization step (21b) to train $\tilde{\boldsymbol{\pi}}(\boldsymbol{X};\boldsymbol{w})$ corresponds to minimizing a max-entropy loss with an additional linear term:

$$\boldsymbol{w}^{k+1} = \underset{\boldsymbol{w}\in\mathbb{R}^{d'}}{\arg\min} \;\; \sum_{i=1}^{n} \left(-\frac{y_i^k}{\rho} \tilde{\pi}(\boldsymbol{x}_i, \boldsymbol{w}) - \pi_i^{k+1} \log \tilde{\pi}(\boldsymbol{x}_i, \boldsymbol{w}) \right). \tag{48}$$

Moreover, the score update step (21a) is again solved via repeated iterations of ILSRX (29), where:

$$\sigma_i(\boldsymbol{\pi}) = \rho \frac{\partial D_p(\boldsymbol{\pi}||\tilde{\boldsymbol{\pi}}^k)}{\partial \pi_i} + y_i^k = \rho\big(1 + \log \frac{\pi_i}{\tilde{\pi}(\boldsymbol{x}_i, \boldsymbol{w}^k)}\big) + y_i^k, \tag{49}$$

for all $i \in [n]$ in the transition rates (27). We refer to this instance of Algorithm 1 as Deep Spectral Ranking-KL (DSR-KL).

Initialization. For both cases of penalties, we initialize all samples with equal scores, i.e., $\boldsymbol{\pi} = \tilde{\boldsymbol{\pi}} = \frac{1}{n}\mathbf{1}$. At each ADMM iteration k, we initialize $\boldsymbol{\pi}$ with $\boldsymbol{\pi}^{k-1}$, and update $\boldsymbol{\pi}$ via ILSRX. Then, we initialize the DNN parameters $\boldsymbol{w}$ with $\boldsymbol{w}^{k-1}$, and fine tune the DNN $\tilde{\boldsymbol{\pi}}(\boldsymbol{X};\boldsymbol{w})$ via SGD over Eq. (45) or Eq. (48).

3.6 Computational Complexity

Each iteration of our spectral algorithm (c.f. Algorithm 1) involves the three steps in Eq. (21). One iteration of ILSRX at step (21a) is $O(Km + n^2)$ for constructing the transition rates via Eq. (27) and for finding the stationary distribution $\boldsymbol{\pi}$ via, e.g., a power method [Lei et al. 2016], respectively. It is important to note that the number of ranking observations m appears only in the construction of the transition rates through the terms $\mu_{j,i}$ in Eq. (21a), which in turn are computed via sums w.r.t. these m quantities (see Eq. (27)). This is important, as the (potentially exponential in K) ranking observations *affect the complexity only through scalar summations*, which themselves can be easily parallelized (via, e.g., reduce-by-key operations). This is in sharp contrast to the classic gradient-based optimization (c.f. Sec. 3.1 and Fig. 1), in which m directly determines the size of each epoch.

We discuss the complexity of the regression step (21b) separately for shallow and DNN model regression cases. For shallow model regression via affine or logistic parametrizations, i.e., SR-l2 and SR-l2-log, the regression step (21b) takes the specific forms (33) and (40), respectively. In both cases, $\boldsymbol{w}$ update is $O\big(n(d+1)\big)$ as a matrix-vector multiplication, since the matrix $(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T$ can be precomputed. For DNN regression via DSR, the regression step (21b) is equivalent to training $\tilde{\boldsymbol{\pi}}(\boldsymbol{X};\boldsymbol{w})$ via Eq. (22). Most importantly, the length of each training epoch, is *linear in number of samples* n: each optimization step goes over $O(n)$ samples to update the d' parameters. Again, this is in contrast to the $m = O(n^K)$-length epochs if one was minimizing the likelihood function directly via standard gradient descent methods. Finally, given $\tilde{\boldsymbol{\pi}}$ and $\boldsymbol{\pi}$, the update of $\boldsymbol{y}$ at step (21c) is $O(n)$.

Overall, updating $\boldsymbol{\pi}$ via ILSRX at step (21a) and $\boldsymbol{w}$ via gradient-based training at step (21b) are both iterative processes, for which the computational complexity until convergence have not yet been theoretically shown. That said, the literature on the complexity and convergence of iterative optimization over such highly non-convex objectives illustrates the challenges behind such a theoretical analysis [Jain and Kar 2017; Taheri et al. 2021]. Nevertheless, our Theorem 3.4 below, establishing convergence guarantees in the affine regression case, is a step towards this direction. This is in sharp contrast to, e.g., siamese networks, in which m characterizes the length of an entire training epoch.

3.7 Theoretical Guarantees

As the ranking regression problem (19) is in general non-convex, we aim to obtain convergence guarantees for our spectral algorithm (c.f. Algorithm 1). To do so, we focus on the affine regression case explained in Section 3.3. Given this case, we wish to establish conditions such that: (i) the scores computed at each ADMM iteration form a global optimum w.r.t. step (21a), and (ii) the scores and model parameters generated by Algorithm 1 converge to a stationary point of Problem (16).

Recall from Section 3.3 that the affine regression problem is given by:

$$\min_{\boldsymbol{w}\in\mathbb{R}^{d+1}:\tilde{\pi}(\boldsymbol{X};\boldsymbol{w})\geq\mathbf{0}} \mathcal{L}\left(\mathcal{D} \mid \tilde{\pi}(\boldsymbol{X};\boldsymbol{w}) \equiv \tilde{\boldsymbol{X}}\boldsymbol{w}\right). \tag{50}$$

Table 2. No. of samples (n), no. of features (d), no. of observations (m), and query size (K) for datasets with image or numerical features ($\boldsymbol{X}$)

Spec.	Dataset							
	ROP-num / ROP-img	FAC	Pairwise Sushi	Triplet Sushi	ICLR-3/4	Movehub-Cost-4/5	Movehub-Quality-4/5	IMDB-4
K	2			3	3/4	4/5	4/5	4
n	100	1000	100			50		
d	$143/224 \times 224$	50	18		768	6	5	36
m	29, 705	728	450	1200	120, 324/ 2, 248, 524	230, 298/ 2, 118, 756	230, 298/ 2, 118, 756	85, 583
$\boldsymbol{X}$	numerical / image	numerical						

In this setting, we employ ADMM with standard ℓ_2 proximal penalty $D_p(\boldsymbol{\pi}||\tilde{\boldsymbol{\pi}}) = \frac{1}{2} \| \boldsymbol{\pi} - \tilde{\boldsymbol{\pi}} \|_2^2$. As Problem (19) is non-convex, condition (23) is in general *not* sufficient for optimality w.r.t. step (21a). To show this, we require the following technical assumption:

Assumption 3.1. *For $\{\boldsymbol{\pi}^k\}_{k\in\mathbb{N}}$, given by* (21a)*, there exists an $\epsilon > 0$ such that $\pi_i^k > \epsilon$ for all $i \in [n]$ and $k \in \mathbb{N}$.*

In other words, we assume that the scores computed at each ADMM iteration are strictly positive. Under this assumption, we show that stationarity implies optimality w.r.t. (21a) for large enough ρ:

Theorem 3.3. *Under Assumption 3.1, for $\rho \geq \frac{2}{\epsilon^2} \max_i \sum_{\ell | i \in A_\ell} \frac{1}{|A_\ell|^2}$ and $D_p(\boldsymbol{\pi}||\tilde{\boldsymbol{\pi}}) = \frac{1}{2} \| \boldsymbol{\pi} - \tilde{\boldsymbol{\pi}} \|_2^2$, a $\boldsymbol{\pi} > \mathbf{0}$ satisfying condition* (23) *is a minimizer of* (21a)*.*

The proof is in Appendix E. Theorem 3.3 implies that given large enough ρ, the stationary scores satisfying condition (23) also form a global optimum of the ADMM step (21a). Using this result, we further establish the following convergence guarantee for Algorithm 1 in the affine regression case:

Theorem 3.4. *For $\tilde{\boldsymbol{\pi}}(\boldsymbol{X}; \boldsymbol{w}) = \tilde{\boldsymbol{X}}\boldsymbol{w}$ and $D_p(\boldsymbol{\pi}||\tilde{\boldsymbol{\pi}}) = \frac{1}{2} \| \boldsymbol{\pi} - \tilde{\boldsymbol{\pi}} \|_2^2$, suppose that there exists $\kappa > 0$ such that $\tilde{\boldsymbol{X}}^T \tilde{\boldsymbol{X}} \succeq \kappa \boldsymbol{I}$ and the sequence $\{(\boldsymbol{\pi}^k, \boldsymbol{y}^k, \boldsymbol{w}^k)\}_{k\in\mathbb{N}}$ generated by* (21) *is bounded. Then, under Assumption 3.1, for $\rho > \frac{2 \max_i |W_i|}{\epsilon^2}$ and $W_i = \{\ell \,|\, i \in A_\ell, c_\ell = i\}$ denoting the observations where sample $i \in [n]$ is chosen, the sequence $\{(\boldsymbol{\pi}^k, \boldsymbol{y}^k, \boldsymbol{w}^k)\}_{k\in\mathbb{N}}$ generated by* (21) *converges to a point that satisfies the Karush-Kuhn-Tucker (KKT) conditions of* (16)*.*

The proof is in Appendix F. By Theorem 3.4, we conclude that the scores $\boldsymbol{\pi}$ and the affine regression parameters $\boldsymbol{w}$ generated by Algorithm 1 converge to a stationary point of Problem (16) for large enough ρ.

4 EXPERIMENTS

4.1 Datasets

We evaluate our algorithms on synthetic and real-life datasets, summarized in Table 2.

Synthetic Datasets. We generate the feature vectors $\boldsymbol{x}_i \in \mathbb{R}^d, \quad i \in [n]$ from $\mathcal{N}(\mathbf{0}, \sigma_x^2 \boldsymbol{I}_{d\times d})$ and a common parameter vector $\boldsymbol{w} \in \mathbb{R}^d$ from $\mathcal{N}(\mathbf{0}, \sigma_\beta^2 \boldsymbol{I}_{d\times d})$. Then, we generate the Plackett-Luce scores via the logistic parametrization $\boldsymbol{\pi} = [e^{\boldsymbol{x}_i^T \boldsymbol{w}}]_{i\in[n]}$. We normalize the resulting scores, so that $\mathbf{1}^\top \boldsymbol{\pi} = 1$. We set $\sigma_x^2 = \sigma_\beta^2 = 0.8$ in all experiments. Given $\boldsymbol{\pi}$, we generate each observation in $\mathcal{D}$ as follows:

we first select $|A_\ell| = 2$ samples out of n samples uniformly at random. Then, we generate the choice c_l, $l \in [m]$ from the Plackett-Luce model given by Eq. (1).

Retinopathy of Prematurity (ROP). The Retinopathy of Prematurity (ROP) dataset contains $n = 100$ vessel-segmented retina images. Experts are provided with two images and are asked to choose the image with higher severity of the ROP disease. Five experts independently label 5941 image pairs; the resulting dataset contains $m = 29705$ pairwise comparisons. Note that some pairs are labelled more than once by different experts. We employ two ROP datasets, ROP-num and ROP-img for shallow and DNN regression, respectively: Each sample in ROP-num has $d = 143$ extracted numerical features [Ataer-Cansızoğlu 2015], while each sample in ROP-img is an image with dimensions $d = 224 \times 224$.

Filter Aesthetic Comparison (FAC). The Filter Aesthetic Comparison (FAC) dataset [Sun et al. 2017] contains 1280 unfiltered images pertaining to 8 different categories. Twenty-two different image filters are applied to each image. Labelers are provided with two filtered images and are asked to identify which image has better quality. We select $n = 1000$ images within one category, as only the filtered image pairs that are within the same category are compared. The resulting dataset contains $m = 728$ pairwise comparisons. Moreover, for each image, we extract features via a state-of-the-art convolutional neural network architecture, namely GoogLeNet [Szegedy et al. 2015], with weights pre-trained on the ImageNet dataset [Deng et al. 2009]. We select $d = 50$ of these features by Principal Component Analysis [Jolliffe 1986].

SUSHI. The SUSHI Preference dataset [Kamishima et al. 2009] contains $n = 100$ sushi ingredients with $d = 18$ features. Each of the 5000 customers independently ranks 10 ingredients according to her preferences. We select the rankings provided by 10 customers, where an ingredient is ranked higher if it precedes the other ingredients in a customer's ranked list. We generate two datasets: triplet Sushi containing $m = 1200$ rankings of $|A_\ell| = 3$ ingredients, and pairwise Sushi containing $m = 450$ pairwise comparisons.

International Conference on Learning Representations (ICLR). The ICLR Dataset contains abstracts and reviewer ratings of 2561 papers that are submitted to ICLR 2020 conference and are available on OpenReview website [Sun 2020]. We choose the top $n = 100$ papers, and extract $d = 768$ numerical features from each abstract using the Deep Bidirectional Transformers (BERT) [Devlin et al. 2019] architecture, pre-trained on the Books Corpus dataset [Zhu et al. 2015] and English Wikipedia. We normalize $\boldsymbol{X}$ to have 0 mean and unit variance over samples $[n]$. We generate all possible $m = 120, 324 (2, 248, 524)$ $K = 3(4)$-way rankings w.r.t. the relative order of the average reviewer ratings. We add noise to the resulting rankings following the same process as Movehub-Cost.

Movehub-Cost. The Movehub-Cost dataset contains the total ranking of 216 cities w.r.t. cost of living [Blitzer 2017]. Each city is associated with $d = 6$ numerical features, which are average costs for cappuccino, cinema, wine, gasoline, rent, and disposable income. We normalize $\boldsymbol{X}$ to have 0 mean and unit variance over samples $[n]$. We select $n = 50$ cities and generate all $m = 230, 298 (2, 118, 756)$ $K = 4(5)$-way rankings w.r.t. the relative order of the queried cities in the total ranking. To mimic the real-life noise introduced by human labelling, we apply the following post-processing to the resulting rankings: For each ranking, we sample a value uniformly at random in $[0, 1]$. If the value is less than 0.1, we add noise to the ranking by a cyclic permutation of the ranked samples.

Movehub-Quality. The Movehub-Quality dataset contains total ranking of the same 216 cities as Movehub-Cost, this time w.r.t. quality of life. Each city is associated with $d = 5$ numerical features, including overall scores for purchase power, healthcare, pollution, quality of life, and crime. We normalize $\boldsymbol{X}$ to have 0 mean and unit variance over samples $[n]$. We select $n = 50$ cities and generate all $m = 230, 298 (2, 118, 756)$ $K = 4(5)$-way rankings w.r.t. the relative order of the

queried cities in the total ranking. We add noise to the rankings following the same process as Movehub-Cost.

