Spectral Ranking with Covariates

Siu Lun Chau
Department of Statistics
University of Oxford
United Kingdom, OX1 3LB

Mihai Cucuringu
Department of Statistics
University of Oxford
United Kingdom, OX1 3LB

Dino Sejdinovic
Department of Statistics
University of Oxford
United Kingdom, OX1 3LB

Abstract

We consider approaches to the classical problem of establishing a statistical ranking on a given set of items from incomplete and noisy pairwise comparisons, and propose spectral algorithms able to leverage available covariate information about the items. We give a comprehensive study of several ways such side information can be useful in spectral ranking. We establish connections of the resulting algorithms to reproducing kernel Hilbert spaces and associated dependence measures, along with an extension to fair ranking using statistical parity. We present an extensive set of numerical experiments showcasing the competitiveness of the proposed algorithms with state-of-the-art methods.

1 Introduction

Data concerning user preferences for items or services is ubiquitous and is often used to detect patterns in user behaviour for the purpose of making recommendations. However, in many cases, an explicit direct feedback from the user-specific objective function of interest (e.g. a product rating), which is needed to make recommendations, is scarce and the quantity of implicit feedback data (in the form of clicks, views or purchases when faced with multiple choices) far outnumbers the explicit data. Moreover, when the feedback comes from human users, they are known to be better in evaluating *relative* differences than *absolute* quantities [Kahneman et al., 1979] and, in the absence of a reference point, explicit feedback may be unreliable.

Motivated by this, we consider the setting of establishing a ranking on n items, given a set of incomplete and noisy relative preferences, but we revisit this classical problem in light of item covariate information. Ranking is a well-studied discipline with an active and rich literature that dates back to the 1950s [Bradley et al., 1952, Luce, 1959]. There exists a wide range of settings under which the problem of ranking arises and can be studied, mainly motivated by information retrieval tasks. In our work, we consider the setting where a single observation between pairs of items is observed, and aim to recover a total ordering that is as consistent as possible with the given data. To study this problem, spectral ranking methods such as SERIAL-RANK [Fogel et al., 2016], SYNC-RANK [Cucuringu, 2016] and SVD-RANK [Alexandre et al., 2019] have been proposed. However, the aforementioned algorithms do not consider cases where side information or relationships between items are available, which is often the case. In fact, there are other ranking methods that do use covariates as well, but they are not designed for our setting.

It is natural to combine ranking with item covariates. For example, when ranking a set of clothing articles, it is no surprise that their attributes such as size, style and colour will have an impact on the ranking. Moreover, covariate-free algorithms cannot take into account instances where new items are added to the set, without generating new matches with existing items. Finally, these methods often break down if the sample complexity of the comparison graph does not scale as $O(n\log(n))$, meaning the graph being disconnected with high probability [Alexandre et al., 2019]. With the presence of features, even if the comparison graph is very sparse or even disconnected, we can still

Preprint. Under review.

produce a meaningful ranking since our proposed algorithm effectively ranks in both match outcomes and feature representations, as opposed to just based on the match outcomes.

Contributions Besides proposing spectral methods that connect item covariates to their ranking, we also consider the following questions:

1. Given a list of item attributes, how does one quantify their relevance to the ranking problem? To build on top of that, can one identify the subset of attributes (if any) that are most relevant for the ranking task?
2. Can we rank a previously unseen item solely based on its attributes, without having access to any pairwise comparisons?
3. Since ranking is a sensitive task arising in a myriad of information retrieval problems handling sensitive personal user data, is there a way to make our algorithm fair with respect to certain sensitive features such as gender and ethnicity?

In this work, we introduce a suite of algorithms, C-SERIAL-RANK, SVDCOV-RANK and KCCA-RANK. Both C-SERIAL-RANK and SVDCOV-RANK are built on existing spectral ranking techniques, while KCCA-RANK directly tackles the question of attribute relevance outlined above. All the proposed methods appeal to a class of expressive non-parametric models based on reproducing kernel Hilbert spaces (RKHS).

The core idea behind our proposed methodologies is to treat the pairwise comparisons and the item covariates as two sources of information describing the same set of items. We consider ranking as a skill function such that similar items will give similar evaluations, where the notion of similarity is computed using the comparisons (classical setting) or a combination of comparisons and covariate information.

Outline The paper is organised as follows. Section 2 covers related work in the spectral ranking and kernel methods literature. Our proposed methodologies will be presented in Section 3. Sections 4 and 5 cover experiments and concluding remarks.

Notations Vectors and matrices are denoted by lower case and upper case letters respectively. In general, there will be n items to compare, and we denote the pairwise comparison matrix by $C \in \mathbb{R}^{n \times n}$, where $C_{i,j} = 1$ if i beats j: -1 if j beats i: or 0 if draw or no match for *ordinal* comparisons. In the case for *cardinal* comparisons, $C_{i,j} = r_i - r_j$ for a given ranking/strength vector r. We denote the covariate matrix by $\Phi \in \mathbb{R}^{n \times p}$. Furthermore, given r, we can infer the ranking by sorting the entries of r, choosing either increasing or decreasing order to *minimise* the number of upsets. We formalise the notion of upsets below:

Definition 1 (Upsets). *Given a ranking vector r, an upset is a pair of items for which the higher ranked item is preferred over the lower ranked item. For example, there is an upset if $r_i > r_j$ but $C_{i,j} < 0$.*

2 Background

We briefly survey related work in the ranking and kernel methods literature, and lay out the foundation for Section 3.

2.1 Ranking Methods

We will briefly review relevant ranking methods that also rely on features, and a number of existing spectral ranking algorithms on which our proposed methods build on.

Ranking with Covariates

The two most relevant lines of work in ranking with covariates are from [Yi et al., 2016] and [Niranjan et al., 2017]. However, it is worth noting that the two methods are designed for rank aggregation, where multiple observations between selected pairs are recorded from multiple sources in order to generate a better single ranking list. This is different from our setting where only a single observation is recorded for each of the selected pairs.

Bayesian Aggregation of Rank-data with Covariates [Yi et al., 2016] incorporated not only side information on the ranked items, but also features about the sources. They set up a Bayesian model and connect the items and sources' features to a latent underlying score vector, and treat observations as a deviation from this truth. A parameter-expanded Gibbs sampler is then used for the inference task.

Inductive Pairwise Ranking (IPR) On the other hand, [Niranjan et al., 2017] looked into settings where each selected pair is compared multiple times (at least twice). They assume features lie in a low-dimensional space and use matrix completion in the presence of side information to obtain a ranking. They computed an empirical probability score using those multiple observations for each pair of matches. In our setting, as we only have a single observation, the empirical probability measure will not be reliable.

To apply the algorithm to our problem, we will appeal to the adjustment similar to how [Cucuringu, 2016] adjusted Rank centrality [Negahban et al., 2016] using similarity measures between matches to construct a proxy for probabilities. For example, given a symmetric similarity matrix S with $S_{ij} \leq n$ (we will demonstrate how one could compute it from C), one considers the following probability matrix

$$\hat{P}_{i,j} = \left\{ \begin{array}{ll} 1 - \frac{S_{i,j}}{2n}, & \text{if } i \text{ won against } j, \\ \frac{S_{i,j}}