IMDB. The IMDB Movies Dataset contains IMDB ratings of 14,762 movies, each of which is associated with $d = 36$ numerical features [Leka 2016]. We normalize $\boldsymbol{X}$ to have 0 mean and unit variance over samples $[n]$. We select $n = 50$ movies and generate all possible $m = 85,583$ $K = 4$-way rankings w.r.t. the relative order of the ratings of queried movies. We add noise to the resulting rankings following the same process as Movehub-Cost.

4.2 Experiment Setup

We partition each dataset into training and test sets in two ways. In *rank partitioning*, we partition the dataset w.r.t. $[m]$, using 90% of the m observations for training, and the remaining 10% for testing. In *sample partitioning*, we partition $[n]$, using 90% of the n samples for training, and the remaining 10% for testing. In this setting, observations containing samples from both training and validation/test sets are discarded. We perform cross validation on the resulting training sets. We use 10 folds for synthetic datasets, ROP-num , FAC, and Triplet Sushi, and 3 folds for the other datasets. For synthetic datasets, we also repeat experiments over 5 random generations.

To evaluate the convergence speed of each algorithm, we measure the elapsed time, including time spent in initialization, in seconds (Time). Moreover, we measure the prediction performance using Top-1 accuracy (Top-1 Acc.) and Kendall-Tau correlation (KT) [Kendall 1938] on validation and test sets (c.f. 4.7). For synthetic datasets, we also measure the quality of convergence by the norm of the difference between estimated and true Plackett-Luce scores ($\triangle\boldsymbol{\pi}$); lower values indicate better estimation. We report averages and standard deviations over folds for all algorithms except for the siamese network method: as training takes several hours, we execute only one fold.

4.3 Shallow Regression Competing Methods

We implement[1] seven competing algorithms. Four are *feature methods*, i.e., algorithms that regress Plackett-Luce scores from features: SR-l2 described in Section 3.3, SR-l2-log described in Section 3.4, sequential least-squares quadratic programming (SLSQP), that solves (16), and Newton on $\boldsymbol{w}$, that solves the convex problem (17) via Newton's method. The remaining three are *featureless methods*, i.e., algorithms that learn the Plackett-Luce scores from the choice observations alone: Iterative Luce Spectral Ranking (ILSR) described by Eq.(7), the Minorization-Maximization (MM) algorithm [Hunter 2004], and Newton on $\boldsymbol{\theta}$ that solves Eq. (3) via the reparametrization $\pi_i = e^{\theta_i}$, $i \in [n]$ using Newton's method on $\boldsymbol{\theta} = [\theta_i]_{i\in[n]}$. We expand upon the implementation of all algorithms below.

SR-l2. We compute the stationary distribution at each iteration of ILSRX (c.f. Eq. (29)) using the power method [Lei et al. 2016]. As the stopping criterion, we use $\|\boldsymbol{\pi}^k - \boldsymbol{\pi}^{k-1}\|_2 < r_{\texttt{tol}} \|\boldsymbol{\pi}^k\|_2$ and $\|\tilde{\boldsymbol{X}}\boldsymbol{w}^k - \tilde{\boldsymbol{X}}\boldsymbol{w}^{k-1}\|_2 < r_{\texttt{tol}} \|\tilde{\boldsymbol{X}}\boldsymbol{w}^k\|_2$. We set the relative tolerance $r_{\texttt{tol}} = 10^{-4}$ for all experiments. We use the same relative tolerance for the stopping criterion of the power method. We set $\rho = 1$ in our experiments, which is a standard choice in the ADMM literature [Boyd et al. 2011]. In our experiments, we consistently observe that Eq.(26) is satisfied. That is why, we use $c_j = \frac{-\pi_j\sigma_j}{\sum_{i\in[n]_-}\pi_i\sigma_i}$, $j \in [n]_+$ to calculate the transition rates (27).

SR-l2-log. As the stopping criterion, we use $\|\boldsymbol{\pi}^k - \boldsymbol{\pi}^{k-1}\|_2 < r_{\texttt{tol}} \|\boldsymbol{\pi}^k\|_2$ and $\|e^{\boldsymbol{X}\boldsymbol{w}^k} - e^{\boldsymbol{X}\boldsymbol{w}^{k-1}}\|_2 < r_{\texttt{tol}} \|e^{\boldsymbol{X}\boldsymbol{w}^k}\|_2$, where exponentiation is applied elementwise.

SLSQP. We initialize SLSQP the same as SR-l2 (c.f. Section 3.3). As stopping criterion, we use $\|\boldsymbol{\pi}^k - \boldsymbol{\pi}^{k-1}\|_2 < r_{\texttt{tol}} \|\boldsymbol{\pi}^k\|_2$, where $\boldsymbol{\pi}^k = \boldsymbol{X}\boldsymbol{a}^k + \boldsymbol{1}b^k$, $k \in \mathbb{N}$. Each iteration of SLSQP is $O\Big(\sum_{\ell\in\mathcal{D}}\big(|A_\ell|(d+1)\big) + (d+1)^2\Big)$ for constructing the gradient of Eq. (15) w.r.t. $\boldsymbol{w}$ and updating $\boldsymbol{w}$, respectively.

[1] Our code for shallow model regression is publicly available at `https://github.com/neu-spiral/FastAndAccurateRankingRegression`

Newton on $\boldsymbol{w}$. We initialize Newton on $\boldsymbol{w}$ the same as SR-l2-log (c.f. Section 3.4). As stopping criterion, we use $\|\boldsymbol{\pi}^k - \boldsymbol{\pi}^{k-1}\|_2 < r_{\texttt{tol}} \|\boldsymbol{\pi}^k\|_2$, where $\boldsymbol{\pi}^k = [e^{\boldsymbol{x}_i^T \boldsymbol{w}^k}]_{i\in[n]}$, $k \in \mathbb{N}$. Each iteration of Newton on $\boldsymbol{w}$ is $O\Big(\sum_{\ell\in\mathcal{D}} \left(|A_\ell|\, d^2\right) + d^2\Big)$ for constructing the Hessian of Eq. (15) w.r.t. $\boldsymbol{w}$ and updating $\boldsymbol{w}$, respectively.

ILSR. We initialize ILSR with $\boldsymbol{\pi}^0 = \frac{1}{n}\mathbf{1}$. We compute the stationary distribution at each iteration of ILSR using the power method. As the stopping criterion, we use $\|\boldsymbol{\pi}^k - \boldsymbol{\pi}^{k-1}\|_2 < r_{\texttt{tol}} \|\boldsymbol{\pi}^k\|_2$. Each iteration of ILSR is $O\Big(\sum_{\ell\in\mathcal{D}} \left(|A_\ell|\right) + n^2\Big)$ for constructing the transition matrix $\boldsymbol{\Lambda}(\boldsymbol{\pi})$ (c.f. Eq.(6)) and finding the stationary distribution $\boldsymbol{\pi}$, respectively.

MM. We initialize MM with $\boldsymbol{\pi}^0 = \frac{1}{n}\mathbf{1}$. As the stopping criterion, we use $\|\boldsymbol{\pi}^k - \boldsymbol{\pi}^{k-1}\|_2 < r_{\texttt{tol}} \|\boldsymbol{\pi}^k\|_2$. Each iteration of MM is $O\Big(\sum_{\ell\in\mathcal{D}} \left(|A_\ell|\right)\Big)$.

Newton on $\boldsymbol{\theta}$. We initialize Newton on $\boldsymbol{\theta}$ with $\boldsymbol{\theta}^0 = [\theta_i^0]_{i\in[n]} = \mathbf{0}$. As stopping criterion, we use $\|\boldsymbol{\pi}^k - \boldsymbol{\pi}^{k-1}\|_2 < r_{\texttt{tol}} \|\boldsymbol{\pi}^k\|_2$, where $\boldsymbol{\pi}_i^k = e^{\theta_i^k}$, $i \in [n]$, $k \in \mathbb{N}$. Each iteration of Newton on $\boldsymbol{\theta}$ is $O\Big(\sum_{\ell\in\mathcal{D}} \left(|A_\ell|^2\right) + n^2\Big)$ for constructing the Hessian of Eq. (15) w.r.t. $\boldsymbol{\theta}$ and updating $\boldsymbol{\theta}$, respectively.

4.4 DNN Regression Competing Methods

We implement[2] five competing algorithms. DSR with KL penalty (DSR-KL), DSR with ℓ_2 norm penalty (DSR-l2), and siamese network competitor regress scores via a DNN $\tilde{\boldsymbol{\pi}}(\boldsymbol{X};\boldsymbol{w})$ with parameters $\boldsymbol{w}$. SR-l2 and SR-KL use an affine regressor $\tilde{\boldsymbol{\pi}} = \boldsymbol{X}\boldsymbol{a} + \mathbf{1}b$ with parameters $\boldsymbol{a}$ and b, and solve (19) via ADMM with ℓ_2 and KL penalties, respectively.

DSR-KL and DSR-l2 are explained in Section 3.5. We compute the stationary distribution at each iteration of ILSRX (c.f. Eq. (29)) using the power method [Lei et al. 2016]. At each ADMM iteration, we fine tune the DNN regressor $\tilde{\pi}(\boldsymbol{X};\boldsymbol{w})$ via Adam optimization [Kingma and Ba 2015] over Eq. (22). We check the convergence of $\mathcal{L}$ on the training set and Kendall-Tau correlation (KT) evaluations on the validation set (c.f. 4.7) as the stopping criterion for: (i) fine-tuning $\tilde{\pi}(\boldsymbol{X};\boldsymbol{w})$ at each ADMM iteration, and (ii) overall DSR-KL algorithm. We declare convergence when KT on validation set does not change for 5 iterations, for maximum total of 50 iterations, with relative tolerance $r_{\texttt{tol}} = 10^{-4}$. We use the same relative tolerance for the stopping criterion of the power method.

To aid convergence in practice [Boyd et al. 2011], we update the dual variable at each ADMM iteration with a multiplicative smoothing parameter $\gamma^k = 1/k$. We also adapt the penalty parameter as:

$$\rho^{k+1} = \begin{cases} \tau\rho^k, & \text{if } \|\tilde{\boldsymbol{\pi}}^k - \boldsymbol{\pi}^k\|_2^2 > \beta \|\tilde{\boldsymbol{\pi}}^k - \tilde{\boldsymbol{\pi}}^{k-1}\|_2^2 \\ \frac{\rho^k}{\tau}, & \text{if } \|\tilde{\boldsymbol{\pi}}^k - \boldsymbol{\pi}^k\|_2^2 < \beta \|\tilde{\boldsymbol{\pi}}^k - \tilde{\boldsymbol{\pi}}^{k-1}\|_2^2 \\ \rho^k, & \text{otherwise,} \end{cases}$$

where $\tau = 2$ and $\beta = 10$.

The siamese network competitor minimizes Eq. (15) w.r.t. $\boldsymbol{w}$ via SGD. We train a siamese network architecture with base network $\tilde{\pi}$ on observations $\mathcal{D}$ via Adam optimization [Kingma and Ba 2015] over Eq. (15). We employ the same convergence criterion and experiment setup as DSR-KL and DSR-l2 to train and optimize the siamese network.

Finally, we implement two spectral algorithms that regress Plackett-scores via an affine model, i.e., $\tilde{\boldsymbol{\pi}} = \boldsymbol{X}\boldsymbol{a} + \mathbf{1}b$: SR-l2, and its variant SR-KL that uses KL proximal penalty instead of ℓ_2 norm penalty for ADMM. As the stopping criterion for both algorithms, we use $\|\boldsymbol{\pi}^k - \boldsymbol{\pi}^{k-1}\|_2 < r_{\texttt{tol}} \|\boldsymbol{\pi}^k\|_2$ and

[2] Our code for DNN regression is publicly available at `https://github.com/neu-spiral/DeepSpectralRanking`

$\|(\boldsymbol{X}\boldsymbol{a}^k + \mathbf{1}b^k) - (\boldsymbol{X}\boldsymbol{a}^{k-1} + \mathbf{1}b^{k-1}\|_2 < r_{\texttt{tol}} \|\boldsymbol{X}\boldsymbol{a}^k + \mathbf{1}b^k\|_2$. We set the ADMM penalty parameter as $\rho = 1$.

4.5 DNN Architecture and Training Details

To evaluate each method on ROP-img , we choose $\tilde{\pi}(\boldsymbol{X};\boldsymbol{w})$ as a state-of-the-art convolutional neural network architecture, namely GoogLeNet [Szegedy et al. 2015], followed by a fully connected output layer comprising a single neuron with sigmoid activation. We initialize the convolutional layers with weights pre-trained on the ImageNet dataset [Deng et al. 2009]. For other datasets, we design $\tilde{\pi}(\boldsymbol{X};\boldsymbol{w})$ as a fully-connected architecture with relu activation for hidden layers and an output layer comprising a single neuron with sigmoid activation. We add ℓ_2 regularizers to all layers. For each configuration of (i) ℓ_2 regularization parameter varying in $[2 \times 10^{-5}, 2 \times 10^{-2}]$, (ii) learning rate varying in $[10^{-4}, 10^{-2}]$, and (iii) number of layers varying in $[1, 10]$ for fully connected $\tilde{\pi}(\boldsymbol{X};\boldsymbol{w})$, we run each method until convergence (c.f. 4.4 for criteria). We determine the best set of hyperparameters w.r.t. the prediction performance via cross validation (c.f. 4.2).

4.6 Execution Environment

In Tables 3 - 5, we measure timing on an Intel Xeon CPU E5-2680v2 2.8GHz with 128GB RAM. Particularly for experiments on larger synthetic datasets (c.f. Figures 3 - 4), we use an Intel Xeon CPU E5-2680v4 2.4GHz with 500GB RAM. For all DNN regression experiments, we employ NVIDIA GPUs with Intel Gold 6132@2.60Ghz CPUs and 128GB RAM.

4.7 Performance Metrics

We measure the prediction performance by Top-1 accuracy (Top-1 Acc.) and Kendall-Tau correlation (KT) on the test set. Let the test set be $\mathcal{D}_{\texttt{rank}} = \{(\alpha^\ell, A_\ell) \mid \ell \in \{1, ..., m_{\texttt{test}}\}\}$, where $\alpha^\ell = \alpha^\ell_1 \succ \alpha^\ell_2 \succ \cdots \succ \alpha^\ell_K$ is an ordered sequence of the samples in A_ℓ. Given A_ℓ, we predict the ℓ-th choice as $\hat{c}_\ell = \arg\max_{i \in A_\ell} \pi_i$. We calculate the Top-1 accuracy (Top-1 Acc.) as:

$$\text{Top-1 Acc.} = \frac{\sum_{\ell=1}^{m_{\texttt{test}}} \mathbb{1}(\hat{c}_\ell = \alpha^\ell_1)}{m_{\texttt{test}}} \in [0, 1]. \tag{51}$$

We also predict the ranking as $\hat{\alpha}^\ell = \arg\text{sort}[\pi_i]_{i \in A_\ell}$, i.e. sequence of the samples in A_ℓ ordered w.r.t. their estimated scores. We calculate Kendall-tau correlation (KT) [Kendall 1938] as a measure of the correlation between each true ranking α^ℓ and predicted ranking $\hat{\alpha}^\ell$, $\ell \in \{1, ..., m_{\texttt{test}}\}$. For observation ℓ, let $T_\ell = \sum_{t=1}^K \sum_{s=1}^K \mathbb{1}(\hat{\alpha}^\ell_t \succ \hat{\alpha}^\ell_s \wedge \alpha^\ell_t \succ \alpha^\ell_s)$ be the number correctly predicted ranking positions, and $F_\ell = \sum_{t=1}^K \sum_{s=1}^K \mathbb{1}(\hat{\alpha}^\ell_t \succ \hat{\alpha}^\ell_s \wedge \alpha^\ell_s \succ \alpha^\ell_t)$ be the number incorrectly predicted ranking positions. Then, KT is computed by:

$$\text{KT} = \frac{\sum_{\ell=1}^{m_{\texttt{test}}} (T_\ell - F_\ell) / \binom{K}{2}}{m_{\texttt{test}}} \in [-1, 1], \tag{52}$$

where $\binom{K}{2}$ is the number of sample pairs.

4.8 Shallow Regression Results

Sample partitioning. We begin with the experiments on sample partitioning, in which we partition samples $[n]$ into training and test sets. Table 3 shows the evaluations of all algorithms trained on a synthetic dataset with $n = 1000$ samples, $d = 100$ features, $m = 1000$ observations, and query size $|A_\ell| = 2$. SR-l2 and SR-l2-log converge $4 - 27$ times faster than other feature methods, i.e., Newton on $\boldsymbol{w}$ and SLSQP. Recall that in sample partitioning, training observations contain *only* training samples: test samples do not participate in any of the training observations. Thus, featureless methods ILSR, MM, and Newton on $\boldsymbol{\theta}$ *are no better than random predictors*, with 0.5 Top-1 Acc. and

Table 3. Evaluations on a synthetic dataset with $n = 1000$, $d = 100$, and $m = 1000$, partitioned w.r.t. sample partitioning and rank partitioning. We report the convergence time (Time), number of iterations until convergence (Iter), norm error in estimating true Plackett-Luce scores ($\triangle\boldsymbol{\pi}$), top-1 accuracy on the test set (Top-1 Acc.), and Kendall-Tau correlation on the test set (KT). ILSR, MM, and Newton on $\boldsymbol{\theta}$ do not use the features $\boldsymbol{X}$. Newton on $\boldsymbol{w}$ and SLSQP regress $\boldsymbol{\pi}$ from $\boldsymbol{X}$.

Partitioning	Method	Training Metrics			Performance Metrics on the Test Set	
		Time (s) ↓	Iter. ↓	$\triangle\boldsymbol{\pi}$ ↓	Top-1 Acc. ↑	KT ↑
Sample Partitioning	**SR-l2**	0.237 ± 0.006	4 ± 0	0.717 ± 0.207	0.831 ± 0.119	0.609 ± 0.247
	SR-l2-log	1.428 ± 2.595	49 ± 79	0.845 ± 0.204	0.668 ± 0.159	0.335 ± 0.318
	ILSR (no $\boldsymbol{X}$)	0.045 ± 0.002	2 ± 0	0.718 ± 0.207	0.5 ± 0.0	−1.0 ± 0.0
	MM (no $\boldsymbol{X}$)	9.728 ± 0.487	500 ± 0	1.2 ± 0.1	0.5 ± 0.0	0.0 ± 0.0
	Newton on $\boldsymbol{\theta}$ (no $\boldsymbol{X}$)	4.537 ± 0.729	14 ± 3	1.236 ± 0.132	0.5 ± 0.0	−0.08 ± 0.272
	Newton on $\boldsymbol{w}$	6.406 ± 2.104	14 ± 5	0.808 ± 0.462	0.844 ± 0.148	0.688 ± 0.296
	SLSQP	43.908 ± 24.469	229 ± 132	0.718 ± 0.206	0.796 ± 0.106	0.592 ± 0.211
Rank Partitioning	**SR-l2**	0.48 ± 0.24	4 ± 0	0.717 ± 0.207	0.837 ± 0.037	0.569 ± 0.072
	SR-l2-log	1.58 ± 2.027	29 ± 14	0.883 ± 0.208	0.699 ± 0.066	0.398 ± 0.132
	ILSR (no $\boldsymbol{X}$)	0.098 ± 0.056	2 ± 0	0.718 ± 0.208	0.708 ± 0.045	0.389 ± 0.088
	MM (no $\boldsymbol{X}$)	11.302 ± 0.515	500 ± 0	0.864 ± 0.19	0.685 ± 0.037	0.354 ± 0.074
	Newton on $\boldsymbol{\theta}$ (no $\boldsymbol{X}$)	8.218 ± 1.782	14 ± 3	1.244 ± 0.121	0.506 ± 0.029	0.01 ± 0.05
	Newton on $\boldsymbol{w}$	7.696 ± 2.35	14 ± 4	0.804 ± 0.463	0.871 ± 0.087	0.742 ± 0.173
	SLSQP	47.824 ± 28.585	219 ± 138	0.718 ± 0.206	0.819 ± 0.035	0.637 ± 0.07

0.0 KT. By regressing the Plackett-Luce scores from features, SR-l2 and SR-l2-log significantly outperform the predictions of ILSR, MM, and Newton on $\boldsymbol{\theta}$, by 16% − 33% Top-1 Acc. and 16% − 30% KT.

Real datasets. We observe an equally significant speed gain on real datasets; Table 4 shows the evaluations on four real datasets partitioned w.r.t. sample partitioning. SR-l2 and SR-l2-log are 3 − 18 times faster than Newton on $\boldsymbol{w}$ and SLSQP. This speed gain is fundamentally due to the smaller per iteration complexity of SR-l2 and SR-l2-log. For instance, compared to Newton on $\boldsymbol{w}$, SR-l2-log requires about 2 times more iterations, but still converges 3 times faster than Newton on $\boldsymbol{w}$ on FAC. Moreover, while significantly decreasing the convergence time, SR-l2 or SR-l2-log consistently attain similar prediction performance to Newton on $\boldsymbol{w}$ and SLSQP, except for Sushi, for which they perform slightly worse (by 13% − 20% Top-1 Acc.), though the convergence time dividends are striking in comparison (78% − 95%).

Aligned with the prediction performance on synthetic datasets, featureless methods ILSR, MM, and Newton on $\boldsymbol{\theta}$ can only attain 0.5 Top-1 Acc. and 0.0 KT. By regressing the Plackett-Luce scores from features, SR-l2 and SR-l2-log significantly outperform the predictions of ILSR, MM, and Newton on $\boldsymbol{\theta}$, by 3% − 31% Top-1 Acc. and 5% − 78% KT.

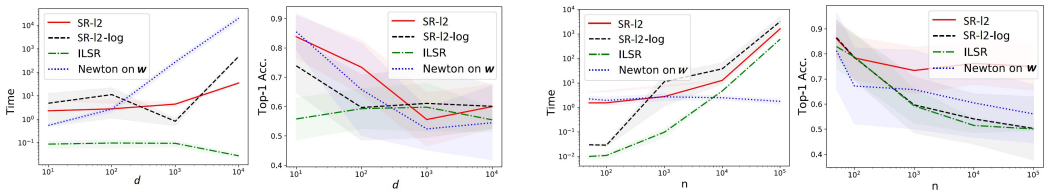
Rank partitioning. A sample can appear in both training and test observations in rank partitioning. Hence, featureless methods ILSR, MM, and Newton on $\boldsymbol{\theta}$ should fare better than in sample partitioning. Nonetheless, in Table 3, as $n = 1000$ is larger than $d = 100$, there are more scores to learn than parameters. As a result, feature methods are advantageous for good predictions compared to featureless methods. Particularly, SR-l2 and SR-l2-log outperform the predictions of ILSR, MM, and Newton on $\boldsymbol{\theta}$ in rank partitioning on Table 3, by 13% Top-1 Acc. and 9% KT. The relative performance of feature vs. featureless methods is governed by the relationship among n, d, and m. We do not include Newton on $\boldsymbol{\theta}$ and MM in this analysis, as they are too slow.

Impact of d. To assess the impact of number of parameters, we fix $n = 1000$, $m = 250$, $|A_\ell| = 2$ and generate synthetic datasets with $d \in \{10, 100, 1000, 10000\}$. Fig. 3a shows the Time and Top-1

Table 4. Evaluations on real datasets partitioned w.r.t. sample partitioning. We report the convergence time (Time), number of iterations until convergence (Iter), top-1 accuracy on the test set (Top-1 Acc.), and Kendall-Tau correlation on the test set (KT). ILSR, MM, and Newton on $\boldsymbol{\theta}$ do not use the features $\boldsymbol{X}$. Newton on $\boldsymbol{w}$ and SLSQP regress $\boldsymbol{\pi}$ from $\boldsymbol{X}$.

Dataset	Method	Training Metrics		Performance Metrics on the Test Set	
		Time (s) ↓	Iter. ↓	Top-1 Acc. ↑	KT ↑
FAC	**SR-l2**	0.301 ± 0.048	4 ± 0	0.654 ± 0.237	0.307 ± 0.473
	SR-l2-log	0.298 ± 0.466	10 ± 15	0.685 ± 0.237	0.369 ± 0.474
	ILSR (no $\boldsymbol{X}$)	0.059 ± 0.016	2 ± 0	0.5 ± 0.0	−1.0 ± 0.0
	MM (no $\boldsymbol{X}$)	5.905 ± 0.282	500 ± 0	0.5 ± 0.0	0.0 ± 0.0
	Newton on $\boldsymbol{\theta}$ (no $\boldsymbol{X}$)	7.604 ± 0.805	18 ± 2	0.5 ± 0.0	−0.4 ± 0.49
	Newton on $\boldsymbol{w}$	0.859 ± 0.077	6 ± 1	0.67 ± 0.17	0.34 ± 0.339
	SLSQP	14.332 ± 5.684	178 ± 67	0.675 ± 0.147	0.349 ± 0.293
ROP-num	**SR-l2**	1.708 ± 0.166	4 ± 0	0.783 ± 0.03	0.565 ± 0.06
	SR-l2-log	0.325 ± 0.028	1 ± 0	0.724 ± 0.105	0.448 ± 0.209
	ILSR (no $\boldsymbol{X}$)	0.649 ± 0.053	2 ± 0	0.5 ± 0.0	−1.0 ± 0.0
	MM (no $\boldsymbol{X}$)	0.001 ± 0.001	1 ± 0	0.5 ± 0.0	−1.0 ± 0.0
	Newton on $\boldsymbol{\theta}$ (no $\boldsymbol{X}$)	68.924 ± 5.521	8 ± 0	0.497 ± 0.012	−0.988 ± 0.036
	Newton on $\boldsymbol{w}$	47.563 ± 8.342	2 ± 1	0.552 ± 0.048	0.103 ± 0.096
	SLSQP	4.823 ± 4.914	2 ± 1	0.769 ± 0.052	0.538 ± 0.104
Pairwise Sushi	**SR-l2**	0.046 ± 0.01	4 ± 0	0.451 ± 0.082	−0.09 ± 0.177
	SR-l2-log	0.141 ± 0.025	27 ± 13	0.532 ± 0.076	0.064 ± 0.152
	ILSR (no $\boldsymbol{X}$)	0.014 ± 0.006	2 ± 0	0.5 ± 0.0	−1.0 ± 0.0
	MM (no $\boldsymbol{X}$)	1.513 ± 0.587	352 ± 183	0.5 ± 0.0	−1.0 ± 0.0
	Newton on $\boldsymbol{\theta}$ (no $\boldsymbol{X}$)	1.282 ± 0.924	18 ± 9	0.5 ± 0.0	−0.666 ± 0.472
	Newton on $\boldsymbol{w}$	0.21 ± 0.115	4 ± 2	0.665 ± 0.035	0.33 ± 0.069
	SLSQP	4.619 ± 6.321	168 ± 235	0.624 ± 0.065	0.248 ± 0.13
Triplet Sushi	**SR-l2**	0.091 ± 0.02	4 ± 0	0.358 ± 0.805	−0.333 ± 0.924
	SR-l2-log	0.556 ± 0.276	40 ± 25	0.393 ± 0.826	0.096 ± 1.069
	ILSR (no $\boldsymbol{X}$)	0.033 ± 0.012	2 ± 0	0.334 ± 0.0	−0.047 ± 1.089
	MM (no $\boldsymbol{X}$)	1.824 ± 0.701	267 ± 151	0.334 ± 0.0	0.0 ± 0.0
	Newton on $\boldsymbol{\theta}$ (no $\boldsymbol{X}$)	2.728 ± 1.475	13 ± 3	0.334 ± 0.0	−0.047 ± 1.089
	Newton on $\boldsymbol{w}$	1.966 ± 3.158	10 ± 19	0.322 ± 0.802	−0.261 ± 0.956
	SLSQP	1.656 ± 1.793	20 ± 30	0.608 ± 0.826	0.358 ± 0.928

Acc. of SR-l2, SR-l2-log, ILSR, and Newton on $\boldsymbol{w}$. As $m = 250$ observations are not enough to learn $n = 1000$ scores, SR-l2 leads to significantly better Top-1 Acc. compared to ILSR. When $d = 10$, SR-l2 and SR-l2-log outperform ILSR by 18% − 28% Top-1 Acc. Moreover, SR-l2 and SR-l2-log are consistently faster than Newton on $\boldsymbol{w}$, for all $d > 100$. Particularly, for $d = 10000$, SR-l2 and SR-l2-log converge *42-579 times faster than Newton on* $\boldsymbol{w}$. Interestingly, the convergence time of SR-l2-log can even decrease with increasing d. This is because the number of iterations until convergence decreases. While significantly decreasing the convergence time, SR-l2 consistently attains better Top-1 Acc. than Newton on $\boldsymbol{w}$, up to 8% for $d = 100$.



(a) Time and Top-1 Acc. vs. d (b) Time and Top-1 Acc. vs. n

Fig. 3. Convergence time (Time) and top-1 test accuracy (Top-1 Acc.) vs. n and d for SR-l2, SR-l2-log, ILSR, and Newton on $\boldsymbol{w}$. Evaluations are on synthetic datasets containing $m = 250$ observations partitioned w.r.t. rank partitioning. Number of samples changes in $n \in \{50, 100, 1000, 10000, 100000\}$ when number of features is $d = 100$, and number of features changes in $d \in \{10, 100, 1000, 10000\}$ when number of samples is $n = 1000$.

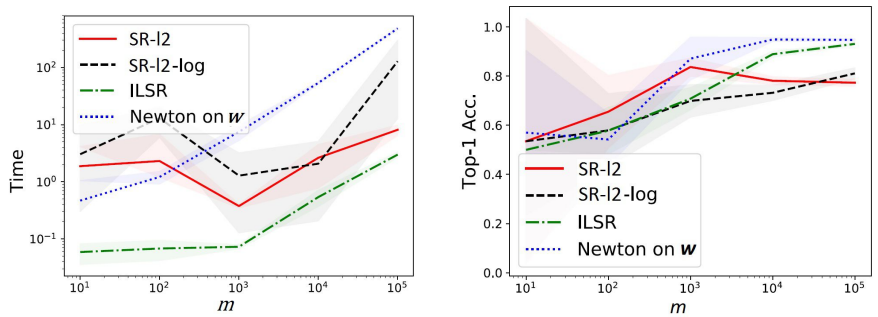


Fig. 4. Convergence time (Conv. Time) and top-1 test accuracy (Top-1 Acc.) of SR-l2, SR-l2-log, ILSR, and Newton on $\boldsymbol{w}$ evaluated on synthetic datasets vs. the number of observations $m \in \{10, 100, 1000, 10000, 100000\}$. Observations are partitioned w.r.t. rank partitioning, where number of samples is $n = 1000$, number of features is $d = 100$, and query size is $|A_\ell| = 2$.

Impact of n. To assess the impact of number of samples, we fix $d = 100$, $m = 250$, $|A_\ell| = 2$ and generate synthetic datasets with $n \in \{50, 100, 1000, 10000, 100000\}$; Fig. 3b shows evaluations on the resulting datasets. For $n > d = 100$, i.e., when there are more scores to learn than parameters, SR-l2 leads to significantly better Top-1 Acc. compared to ILSR. Particularly, for $n = 100000$, SR-l2 outperforms ILSR by 25% Top-1 Acc. This confirms that, especially when the number of observations m is not sufficient to learn n scores, exploiting the features associated with the samples is crucial in attaining good prediction performance. As expected, convergence time of Newton on $\boldsymbol{w}$ is not significantly affected by n. Despite this, SR-l2 and SR-l2-log are faster than Newton on $\boldsymbol{w}$ for all $n < 1000$. Particularly, for $n = 50$, SR-l2 and SR-l2-log converge *2-75 times faster than Newton on* $\boldsymbol{w}$. While decreasing the convergence time, SR-l2 consistently attains better Top-1 Acc. than Newton on $\boldsymbol{w}$, up to 19% for $n = 100000$.

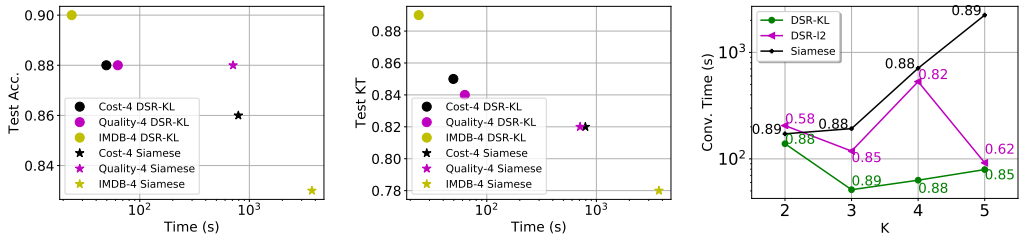
Impact of m. Fig. 4 shows the convergence time (Time) and top-1 test accuracy (Top-1 Acc.) of SR-l2, SR-l2-log, ILSR, and Newton on $\boldsymbol{w}$ when trained on synthetic datasets with number of observations $m \in \{10, 100, 1000, 10000, 100000\}$. Observations are partitioned w.r.t. rank partitioning, where number of samples is $n = 1000$, number of parameters is $d = 100$, and size of each query is $|A_\ell| = 2$. As $n > d$, SR-l2 benefits from being able to regress n scores from a smaller number of d parameters and leads to significantly better Top-1 Acc compared to ILSR in Fig. 4. Especially when m is not enough to learn $n = 1000$ scores, but to learn $d = 100$ parameters, SR-l2 gains the most performance

Table 5. Evaluations on real datasets partitioned w.r.t. rank partitioning. We report the convergence time in seconds (Time), number of iterations until convergence (Iter), top-1 accuracy on the test set (Top-1 Acc.), and Kendall-Tau correlation on the test set (KT). ILSR, MM, and Newton on $\boldsymbol{\theta}$ learn the Plackett-Luce scores $\boldsymbol{\pi}$ from the choice observations alone and do not use the features $\boldsymbol{X}$. Newton on $\boldsymbol{w}$ and sequential least squares quadratic programming (SLSQP) regress $\boldsymbol{\pi}$ from $\boldsymbol{X}$. (c.f. Sec. 4.3).

Dataset	Method	Training Metrics		Performance Metrics on the Test Set	
		Time (s) ↓	Iter. ↓	Top-1 Acc. ↑	KT ↑
FAC	**SR-l2**	0.352 ± 0.044	4 ± 0	0.68 ± 0.048	0.35 ± 0.089
	SR-l2-log	0.17 ± 0.033	4 ± 0	0.691 ± 0.054	0.378 ± 0.11
	ILSR (no $\boldsymbol{X}$)	0.066 ± 0.012	2 ± 0	0.591 ± 0.067	−0.13 ± 0.164
	MM (no $\boldsymbol{X}$)	10.7 ± 0.501	500 ± 0	0.544 ± 0.046	0.046 ± 0.087
	Newton on $\boldsymbol{\theta}$ (no $\boldsymbol{X}$)	9.152 ± 1.284	17 ± 3	0.5 ± 0.0	0.0 ± 0.0
	Newton on $\boldsymbol{w}$	1.531 ± 0.169	6 ± 1	0.701 ± 0.04	0.398 ± 0.08
	SLSQP	22.73 ± 19.151	160 ± 135	0.689 ± 0.063	0.375 ± 0.125
ROP-num	**SR-l2**	1.953 ± 0.217	4 ± 0	0.896 ± 0.005	0.791 ± 0.009
	SR-l2-log	0.359 ± 0.027	1 ± 0	0.904 ± 0.005	0.807 ± 0.01
	ILSR (no $\boldsymbol{X}$)	0.716 ± 0.058	2 ± 0	0.891 ± 0.005	0.781 ± 0.009
	MM (no $\boldsymbol{X}$)	356.497 ± 29.11	500 ± 0	0.905 ± 0.004	0.81 ± 0.008
	Newton on $\boldsymbol{\theta}$ (no $\boldsymbol{X}$)	85.42 ± 6.849	9 ± 0	0.906 ± 0.004	0.811 ± 0.008
	Newton on $\boldsymbol{w}$	55.718 ± 6.293	2 ± 0	0.904 ± 0.005	0.808 ± 0.009
	SLSQP	9.595 ± 7.136	2 ± 1	0.683 ± 0.049	0.366 ± 0.098
Pairwise Sushi	**SR-l2**	0.061 ± 0.002	4 ± 0	0.669 ± 0.034	0.338 ± 0.068
	SR-l2-log	0.764 ± 1.192	58 ± 30	0.634 ± 0.075	0.267 ± 0.15
	ILSR (no $\boldsymbol{X}$)	0.027 ± 0.003	2 ± 0	0.763 ± 0.039	0.521 ± 0.084
	MM (no $\boldsymbol{X}$)	5.191 ± 0.345	490 ± 31	0.773 ± 0.048	0.543 ± 0.094
	Newton on $\boldsymbol{\theta}$ (no $\boldsymbol{X}$)	2.342 ± 0.689	18 ± 5	0.735 ± 0.095	0.465 ± 0.185
	Newton on $\boldsymbol{w}$	0.176 ± 0.17	2 ± 2	0.685 ± 0.044	0.369 ± 0.087
	SLSQP	16.198 ± 8.728	245 ± 134	0.64 ± 0.06	0.28 ± 0.119
Triplet Sushi	**SR-l2**	0.127 ± 0.007	4 ± 0	0.569 ± 0.035	0.218 ± 0.045
	SR-l2-log	0.804 ± 0.349	36 ± 18	0.487 ± 0.034	0.19 ± 0.072
	ILSR (no $\boldsymbol{X}$)	0.054 ± 0.003	2 ± 0	0.678 ± 0.036	0.454 ± 0.06
	MM (no $\boldsymbol{X}$)	15.349 ± 0.617	500 ± 0	0.715 ± 0.035	0.522 ± 0.059
	Newton on $\boldsymbol{\theta}$ (no $\boldsymbol{X}$)	5.122 ± 0.34	14 ± 1	0.73 ± 0.036	0.496 ± 0.089
	Newton on $\boldsymbol{w}$	1.12 ± 0.659	3 ± 2	0.605 ± 0.058	0.285 ± 0.062
	SLSQP	21.738 ± 39.761	107 ± 197	0.521 ± 0.043	0.191 ± 0.059

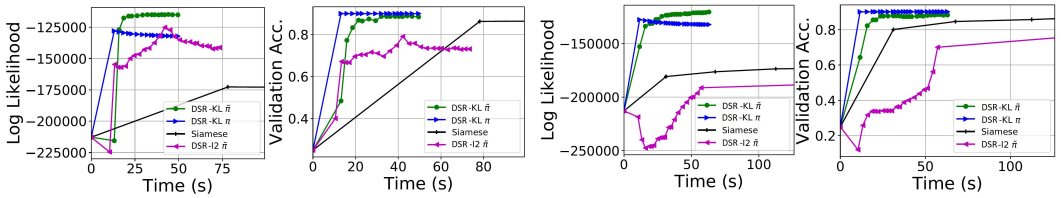
advantage over ILSR, up to 13% Top-1 Acc. Moreover, SR-l2 and SR-l2-log are consistently faster than Newton on $\boldsymbol{w}$, for all number of observations $m > 100$. Particularly, for $m = 100000$, SR-l2 and SR-l2-log converge $4 - 60$ times faster than Newton on $\boldsymbol{w}$.

Real datasets. Performance agrees with observations above regarding the dependence on n an d. For datasets where $n > m > d$, e.g., FAC, SR-l2 and SR-l2-log significantly outperform the predictions of ILSR, by 10% Top-1 Acc. and 25% KT. For datasets where m is much larger than n (c.f. Table 2), feature methods lead to similar prediction performance to each other and slightly lower performance than ILSR, MM, and Newton on $\boldsymbol{\theta}$. Overall, SR-l2 and SR-l2-log consistently converge faster than Newton on $\boldsymbol{w}$ and SLSQP, by $3 - 27$ times across all real datasets. Table 5 also confirms the observations from Fig. 3a and 3b regarding the effect of n and d. For FAC, $n = 1000 > m = 728 > d = 50$, i.e.,



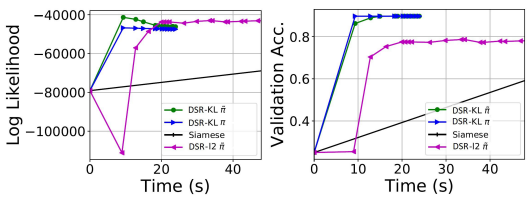
(a) Training time vs. Top-1 Acc. (b) Training time vs. KT (c) Query size (K) vs. performance

Fig. 5. **(a)-(b)**. Training time of DSR-KL and siamese network vs. Top-1 Acc. and KT on test sets of Movehub-Cost-4, Movehub-Quality-4, and IMDB-4 datasets, partitioned w.r.t. rank partitioning. **(c)**. Training time and prediction performances of DSR-KL, DSR-l2, and siamese network vs. query size (K) on Movehub-Quality dataset partitioned w.r.t. rank partitioning. Top-1 accuracy for each K is next to the corresponding marker.



(a) Convergence on Movehub-Cost-4

(b) Convergence on Movehub-Quality-4



(c) Convergence on IMDB-4

Dataset	Method	Performance on the Test Set	
		Top-1 Acc. ↑	KT ↑
Movehub-Cost-4	DSR-KL	**0.88**	**0.85**
	One-iter.-DSR-KL	0.67	0.53
Movehub-Quality-4	DSR-KL	**0.88**	**0.84**
	One-iter.-DSR-KL	0.75	0.73

(d) Predictions of DSR-KL vs. One-iter.-DSR-KL

Fig. 6. **(a)-(c).** Log-likelihood $-\mathcal{L}$ on training and Top-1 Acc. on validation sets of Movehub-Cost-4, Movehub-Quality-4, and IMDB-4, partitioned w.r.t. rank partitioning. Each point for DSR-KL and DSR-l2 correspond to an iteration of ADMM, while each point for siamese corresponds to a training epoch. **(d).** Test set predictions of DSR-KL vs. One-iter.-DSR-KL on Movehub-Cost-4 and Movehub-Quality-4.

there are enough observations to learn parameters, but not scores. As a result, SR-l2 and SR-l2-log significantly outperform the predictions of ILSR, by 10% Top-1 Acc. and 25% KT. For the other real datasets, m is much larger than n (c.f.Table2). Thus, feature methods lead to similar prediction performance to each other and lower performance than ILSR, MM, and Newton on $\boldsymbol{\theta}$.

Table 6. Training time vs. Top-1 Acc. and KT on test sets, partitioned w.r.t. rank and sample partitioning, respectively. We report averages and standard deviations over folds for all algorithms except for siamese. Our algorithms, as well as the algorithm that attains the best performance for each dataset are indicated in bold.

Dataset	Method	Rank Partitioning			Sample Partitioning		
		Time (s) ↓	Performance on the Test Set		Time (s) ↓	Performance on the Test Set	
			Top-1 Acc. ↑	KT ↑		Top-1 Acc. ↑	KT ↑
ICLR-3	**DSR-KL**	**152.86** ± 29.98	**0.9** ± 0.0	**0.86** ± 0.0	145.76 ± 9.78	0.48 ± 0.06	**0.28** ± 0.09
	DSR-l2	165.59 ± 22.63	0.79 ± 0.0	0.5 ± 0.0	**122.92** ± 75.43	**0.51** ± 0.02	0.07 ± 0.08
	Siamese	1445.77	0.88	0.8	827.05	0.48	0.05
	SR-KL	529.02 ± 117.26	0.37 ± 0.0	0.02 ± 0.0	96.49 ± 90.02	0.48 ± 0.0	0.0 ± 0.0
	SR-l2	20.59 ± 3.02	0.87 ± 0.0	0.82 ± 0.0	4.91 ± 4.6	0.47 ± 0.0	0.06 ± 0.0
Movehub-Cost-4	**DSR-KL**	**49.54** ± 34.2	**0.88** ± 0.07	**0.85** ± 0.09	**17.65** ± 7.68	**0.61** ± 0.07	0.45 ± 0.12
	DSR-l2	73.38 ± 32.82	0.72 ± 0.02	0.58 ± 0.03	19.85 ± 3.7	0.05 ± 0.08	-0.3 ± 0.22
	Siamese	523.64	0.87	0.82	216.22	0.6	0.66
	SR-KL	29.64 ± 3.59	0.47 ± 0.0	0.27 ± 0.0	7.45 ± 0.3	0.31 ± 0.0	0.44 ± 0.0
	SR-l2	19.7 ± 0.72	0.8 ± 0.0	0.54 ± 0.0	4.13 ± 0.2	0.5 ± 0.0	**0.71** ± 0.0
Movehub-Quality-4	**DSR-KL**	**63.05** ± 4.14	**0.88** ± 0.0	**0.84** ± 0.0	**31.41** ± 5.02	**0.88** ± 0.0	0.74 ± 0.05
	DSR-l2	531.86 ± 212.91	0.82 ± 0.07	0.75 ± 0.08	34.72 ± 8.68	0.67 ± 0.11	0.81 ± 0.31
	Siamese	710.35	0.88	0.82	258.87	0.88	**0.88**
	SR-KL	98.02 ± 3.73	0.17 ± 0.0	-0.1 ± 0.0	7.68 ± 0.2	0.48 ± 0.0	0.05 ± 0.0
	SR-l2	18.98 ± 4.11	0.84 ± 0.0	0.62 ± 0.0	4.69 ± 0.1	0.51 ± 0.0	0.55 ± 0.0
IMDB-4	**DSR-KL**	**23.93** ± 16.15	**0.9** ± 0.0	**0.9** ± 0.05	526.01 ± 11.57	**0.73** ± 0.29	-0.14 ± 0.25
	DSR-l2	54.16 ± 13.22	0.78 ± 0.035	0.38 ± 0.07	**27.73** ± 26.63	0.21 ± 0.21	-0.02 ± 0.08
	Siamese	3409.03	0.87	0.78	1240.98	0.47	0.02
	SR-KL	57.93 ± 0.87	0.16 ± 0.0	-0.04 ± 0.0	31.33 ± 13.26	0.04 ± 0.0	**0.05** ± 0.0
	SR-l2	9.43 ± 1.02	0.52 ± 0.0	0.08 ± 0.0	6.97 ± 2.71	0.6 ± 0.06	0.04 ± 0.06
ICLR-4	**DSR-KL**	84.93 ± 1.76	**0.88** ± 0.01	0.62 ± 0.06	288.45 ± 3.84	**0.48** ± 0.13	**0.06** ± 0.18
	DSR-l2	**84.78** ± 1.21	0.63 ± 0.01	0.19 ± 0.01	**280.72** ± 270.75	0.47 ± 0.02	0.032 ± 0.1
	Siamese	623.18	0.84	0.76	10142.55	0.32	-0.07
	SR-KL	347.29 ± 39.6	0.29 ±	0.0 ± 0.0	131.8 ± 120.01	0.45 ± 0.0	-0.07 ± 0.0
	SR-l2	503.11 ± 40.35	0.86 ±	**0.82** ± 0.0	198.49 ± 6.87	0.37 ± 0.03	-0.01 ± 0.0
ROP-img	**DSR-KL**	776.71 ± 136.74	**0.89** ± 0.0	**0.79** ± 0.0	**25.3** ± 15.4	0.8 ± 0.01	0.6 ± 0.02
	DSR-l2	**694.99** ± 431.12	0.84 ± 0.03	0.68 ± 0.064	210.45 ± 47.16	0.79 ± 0.01	0.59 ± 0.02
	Siamese	1152.06	0.86	0.73	4438.61	**0.82**	**0.65**
	SR-KL	610.91 ± 20.69	0.5 ± 0.0	0.0 ± 0.0	527.76 ± 163.76	0.61 ± 0.0	0.23 ± 0.0
	SR-l2	3.08 ± 0.59	0.89 ± 0.0	0.79 ± 0.0	1.96 ± 1.1	0.45 ± 0.0	-0.08 ± 0.0
Movehub-Cost-5	**DSR-KL**	**183.6** ± 48.2	0.85 ± 0.047	**0.84** ± 0.08	**111.13** ± 31.61	**0.76** ± 0.29	0.67 ± 0.15
	DSR-l2	216.5 ± 64.34	0.81 ± 0.04	0.69 ± 0.02	137.14 ± 65.55	0.19 ± 0.05	0.52 ± 0.34
	Siamese	7986.22	**0.89**	0.83	2842.12	0.62	**0.74**
	SR-KL	189.85 ± 0.72	0.45 ± 0.0	0.28 ± 0.0	30.51 ± 0.3	0.16 ± 0.0	0.39 ± 0.0
	SR-l2	209.5 ± 1.5	0.78 ± 0.0	0.55 ± 0.0	30.14 ± 0.2	0.37 ± 0.0	0.73 ± 0.0
Movehub-Quality-5	**DSR-KL**	**79.35** ± 2.91	0.85 ± 0.05	0.79 ± 0.08	114.03 ± 30.18	**0.92** ± 0.06	0.35 ± 0.25
	DSR-l2	91.87 ± 6.31	0.62 ± 0.04	0.5 ± 0.05	**46.99** ± 13.23	0.57 ± 0.34	**0.73** ± 0.46
	Siamese	2241.49	**0.89**	**0.83**	752.06	0.76	0.08
	SR-KL	924.07 ± 0.2	0.13 ± 0.0	-0.1 ± 0.0	29.68 ± 1.2	0.62 ± 0.0	0.08 ± 0.0
	SR-l2	208.1 ± 0.1	0.84 ± 0.0	0.65 ± 0.0	30.03 ± 1.1	0.39 ± 0.0	0.59 ± 0.0

4.9 DNN Regression Results

Training Time vs. Prediction Performance. Figure 5a and 5b show the training time of DSR-KL and siamese network vs. Top-1 Acc. and KT on test sets of Movehub-Cost-4, Movehub-Quality-4, and IMDB-4 datasets, partitioned with rank partitioning. For all datasets, DSR-KL results lie on the top left region of both Top-1 Acc. and KT plots: DSR-KL consistently leads to much *faster training and better predictions* than the siamese counterpart.

Columns 3-5 in Table 6 show the training time and test set prediction performance of DSR-KL, DSR-l2, siamese network, SR-l2, and SR-KL trained on datasets partitioned with rank partitioning. DSR-KL and DSR-l2 are $1.5 - 142$ *times faster* than siamese network over all datasets. Moreover, DSR-KL consistently attains equivalent or better prediction performance than both siamese network and DSR-l2 w.r.t. both Top-1 Acc. and KT. DSR-KL leads to particularly better performance in ranking predictions, by up to 6% higher KT than siamese and 25% higher KT than DSR-l2 on IMDB-4.

Deeper regression methods consistently outperform the predictions of shallow regression methods. Particularly, our deep spectral algorithms DSR-KL and DSR-l2 lead to significantly better predictions than the shallow versions SR-l2 and SR-KL, up to 38% Top-1 Acc. and 41% KT on IMDB-4. The training times of DSR-KL and DSR-l2 are also not noticeably larger than the ones of shallow versions; SR-KL converges even slower than DSR-KL, by up to 12 times on Movehub-Quality-5. Unlike SR-l2 that solves a least-squares problem with closed form solution, parameter update step of SR-KL (c.f. 48) requires an iterative optimization at each ADMM iteration.

Columns 6-8 in Table 6 show the training time and test set prediction performance of DSR-KL, DSR-l2, siamese network, SR-l2, and SR-KL trained on all datasets, partitioned with sample partitioning. Note that this is the setting when regressing scores from features is essential, as training samples cannot participate in any observations in validation or test sets. Agreeing with the speed gain in rank partitioning, DSR-KL and DSR-l2 are $8 - 175$ *times faster* than the siamese network over all datasets. Moreover, DSR-KL leads to particularly better performance in maximal-choice predictions, by up to 26% higher Top-1 Acc. than siamese on IMDB-4 and 56% higher Top-1 Acc. than DSR-l2 on Movehub-Cost-4. Finally, deeper regression methods DSR-KL, DSR-l2, and siamese network, outperform the predictions of shallow counterparts SR-l2 and SR-KL, by up to 39% Top-1 Acc. and 11% KT on Movehub-cost-5 and ICLR-3, respectively.

Impact of K. Figures 1 and 5c show training time and prediction performances of DSR-KL, DSR-l2, and siamese network vs. query size (K) on Movehub-Cost and Movehub-Quality datasets partitioned w.r.t. rank partitioning. The speed gain of DSR-KL and DSR-l2 over siamese reach up to 43 times as K increases, as each epoch of siamese grows exponentially with K. Moreover, agreeing with Table 6, DSR-KL consistently outperforms the predictions of DSR-l2.

Details on Convergence. Figure 6 shows the log-likelihood $-\mathcal{L}$ on training and Top-1 Acc. on validation sets of Movehub-Cost-4, Movehub-Quality-4, and IMDB-4 datasets, partitioned w.r.t. rank partitioning. Each point for DSR-KL and DSR-l2 correspond to an overall iteration of ADMM (c.f. (21)), while each point for siamese corresponds to a training epoch. DSR-KL consistently attains higher log-likelihood and better validation performance than both siamese network and DSR-l2, while converging faster.

Comparison to Naïve Approach. A naïve spectral algorithm can be constructed by a single iteration of the primal steps in (21): (i) solving (21a) to learn $\boldsymbol{\pi}$ via repeated iterations of (29), and (ii) given $\boldsymbol{\pi}$, solving (21b) by training the DNN regressor $\tilde{\boldsymbol{\pi}}(\boldsymbol{X};\boldsymbol{w})$ via SGD over Eq. (22). Intuitively, this ignores/does not exploit the fact that samples with similar features ought to have similar scores. We denote this naive approach as One-iter.-DSR-KL. Unlike One-iter.-DSR-KL, our algorithm DSR-KL has the capability of repeatedly adapting both $\boldsymbol{\pi}$ and $\boldsymbol{w}$ by solving (19) via ADMM. This advantage is

illustrated in Figures 6a and 6b, where not only $\tilde{\boldsymbol{\pi}}(\boldsymbol{X};\boldsymbol{w})$, but also $\boldsymbol{\pi}$ are adjusted until convergence. Moreover, Figure 6d shows the corresponding test set predictions of DSR-KL vs. One-iter.-DSR-KL on Movehub-Cost-4 and Movehub-Quality-4; DSR-KL outperforms One-iter.-DSR-KL by 13-21% Top-1 Acc. and 6-15% KT.

5 CONCLUSION

We solve the maximum likelihood estimation problem for the Plackett-Luce scores via ADMM. We show that the scores are equivalent to the stationary distribution of a Markov Chain and propose spectral algorithms for both shallow and deep neural network (DNN) models. Our resulting spectral algorithms significantly outperform the traditional Newton's method and siamese networks for ranking regression.

Given that the number of rankings grows exponentially in query size, designing active learning algorithms to identify which rankings to solicit from labelers is an interesting open problem. Generalizing existing active learning algorithms for shallow models, e.g. Guo et al. [2018], to deeper models via our efficient algorithms is a promising direction.

ACKNOWLEDGMENTS

Our work is supported by NIH (R01EY019474), NSF (SCH-1622542 at MGH, SCH-1622536 at Northeastern, SCH-1622679 at OHSU), a Facebook Statistics Research Award, and by unrestricted departmental funding from Research to Prevent Blindness (OHSU).

REFERENCES

Arpit Agarwal, Prathamesh Patil, and Shivani Agarwal. 2018. Accelerated spectral ranking. In *International Conference on Machine Learning*. 70–79.

Shivani Agarwal. 2016. On Ranking and Choice Models.. In *IJCAI*. 4050–4053.

Ammar Ammar and Devavrat Shah. 2011. Ranking: Compare, don't score. In *2011 49th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 776–783.

Esra Ataer-Cansızoğlu. 2015. *Retinal image analytics: A complete framework from segmentation to diagnosis*. Northeastern University.

Jose Blanchet, Guillermo Gallego, and Vineet Goyal. 2016. A markov chain approximation to choice modeling. *Operations Research* 64, 4 (2016), 886–905.

Blitzer. 2017. Movehub City Rankings. https://www.kaggle.com/blitzr/movehub-city-rankings?select=movehubqualityoflife.csv

Anna Sergeevna Bosman, Andries Engelbrecht, and Mardé Helbig. 2020. Visualising basins of attraction for the cross-entropy and the squared error neural network loss functions. *Neurocomputing* (2020).

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning* 3, 1 (2011), 1–122.

Stephen Boyd and Lieven Vandenberghe. 2004. *Convex optimization*. Cambridge university press.

Ralph Allan Bradley and Milton E Terry. 1952. Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons. *Biometrika* 39, 3/4 (1952), 324–345.

Mark Braverman and Elchanan Mossel. 2008. Noisy sorting without resampling. In *Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics, 268–276.

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

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1994. Signature verification using a "siamese" time delay neural network. In *Advances in Neural Information Processing Systems*. 737–744.

Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to rank using gradient descent. In *Proceedings of the International Conference on Machine Learning*. ACM, 89–96.

Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th International Conference on Machine learning*. 129–136.

Rich Caruana and Alexandru Niculescu-Mizil. 2004. Data mining in metric space: an empirical analysis of supervised learning performance criteria. In *Proceedings of the tenth ACM SIGKDD International Conference on Knowledge Discovery*

and Data Mining. 69–78.

Manuela Cattelan. 2012. Models for paired comparison data: A review with emphasis on dependent data. *Statist. Sci.* (2012), 412–433.

Huiwen Chang, Fisher Yu, Jue Wang, Douglas Ashley, and Adam Finkelstein. 2016. Automatic triage for a photo series. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 148.

Rick Chartrand and Brendt Wohlberg. 2013. A nonconvex ADMM algorithm for group sparsity with sparse groups. In *IEEE International Conference on Acoustics, Speech and Signal Processing*. 6009–6013.

Lin Chen, Peng Zhang, and Baoxin Li. 2015. Fusing Pointwise and Pairwise Labels for Supporting User-adaptive Image Retrieval. In *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*. ACM, 67–74.

Yuxin Chen and Changho Suh. 2015. Spectral mle: Top-k rank aggregation from pairwise comparisons. In *International Conference on Machine Learning*. 371–380.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 248–255.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 4171–4186.

Hazel Doughty, Dima Damen, and Walterio Mayol-Cuevas. 2018. Who's better? Who's best? Pairwise deep ranking for skill determination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 6057–6066.

Abhimanyu Dubey, Nikhil Naik, Devi Parikh, Ramesh Raskar, and César A Hidalgo. 2016. Deep learning the city: Quantifying urban perception at a global scale. In *European Conference on Computer Vision*. Springer, 196–212.

Cynthia Dwork, Ravi Kumar, Moni Naor, and Dandapani Sivakumar. 2001. Rank aggregation methods for the web. In *Proceedings of the 10th International Conference on World Wide Web*. ACM, 613–622.

Otto Dykstra. 1960. Rank analysis of incomplete block designs: A method of paired comparisons employing unequal repetitions on pairs. *Biometrics* 16, 2 (1960), 176–188.

Arpad E Elo. 1978. *The rating of chessplayers, past and present*. Arco Pub.

Michael A Fligner and Joseph S Verducci. 1993. *Probability models and statistical analyses for ranking data*. Vol. 80. Springer.

Robert G Gallager. 2013. *Stochastic Processes: Theory for Applications*. Cambridge University Press.

Pavel Golik, Patrick Doetsch, and Hermann Ney. 2013. Cross-entropy vs. squared error training: a theoretical and experimental comparison.. In *Interspeech*, Vol. 13. 1756–1760.

Ke Guo, DR Han, and Ting-Ting Wu. 2017. Convergence of alternating direction method for minimizing sum of two nonconvex functions with linear constraints. *International Journal of Computer Mathematics* 94, 8 (2017), 1653–1669.

Yuan Guo, Peng Tian, Jayashree Kalpathy-Cramer, Susan Ostmo, J Peter Campbell, Michael F Chiang, Deniz Erdoğmuş, Jennifer G Dy, and Stratis Ioannidis. 2018. Experimental Design under the Bradley-Terry Model.. In *IJCAI*. 2198–2204.

Bruce Hajek, Sewoong Oh, and Jiaming Xu. 2014. Minimax-optimal inference from partial rankings. In *Advances in Neural Information Processing Systems*. 1475–1483.

Bo Han. 2018. DATELINE: Deep Plackett-Luce Model with Uncertainty Measurements. *arXiv preprint arXiv:1812.05877* (2018).

Mingyi Hong. 2018. A Distributed, Asynchronous, and Incremental Algorithm for Nonconvex Optimization: An ADMM Approach. *IEEE Transactions on Control of Network Systems* 5, 3 (2018), 935–945.

Roger A Horn and Charles R Johnson. 2012. *Matrix Analysis*. Cambridge university press.

David R Hunter. 2004. MM algorithms for generalized Bradley-Terry models. *The Annals of Statistics* 32, 1 (2004), 384–406.

Prateek Jain and Purushottam Kar. 2017. Non-convex Optimization for Machine Learning. *Foundations and Trends® in Machine Learning* 10, 3-4 (2017), 142–336.

Minje Jang, Sunghyun Kim, Changho Suh, and Sewoong Oh. 2017. Optimal sample complexity of m-wise data for top-k ranking. In *Advances in Neural Information Processing Systems*. 1686–1696.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 133–142.

Ian T Jolliffe. 1986. Principal component analysis and factor analysis. In *Principal Component Analysis*. Springer.

T. Kamishima, M. Hamasaki, and S. Akaho. 2009. A simple transfer learning method and its application to personalization in collaborative tagging. In *ICDM*.

Maurice G Kendall. 1938. A new measure of rank correlation. *Biometrika* 30, 1/2 (1938), 81–93.

Ashish Khetan and Sewoong Oh. 2016. Computational and statistical tradeoffs in learning to rank. In *Advances in Neural Information Processing Systems*. 739–747.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR (Poster)*.

Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The Annals of Mathematical Statistics* 22, 1 (1951), 79–86.

Qi Lei, Kai Zhong, and Inderjit S Dhillon. 2016. Coordinate-wise power method. In *Advances in Neural Information Processing Systems*. 2064–2072.

Orges Leka. 2016. IMDB Movies Dataset. https://www.kaggle.com/orgesleka/imdbmovies

Y. Li, X. Cheng, and G. Gui. 2018. Co-Robust-ADMM-Net: Joint ADMM Framework and DNN for Robust Sparse Composite Regularization. *IEEE Access* 6 (2018), 47943–47952.

R. Liu, Z. Jiang, X. Fan, H. Li, and Z. Luo. 2018. Single Image Layer Separation via Deep ADMM Unrolling. In *2018 IEEE International Conference on Multimedia and Expo*. 1–6.

Tyler Lu and Craig Boutilier. 2011. Learning Mallows models with pairwise preferences. In *Proceedings of the 28th International Conference on Machine Learning (icml-11)*. 145–152.

Jiawei Ma, Xiao-Yang Liu, Zheng Shou, and Xin Yuan. 2019. Deep tensor ADMM-net for snapshot compressive imaging. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 10223–10232.

Jiaqi Ma, Xinyang Yi, Weijing Tang, Zhe Zhao, Lichan Hong, Ed H Chi, and Qiaozhu Mei. 2020. Learning-to-Rank with Partitioned Preference: Fast Estimation for the Plackett-Luce Model. *arXiv preprint arXiv:2006.05067* (2020).

Sindri Magnússon, Pradeep Chathuranga Weeraddana, Michael G Rabbat, and Carlo Fischione. 2015. On the convergence of alternating direction lagrangian methods for nonconvex structured optimization problems. *IEEE Transactions on Control of Network Systems* 3, 3 (2015), 296–309.

Colin L Mallows. 1957. Non-null ranking models. I. *Biometrika* 44, 1/2 (1957), 114–130.

Cheng Mao, Ashwin Pananjady, and Martin J Wainwright. 2018a. Breaking the $1/\sqrt{n}$ Barrier: Faster Rates for Permutation-based Models in Polynomial Time. In *Conference On Learning Theory*. 2037–2042.

Cheng Mao, Jonathan Weed, and Philippe Rigollet. 2018b. Minimax rates and efficient algorithms for noisy sorting. In *Algorithmic Learning Theory*. PMLR, 821–847.

John I Marden. 2014. *Analyzing and modeling rank data*. Chapman and Hall/CRC.

Lucas Maystre and Matthias Grossglauser. 2015. Fast and Accurate Inference of Plackett-Luce Models. In *Advances in Neural Information Processing Systems*. 172–180.

Daniel McFadden. 1973. Conditional logit analysis of qualitative choice behavior. (1973).

Sahand Negahban, Sewoong Oh, and Devavrat Shah. 2012. Iterative ranking from pair-wise comparisons. In *Advances in Neural Information Processing Systems*. 2474–2482.

Sahand Negahban, Sewoong Oh, Kiran K Thekumparampil, and Jiaming Xu. 2018. Learning from comparisons and choices. *The Journal of Machine Learning Research* 19, 1 (2018), 1478–1572.

Jorge Nocedal and Stephen Wright. 2006. *Numerical optimization*. Springer Science & Business Media.

Tapio Pahikkala, Evgeni Tsivtsivadze, Antti Airola, Jouni Järvinen, and Jorma Boberg. 2009. An efficient algorithm for learning to rank from preference graphs. *Machine Learning* 75, 1 (2009), 129–165.

Robin L Plackett. 1975. The analysis of permutations. *Applied Statistics* (1975), 193–202.

Stephen Ragain and Johan Ugander. 2016. Pairwise choice Markov chains. In *Advances in Neural Information Processing Systems*. 3198–3206.

Arun Rajkumar and Shivani Agarwal. 2014. A statistical convergence perspective of algorithms for rank aggregation from pairwise data. In *International Conference on Machine Learning*. 118–126.

Arun Rajkumar and Shivani Agarwal. 2016. When can we rank well from comparisons of O (n\log (n)) non-actively chosen pairs?. In *Conference on Learning Theory*. 1376–1401.

Garrett van Ryzin and Siddharth Mahajan. 1999. On the relationship between inventory costs and variety benefits in retail assortments. *Management Science* 45, 11 (1999), 1496–1509.

Aadirupa Saha and Arun Rajkumar. 2018. Ranking with Features: Algorithm and A Graph Theoretic Analysis. *arXiv preprint arXiv:1808.03857* (2018).

David Sculley. 2010. Combined regression and ranking. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 979–988.

Nihar Shah, Sivaraman Balakrishnan, Aditya Guntuboyina, and Martin Wainwright. 2016b. Stochastically transitive models for pairwise comparisons: Statistical and computational issues. In *International Conference on Machine Learning*. 11–20.

Nihar B Shah, Sivaraman Balakrishnan, Joseph Bradley, Abhay Parekh, Kannan Ramchandran, and Martin J Wainwright. 2016a. Estimation from pairwise comparisons: Sharp minimax bounds with topology dependence. *The Journal of Machine Learning Research* 17, 1 (2016), 2049–2095.

Zhan Shi, Xinhua Zhang, and Yaoliang Yu. 2017. Bregman divergence for stochastic variance reduction: saddle-point and adversarial prediction. In *Advances in Neural Information Processing Systems (NeurIPS)*. 6031–6041.

Hossein Azari Soufiani, William Chen, David C Parkes, and Lirong Xia. 2013. Generalized method-of-moments for rank aggregation. In *Advances in Neural Information Processing Systems*. 2706–2714.

Jian Sun, Huibin Li, Zongben Xu, et al. 2016. Deep ADMM-Net for compressive sensing MRI. In *Advances in Neural Information Processing Systems (NeurIPS)*. 10–18.

Shao-Hua Sun. 2020. Crawl and Visualize ICLR 2020 OpenReview Data. https://github.com/shaohua0116/ICLR2020-OpenReviewData.

Wei-Tse Sun, Ting-Hsuan Chao, Yin-Hsi Kuo, and Winston H Hsu. 2017. Photo filter recommendation by category-aware aesthetic learning. *IEEE Transactions on Multimedia* 19, 8 (2017), 1870–1880.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, et al. 2015. Going deeper with convolutions. In *Computer Vision and Pattern Recognition, 2015*.

Mahsa Taheri, Fang Xie, and Johannes Lederer. 2021. Statistical guarantees for regularized neural networks. *Neural Networks* 142 (2021), 148–161.

Louis L Thurstone. 1927. The method of paired comparisons for social values. *The Journal of Abnormal and Social Psychology* 21, 4 (1927), 384.

Peng Tian, Yuan Guo, Jayashree Kalpathy-Cramer, Susan Ostmo, John Peter Campbell, Michael F Chiang, Jennifer Dy, Deniz Erdoğmuş, and Stratis Ioannidis. 2019. A Severity Score for Retinopathy of Prematurity. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1809–1819.

Milan Vojnovic and Seyoung Yun. 2016. Parameter estimation for generalized thurstone choice models. In *International Conference on Machine Learning*. 498–506.

Milan Vojnovic, Se-Young Yun, and Kaifang Zhou. 2020. Convergence Rates of Gradient Descent and MM Algorithms for Bradley-Terry Models. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 1254–1264.

Fenghui Wang, Wenfei Cao, and Zongben Xu. 2018. Convergence of multi-block Bregman ADMM for nonconvex composite problems. *Science China Information Sciences* 61, 12 (2018), 122101.

Huahua Wang and Arindam Banerjee. 2014. Bregman Alternating Direction Method of Multipliers. In *Advances in Neural Information Processing Systems (NeurIPS)*. 2816–2824.

Yu Wang, Wotao Yin, and Jinshan Zeng. 2019. Global convergence of ADMM in nonconvex nonsmooth optimization. *Journal of Scientific Computing* 78, 1 (2019), 29–63.

Fabian Wauthier, Michael Jordan, and Nebojsa Jojic. 2013. Efficient ranking from pairwise comparisons. In *International Conference on Machine Learning*. 109–117.

Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li. 2008. Listwise approach to learning to rank: theory and algorithm. In *Proceedings of the 25th international conference on Machine learning*. 1192–1199.

Y. Yang, J. Sun, H. Li, and Z. Xu. 2020. ADMM-CSNet: A Deep Learning Approach for Image Compressive Sensing. *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI* 42, 3 (2020), 521–538.

Shaokai Ye, Xiaoyu Feng, Tianyun Zhang, Xiaolong Ma, Sheng Lin, Zhengang Li, Kaidi Xu, Wujie Wen, Sijia Liu, Jian Tang, Makan Fardad, Xue Lin, Yongpan Liu, and Yanzhi Wang. 2019. Progressive DNN Compression: A Key to Achieve Ultra-High Weight Pruning and Quantization Rates using ADMM. arXiv:1903.09769 [cs.NE]

Shaokai Ye, Tianyun Zhang, Kaiqi Zhang, Jiayu Li, Jiaming Xie, Yun Liang, Sijia Liu, Xue Lin, and Yanzhi Wang. 2018. A Unified Framework of DNN Weight Pruning and Weight Clustering/Quantization Using ADMM. arXiv:1811.01907 [cs.NE]

İlkay Yıldız, Jennifer Dy, Deniz Erdoğmuş, Jayashree Kalpathy-Cramer, Susan Ostmo, J Peter Campbell, Michael F Chiang, and Stratis Ioannidis. 2020. Fast and Accurate Ranking Regression. In *International Conference on Artificial Intelligence and Statistics*.

Ilkay Yıldız, Jennifer Dy, Deniz Erdoğmuş, Susan Ostmo, J Peter Campbell, Michael F Chiang, and Stratis Ioannidis. 2021. Deep Spectral Ranking. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 361–369.

İlkay Yıldız, Peng Tian, Jennifer Dy, Deniz Erdoğmuş, James Brown, Jayashree Kalpathy-Cramer, Susan Ostmo, J Peter Campbell, Michael F Chiang, and Stratis Ioannidis. 2019. Classification and comparison via neural networks. *Neural Networks* (2019).

Yue Yu and Behçet Açıkmeşe. 2019. Stochastic Bregman parallel direction method of multipliers for distributed optimization. In *2019 IEEE 58th Conference on Decision and Control (CDC)*. IEEE, 5550–5555.

Yue Yu, Behçet Açıkmeşe, and Mehran Mesbahi. 2018. Bregman parallel direction method of multipliers for distributed optimization via mirror averaging. *IEEE Control Systems Letters* 2, 2 (2018), 302–306.

Jinshan Zeng, Shikang Ouyang, Tim Tsz-Kit Lau, Shaobo Lin, and Yuan Yao. 2018. Global convergence in deep learning with variable splitting via the Kurdyka-Lojasiewicz property. *arXiv preprint arXiv:1803.00225* (2018).

Haimeng Zhao and Peiyuan Liao. 2019. CAE-ADMM: Implicit Bitrate Optimization via ADMM-based Pruning in Compressive Autoencoders. arXiv:1901.07196 [cs.CV]

Pu Zhao, Sijia Liu, Yanzhi Wang, and Xue Lin. 2018. An ADMM-Based Universal Framework for Adversarial Attacks on Deep Neural Networks. In *Proceedings of the 26th ACM International Conference on Multimedia* (Seoul, Republic of Korea). Association for Computing Machinery, New York, NY, USA, 1065–1073. https://doi.org/10.1145/3240508.3240639

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE International Conference on Computer Vision*. 19–27.

APPENDICES

In this section, we provide proofs for all theoretical results established in Sections 2 and 3, as well as a brief technical preliminary on finite-state homogeneous Markov Chains.

A FINITE-STATE HOMOGENEOUS MARKOV CHAINS

We briefly review finite-state homogeneous Markov Chains due to the recent emergence of spectral algorithms for Plackett-Luce inference. All the results mentioned in this section are classic; our main reference is the book by Gallager [2013]. A finite-state Markov Chain is a sequence of random variables $\{Z_1, Z_2, \cdots\}$, each taking values from a finite-state sample space $[n] \equiv \{1, \ldots, n\}$, that satisfies the Markov property:

$$\mathbf{P}(Z_t|Z_{1:t-1}) = \mathbf{P}(Z_t|Z_{t-1}), \quad \text{for all } t. \tag{53}$$

A homogeneous Markov Chain additionally satisfies: $\lambda_{ij} = \mathbf{P}(Z_t = j|Z_{t-1} = i) = \mathbf{P}(Z_{t-1} = j|Z_{t-2} = i) = \cdots = \mathbf{P}(Z_2 = j|Z_1 = i)$. In this case, the Markov Chain can be represented by a directed graph, in which there is a node for each $i \in [n]$, and λ_{ij} denotes the edge weight on the transition from node $i \in [n]$ to $j \in [n]$. Particularly, λ_{ij} for all (i, j) are called the transition rates, and the corresponding matrix $\mathbf{\Lambda} = [\lambda_{ij}]_{i,j\in[n]}$ is called the transition matrix of the Markov Chain.

If the transition matrix $\mathbf{\Lambda}$, and accordingly the corresponding directed graph, is irreducible, the Markov chain is said to be ergodic and admits a unique stationary distribution $\boldsymbol{\pi} \in \mathbb{R}^n_+$, that satisfies the so-called global *balance equations*:

$$\textstyle\sum_{j\neq i} \pi_j \lambda_{ji} = \sum_{j\neq i} \pi_i \lambda_{ij}, \quad \text{for all } i \in [n]\,. \tag{54}$$

$\boldsymbol{\pi}$ is the unique solution to the linear system defined by these balance equations and $\mathbf{1}^\top \boldsymbol{\pi} = 1$, as it is a distribution. This follows from the Perron-Frobenius Theorem [Horn and Johnson 2012]. In practice, the stationary distribution $\boldsymbol{\pi}$ can be computed by uniformizing $\mathbf{\Lambda}$, i.e., increasing self-transition rates until all states have the same outgoing rate, and finding the leading left eigenvector via, e.g., the power method [Lei et al. 2016]:

$$\boldsymbol{\pi}^{l+1} = \frac{\mathbf{\Lambda}\boldsymbol{\pi}^l}{\| \mathbf{\Lambda}\boldsymbol{\pi}^l \|_2}, \quad \text{for } l \in \mathbb{N}. \tag{55}$$

B PROOF OF THEOREM 2.1

We start by showing that $\frac{\partial \mathcal{L}(\mathcal{D}\,|\,\boldsymbol{\pi})}{\partial \pi_i} = 0$, $i \in [n]$ is the optimality condition to minimize Eq. (15). Consider the reparametrization $\pi_i = e^{\theta_i}$, $i \in [n]$. Eq. (15) under this reparametrization is given by:

$$\mathcal{L}(\mathcal{D}\,|\,\boldsymbol{\theta}) = \sum_{\ell=1}^{n} \left(\log \sum_{j\in A_\ell} e^{\theta_j} - \theta_\ell \right), \tag{56}$$

which is convex w.r.t. $\boldsymbol{\theta} = [\theta_i]_{i\in[n]}$, i.e., even though Eq. (15) is not convex w.r.t. $\boldsymbol{\pi}$, it is convex under the reparametrization $\pi_i = e^{\theta_i}$, $i \in [n]$. This implies that $\frac{\partial \mathcal{L}(\mathcal{D}\,|\,\boldsymbol{\theta})}{\partial \theta_i} = 0$, $i \in [n]$ is the optimality condition to minimize Eq. (56) w.r.t. $\boldsymbol{\theta}$. By the chain rule, this condition can be written in terms of $\pi_i = e^{\theta_i}$, $i \in [n]$ as:

$$\frac{\partial \mathcal{L}(\mathcal{D}\,|\,\boldsymbol{\theta})}{\partial \theta_i} = \frac{\partial \mathcal{L}(\mathcal{D}\,|\,\boldsymbol{\pi})}{\partial \pi_i} e^{\theta_i} = 0 \quad \forall i \in [n]\,. \tag{57}$$

Note that $e^{\theta_i} > 0$, $i \in [n]$. Then, $\frac{\partial \mathcal{L}(\mathcal{D}\,|\,\boldsymbol{\theta})}{\partial \theta_i} = 0$ is equivalent to $\frac{\partial \mathcal{L}(\mathcal{D}\,|\,\boldsymbol{\pi})}{\partial \pi_i} = 0$, $i \in [n]$, i.e., $\pi_i = e^{\theta_i}$, $i \in [n]$ satisfies Eq.(57) if and only if θ_i, $i \in [n]$ is the minimizer of Eq. (56). Hence, the stationarity condition $\frac{\partial \mathcal{L}(\mathcal{D}\,|\,\boldsymbol{\pi})}{\partial \pi_i} = 0$, $i \in [n]$ is also the optimality condition for problem (3).

The optimality condition is given explicitly by:

$$\frac{\partial \mathcal{L}}{\partial \pi_i} = \sum_{\ell \in W_i} \left(\frac{1}{\sum_{t \in A_\ell} \pi_t} - \frac{1}{\pi_i} \right) + \sum_{\ell \in L_i} \frac{1}{\sum_{t \in A_\ell} \pi_t} = 0 \quad \forall i \in [n] \,, \tag{58}$$

where $W_i = \{\ell \,|i \in A_\ell, c_\ell = i\}$ is the set of observations where sample $i \in [n]$ is chosen and $L_i = \{\ell \,|i \in A_\ell, c_\ell \neq i\}$ is the set of observations where sample $i \in [n]$ is not chosen. Multiplying both sides of Eq. (58) with π_i, $i \in [n]$, we have:

$$\sum_{\ell \in L_i} \left(\frac{\pi_i}{\sum_{t \in A_\ell} \pi_t} \right) - \sum_{\ell \in W_i} \left(\frac{\sum_{j \neq i \in A_\ell} \pi_j}{\sum_{t \in A_\ell} \pi_t} \right) = 0, \tag{59}$$

for all $i \in [n]$. Note that $\sum_{\ell \in W_i} \sum_{j \neq i \in A_\ell} \cdot = \sum_{j \neq i} \sum_{\ell \in W_i \cap L_j} \cdot$ and $\sum_{\ell \in L_i} \cdot = \sum_{j \neq i} \sum_{\ell \in W_j \cap L_i} \cdot$. Accordingly, we rewrite Eq. (59) as:

$$\sum_{j \neq i} \sum_{\ell \in W_j \cap L_i} \left(\frac{\pi_i}{\sum_{t \in A_\ell} \pi_t} \right) = \sum_{j \neq i} \sum_{\ell \in W_i \cap L_j} \left(\frac{\pi_j}{\sum_{t \in A_\ell} \pi_t} \right) \quad \forall i \in [n] \,. \tag{60}$$

Then, an optimal solution $\boldsymbol{\pi} \in \mathbb{R}^n_+$ to Eq. (3) satisfies:

$$\sum_{j \neq i} \pi_j \lambda_{ji}(\boldsymbol{\pi}) = \sum_{j \neq i} \pi_i \lambda_{ij}(\boldsymbol{\pi}) \quad \forall i \in [n] \,, \tag{61}$$

where $\lambda_{ji}(\boldsymbol{\pi})$, $i, j \in [n] \,, i \neq j$ are given by Eq. (5).

C PROOF OF LEMMA 3.1

Let a stationary point $\boldsymbol{\pi} \in \mathbb{R}^n_+$ of the Augmented Lagrangian (20) be such that:

$$\frac{\partial L_\rho(\boldsymbol{\pi}, \boldsymbol{w}^k, \boldsymbol{y}^k)}{\partial \pi_i} = 0 \Leftrightarrow \frac{\partial \mathcal{L}(\mathcal{D}|\boldsymbol{\pi})}{\partial \pi_i} + y_i^k + \rho \frac{\partial D_p(\boldsymbol{\pi}||\tilde{\boldsymbol{\pi}}^k)}{\partial \pi_i} = 0. \ \forall i \in [n] \tag{62a}$$

Let $\sigma_i(\boldsymbol{\pi}) = \rho \frac{\partial D_p(\boldsymbol{\pi}||\tilde{\boldsymbol{\pi}}^k)}{\partial \pi_i} + y_i^k$, for all $i \in [n]$. Then, Eq.(62a) is equivalent to:

$$\frac{\partial \mathcal{L}(\mathcal{D}|\boldsymbol{\pi})}{\partial \pi_i} + \sigma_i(\boldsymbol{\pi}) = 0 \ \ \forall i \in [n] \,. \tag{63}$$

Partial derivatives of the negative log-likelihood $\mathcal{L}(\mathcal{D}|\boldsymbol{\pi})$ are given by:

$$\frac{\partial \mathcal{L}(\mathcal{D}|\boldsymbol{\pi})}{\partial \pi_i} = \sum_{\ell \in W_i} \left(\frac{1}{\sum_{t \in A_\ell} \pi_t} - \frac{1}{\pi_i} \right) + \sum_{\ell \in L_i} \frac{1}{\sum_{t \in A_\ell} \pi_t} \ \ \forall i \in [n] \,, \tag{64}$$

where $W_i = \{\ell \,|i \in A_\ell, c_\ell = i\}$ is the set of observations where sample $i \in [n]$ is chosen and $L_i = \{\ell \,|i \in A_\ell, c_\ell \neq i\}$ is the set of observations where sample $i \in [n]$ is not chosen. Setting $\frac{\partial \mathcal{L}(\mathcal{D}|\boldsymbol{\pi})}{\partial \pi_i}$ from Eq. (64) to Eq. (63), we have:

$$\frac{\partial L_\rho(\boldsymbol{\pi}, \boldsymbol{w}^k, \boldsymbol{y}^k)}{\partial \pi_i} = \sum_{\ell \in W_i} \left(\frac{1}{\sum_{t \in A_\ell} \pi_t} - \frac{1}{\pi_i} \right) + \sum_{\ell \in L_i} \frac{1}{\sum_{t \in A_\ell} \pi_t} + \sigma_i(\boldsymbol{\pi}) = 0, \ \ \forall i \in [n] \,. \tag{65}$$

Multiplying both sides of Eq. (65) with $-\pi_i$, $i \in [n]$, we have:

$$\sum_{\ell \in W_i} \left(\frac{\sum_{j \neq i \in A_\ell} \pi_j}{\sum_{t \in A_\ell} \pi_t} \right) - \sum_{\ell \in L_i} \left(\frac{\pi_i}{\sum_{t \in A_\ell} \pi_t} \right) - \pi_i \sigma_i(\boldsymbol{\pi}) = 0 \ \ \forall i \in [n] \,. \tag{66}$$

Note that $\sum_{\ell \in W_i} \sum_{j \neq i \in A_\ell} \cdot = \sum_{j \neq i} \sum_{\ell \in W_i \cap L_j} \cdot$ and $\sum_{\ell \in L_i} \cdot = \sum_{j \neq i} \sum_{\ell \in W_j \cap L_i} \cdot$. Accordingly, we rewrite Eq. (66) as:

$$\sum_{j \neq i} \sum_{\ell \in W_i \cap L_j} \left(\frac{\pi_j}{\sum_{t \in A_\ell} \pi_t} \right) - \sum_{j \neq i} \sum_{\ell \in W_j \cap L_i} \left(\frac{\pi_i}{\sum_{t \in A_\ell} \pi_t} \right) - \pi_i \sigma_i(\boldsymbol{\pi}) = 0 \ \ \forall i \in [n] \,. \tag{67}$$

Then, the stationarity condition given by Eq.(62a) is equivalent to:

$$\sum_{j \neq i} \pi_j \lambda_{ji}(\boldsymbol{\pi}) - \sum_{j \neq i} \pi_i \lambda_{ij}(\boldsymbol{\pi}) = \pi_i \sigma_i(\boldsymbol{\pi}) \ \ \forall i \in [n] \,, \tag{68}$$

where $\lambda_{ji}(\boldsymbol{\pi})$, $i, j \in [n] \,, i \neq j$ are given by Eq. (5).

D PROOF OF THEOREM 3.2

Summing Eq. (26) for $i \in [n]$, we get:

$$\sum_i \sum_j \left(\pi_j \lambda_{ji}(\boldsymbol{\pi}) - \pi_i \lambda_{ij}(\boldsymbol{\pi}) \right) \mathbb{1}_{j \neq i} = \sum_i \pi_i \sigma_i = 0. \tag{69}$$

Since the Plackett-Luce scores are non-negative, i.e. $\pi_i \geq 0$, $i \in [n]$, Eq. (69) implies that $\boldsymbol{\sigma} \equiv [\sigma_i]_{i \in [n]}$ contains both positive and negative elements. Let $([n]_+, [n]_-)$ be a partition of $[n]$ such that $\sigma_i \geq 0$ for all $i \in [n]_+$ and $\sigma_i < 0$ for all $i \in [n]_-$. Then, for $i \in [n]_+$ in Eq. (26), we have:

$$\sum_{j \neq i} \pi_j \lambda_{ji}(\boldsymbol{\pi}) = \pi_i \left(\sum_{j \neq i} \lambda_{ij}(\boldsymbol{\pi}) + \sigma_i \right), \ \ \forall i \in [n]_+ \,, \tag{70}$$

where $\lambda_{ij}(\boldsymbol{\pi}) + \sigma_i \geq 0$, $i \in [n]_+$ and $j \in [n]$. Eq. (70) shows that from each state $i \in [n]_+$ into the states in $[n]_-$, there exists a total of σ_i "additional outgoing rate", compared to Eq. (4). At the same time, for $i \in [n]_-$ in Eq. (26), we have:

$$\sum_{j \in [n]_+} \pi_j \lambda_{ji}(\boldsymbol{\pi}) + \sum_{j \in [n]_- | j \neq i} \pi_j \lambda_{ji}(\boldsymbol{\pi}) = \pi_i \sum_{j \neq i} \lambda_{ij}(\boldsymbol{\pi}) + \pi_i \sigma_i, \quad \forall i \in [n]_- \,. \tag{71}$$

Since $\pi_i \sigma_i < 0$, for $i \in [n]_-$, we distribute these terms into the first sum on the left hand side. Then, Eq. (71) is equivalent to:

$$\sum_{j \in [n]_+} \pi_j \left(\lambda_{ji}(\boldsymbol{\pi}) - \frac{\pi_i \sigma_i c_j}{\pi_j} \right) + \sum_{j \in [n]_- | j \neq i} \pi_j \lambda_{ji}(\boldsymbol{\pi}) = \pi_i \sum_{j \neq i} \lambda_{ij}(\boldsymbol{\pi}), \quad \forall i \in [n]_- \,, \tag{72}$$

where $\sum_{j \in [n]_+} c_j = 1$.

To determine the $\{c_j\}_{j \in [n]_+}$, recall from Eq. (70) that from each state $j \in [n]_+$ into the states $i \in [n]_-$, there exists a total of σ_j additional outgoing rate. Then, Eq. (72) implies that $\sum_{i \in [n]_-} -\frac{\pi_i \sigma_i c_j}{\pi_j} = \sigma_j$, i.e., $c_j = \frac{-\pi_j \sigma_j}{\sum_{i \in [n]_-} \pi_i \sigma_i}$, $j \in [n]_+$. Using $-\sum_{i \in [n]_-} \pi_i \sigma_i = \sum_{i \in [n]_+} \pi_i \sigma_i$ from Eq. (69), we confirm that $\sum_{j \in [n]_+} c_j = \frac{-\sum_{j \in [n]_+} \pi_j \sigma_j}{\sum_{i \in [n]_-} \pi_i \sigma_i} = 1$, and rewrite $\{c_j\}_{j \in [n]_+}$ as:

$$c_j = \frac{-2\pi_j \sigma_j}{\sum_{i \in [n]_-} \pi_i \sigma_i - \sum_{i \in [n]_+} \pi_i \sigma_i}, \quad \forall j \in [n]_+ \,. \tag{73}$$

Finally, setting $\{c_j\}_{j \in [n]_+}$ into Eq. (72), we have:

$$\sum_{j \in [n]_+} \pi_j \left(\lambda_{ji}(\boldsymbol{\pi}) + \frac{2\pi_i \sigma_i \sigma_j}{\sum_{t \in [n]_-} \pi_t \sigma_t - \sum_{t \in [n]_+} \pi_t \sigma_t} \right) + \sum_{j \in [n]_- | j \neq i} \pi_j \lambda_{ji}(\boldsymbol{\pi}) = \pi_i \sum_{j \neq i} \lambda_{ij}(\boldsymbol{\pi}), \tag{74}$$

where $\lambda_{ji}(\boldsymbol{\pi}) + \frac{2\pi_i \sigma_i \sigma_j}{\sum_{t \in [n]_-} \pi_t \sigma_t - \sum_{t \in [n]_+} \pi_t \sigma_t} \geq 0$, $j \in [n]_+$ and $i \in [n]_-$.

Eq. (26), partitioned as Eq. (70) and Eq. (74), is the balance equation of a continuous-time MC with transition rates given by:

$$\mu_{ji}(\boldsymbol{\pi}) = \begin{cases} \lambda_{ji}(\boldsymbol{\pi}) + \frac{2\pi_i\sigma_i\sigma_j}{\sum_{t\in[n]_-}\pi_t\sigma_t - \sum_{t\in[n]_+}\pi_t\sigma_t} \\ \qquad\qquad \text{if } j \in [n]_+ \text{ and } i \in [n]_- \\ \lambda_{ji}(\boldsymbol{\pi}) \quad \text{otherwise.} \end{cases} \tag{75}$$

Hence, $\boldsymbol{\pi}$ is the stationary distribution of this MC (c.f. Section 2).

E PROOF OF THEOREM 3.3

We use the following definition.

Definition E.1 (Diagonal Dominance). *A matrix $\boldsymbol{H}$ is diagonally dominant if $|\boldsymbol{H}_{ii}| \geq \sum_{j\neq i}|\boldsymbol{H}_{ij}|$, $i \in [n]$, i.e., for every row, magnitude of the diagonal element is larger than the sum of magnitudes of the corresponding off-diagonal elements [Horn and Johnson 2012].*

For $\boldsymbol{\sigma} = [\sigma_i]_{i\in[n]}$ given by (46), Eq. (65) is equivalent to:

$$\frac{\partial L_\rho(\boldsymbol{\pi}, \boldsymbol{w}^k, \boldsymbol{y}^k)}{\partial \pi_i} = \sum_{\ell\in W_i} -\frac{1}{\pi_i} + \sum_{\ell | i\in A_\ell} \frac{1}{\sum_{t\in A_\ell}\pi_t} + \rho(\boldsymbol{\pi} - \tilde{\boldsymbol{\pi}}(X;\boldsymbol{w}^k))_i + y_i^k \,, \tag{76}$$

for all $i \in [n]$. At the k-th iteration of (21), let $\nabla^2 L_\rho(\boldsymbol{\pi}, \boldsymbol{w}^k, \boldsymbol{y}^k)$ be the Hessian of the augmented Lagrangian w.r.t. $\boldsymbol{\pi}$. Differentiating Eq. (76) w.r.t. π_j, $\nabla^2 L_\rho(\boldsymbol{\pi}, \boldsymbol{w}^k, \boldsymbol{y}^k)$ has the following form:

$$\nabla^2_{ij} L_\rho(\boldsymbol{\pi}, \boldsymbol{w}^k, \boldsymbol{y}^k) = \begin{cases} \sum_{\ell\in W_i}\frac{1}{\pi_i^2} - \sum_{\ell | i\in A_\ell}\frac{1}{(\sum_{t\in A_\ell}\pi_t)^2} + \rho, & i = j \\ -\sum_{\ell | i,j\in A_\ell}\frac{1}{(\sum_{t\in A_\ell}\pi_t)^2}, & i \neq j. \end{cases} \tag{77}$$

Consider $\rho \geq \frac{2}{\epsilon^2}\max_i \sum_{\ell|i\in A_\ell}\frac{1}{|A_\ell|^2}$. By Assumption 3.1, we have:

$$\begin{aligned} &\rho \geq \frac{2}{\epsilon^2}\sum_{\ell|i\in A_\ell}\frac{1}{|A_\ell|^2} \;\forall i \in [n]\,, \\ \Leftrightarrow\;& \rho \geq \sum_{\ell|i\in A_\ell}\frac{2}{(\sum_{t\in A_\ell}\pi_t)^2} \;\forall i \in [n]\,, \\ \Leftrightarrow\;& \rho + \sum_{\ell\in W_i}\frac{1}{\pi_i^2} > \sum_{\ell|i\in A_\ell}\frac{2}{(\sum_{t\in A_\ell}\pi_t)^2} \;\forall i \in [n]\,, \qquad (78a) \\ \Leftrightarrow\;& \rho + \sum_{\ell\in W_i}\frac{1}{\pi_i^2} > \sum_{\ell|i\in A_\ell}\frac{1}{(\sum_{t\in A_\ell}\pi_t)^2} + \sum_{j\neq i}\sum_{\ell|i,j\in A_\ell}\frac{1}{(\sum_{t\in A_\ell}\pi_t)^2} \;\forall i \in [n]\,. \qquad (78b) \end{aligned}$$

Eq. (78a) implies that all diagonal elements of $\nabla^2 L_\rho(\boldsymbol{\pi}, \boldsymbol{w}^k, \boldsymbol{y}^k)$ are positive. Moreover, by Eq. (78b), $\nabla^2 L_\rho(\boldsymbol{\pi}, \boldsymbol{w}^k, \boldsymbol{y}^k)$ is diagonally dominant (c.f. Definition E.1). Thus, $\nabla^2 L_\rho(\boldsymbol{\pi}, \boldsymbol{w}^k, \boldsymbol{y}^k)$ is positive definite [Horn and Johnson 2012], i.e., $L_\rho(\boldsymbol{\pi}, \boldsymbol{w}^k, \boldsymbol{y}^k)$ is convex w.r.t. $\boldsymbol{\pi}$. As a result, under Assumption 3.1, for $\rho \geq \frac{2}{\epsilon^2}\max_i \sum_{\ell|i\in A_\ell}\frac{1}{|A_\ell|^2}$ and $D_p(\boldsymbol{\pi}||\tilde{\boldsymbol{\pi}}) = \frac{1}{2}\,\|\,\boldsymbol{\pi} - \tilde{\boldsymbol{\pi}}\,\|_2^2$, a stationary $\boldsymbol{\pi} > \mathbf{0}$ satisfying condition (23) is also a minimizer of step (21a).

F PROOF OF THEOREM 3.4

We make use of the following lemmas.

Lemma F.1 (Zeng et al. [2018]). *Logarithm and polynomials are Kurdyka–Łojasiewicz functions. Moreover, sums, products, compositions, and quotients (with denominator bounded away from 0) of Kurdyka–Łojasiewicz functions are also Kurdyka–Łojasiewicz.*

Lemma F.2 (Guo et al. [2017]). *Consider the optimization problem:*

$$\begin{aligned} \underset{\boldsymbol{w},\,\boldsymbol{\pi}}{\text{minimize}} \quad & g(\boldsymbol{\pi}) \\ \text{subject to} \quad & \tilde{\boldsymbol{X}}\boldsymbol{w} = \boldsymbol{\pi}, \end{aligned} \tag{79}$$

and solve Eq. (79) *via Alternating Direction Method of Multipliers (ADMM) [Boyd et al. 2011]. Let* $\{(\boldsymbol{\pi}^k, \boldsymbol{y}^k, \boldsymbol{w}^k)\}_{k\in\mathbb{N}}$ *be the sequence generated by the ADMM algorithm, and* ρ *be the penalty parameter of ADMM. Suppose that there exists* $\kappa > 0$ *such that* $\tilde{\boldsymbol{X}}^T\tilde{\boldsymbol{X}} \succeq \kappa\,\boldsymbol{I}$*, and the sequence* $\{(\boldsymbol{\pi}^k, \boldsymbol{y}^k, \boldsymbol{w}^k)\}_{k\in\mathbb{N}}$ *is bounded.*

If there exist solutions for the minimization steps of ADMM w.r.t. both $\boldsymbol{\pi}$ *and* $\boldsymbol{w}$*,* $g(\boldsymbol{\pi})$ *is a continuous differentiable function with an L-Lipschitz continuous gradient at* $\boldsymbol{\pi}^k$*,* $k \in \mathbb{N}$ *where* $L > 0$*, and the augmented Lagrangian of Eq.* (79) *is a Kurdyka–Łojasiewicz function, then, for* $\rho > 2L$*,* $\{(\boldsymbol{\pi}^k, \boldsymbol{y}^k, \boldsymbol{w}^k)\}_{k\in\mathbb{N}}$ *converges to a point that satisfies the Karush-Kuhn-Tucker (KKT) conditions of Eq.* (79)*.*

For affine regression described in Section 3.3, there exist solutions for the minimization steps in (21): $\boldsymbol{w}$ update has the closed form solution given by Eq. (33) and $\boldsymbol{\pi}$ update admits a minimizer for large enough ρ by Lemma 3.3.

By Assumption 3.1, $\nabla_{\boldsymbol{\pi}}\mathcal{L}$ given by Eq. (58) exists, i.e. $\mathcal{L}$ is continuous differentiable at $\boldsymbol{\pi}^k$, $k \in \mathbb{N}$ generated by (21a). Let $\nabla^2(\mathcal{L})$ be the Hessian of $\mathcal{L}$. Differentiating Eq. (58) w.r.t. π_j, $\nabla^2(\mathcal{L})$ has the following form:

$$\nabla^2_{ij}(\mathcal{L}) = \begin{cases} \sum_{\ell\in W_i} \frac{1}{\pi_i^2} - \sum_{\ell|i\in A_\ell} \frac{1}{(\sum_{t\in A_\ell}\pi_t)^2}, & i = j \\ -\sum_{\ell|i,j\in A_\ell} \frac{1}{(\sum_{t\in A_\ell}\pi_t)^2}, & i \neq j. \end{cases} \tag{80}$$

Consider $L = \frac{\max_i |W_i|}{\epsilon^2}$, where W_i is the set of observations where sample $i \in [n]$ is chosen. By Assumption 3.1, we have:

$$\begin{aligned} & L = \frac{\max_i |W_i|}{\epsilon^2} \geq \sum_{\ell\in W_i} \frac{1}{\pi_i^2} \;\forall i \in [n]\,, \\ \Leftrightarrow\;\; & L - \sum_{\ell\in W_i} \frac{1}{\pi_i^2} + \sum_{\ell|i\in A_\ell} \frac{1}{\left(\sum_{t\in A_\ell}\pi_t\right)^2} \geq \sum_{\ell|i\in A_\ell} \frac{1}{\left(\sum_{t\in A_\ell}\pi_t\right)^2} \;\forall i \in [n]\,, \\ \Leftrightarrow\;\; & L - \sum_{\ell\in W_i} \frac{1}{\pi_i^2} + \sum_{\ell|i\in A_\ell} \frac{1}{\left(\sum_{t\in A_\ell}\pi_t\right)^2} \geq \sum_{j\neq i} \sum_{\ell|i,j\in A_\ell} \frac{1}{\left(\sum_{t\in A_\ell}\pi_t\right)^2} \;\forall i \in [n]\,. \end{aligned} \tag{81}$$

Now, consider the matrix $L\boldsymbol{I}_{n\times n} - \nabla^2(\mathcal{L})$. By Eq. (81), $L\boldsymbol{I}_{n\times n} - \nabla^2(\mathcal{L})$ is diagonally dominant (c.f. Definition E.1) and all of its diagonal elements are positive, i.e., $\nabla^2(\mathcal{L})$ is upper bounded by $L\boldsymbol{I}_{n\times n}$. Thus, the objective function of Eq. (16), i.e. $\mathcal{L}$, has an L-Lipschitz continuous gradient at $\boldsymbol{\pi}^k$, $k \in \mathbb{N}$, where $L = \frac{\max_i |W_i|}{\epsilon^2} > 0$.

Moreover, the augmented Lagrangian given by Eq. (20) is a sum of three functions: logarithm of the ratio of two polynomials where the denominator is bounded away from 0 for all $\boldsymbol{\pi}^k$, $k \in \mathbb{N}$ by Assumption 3.1, and two other polynomial functions. By Lemma F.1, these three functions and their sum is Kurdyka–Łojasiewicz on the set $\{\boldsymbol{\pi}^k \mid \pi_i^k > \epsilon\,, i \in [n]\,, k \in \mathbb{N}\}$. As a result, the augmented Lagrangian of Eq. (16) is a Kurdyka–Łojasiewicz function.

Putting it all together, by Lemma F.2, for affine regression (c.f. Section 3.3) with $\rho > \frac{2 \max_i |W_i|}{\epsilon^2}$, the sequence $\{(\boldsymbol{\pi}^k, \boldsymbol{y}^k, \boldsymbol{w}^k)\}_{k \in \mathbb{N}}$ generated by (21) converges to a point that satisfies the KKT conditions [Nocedal and Wright 2006] of Problem (16).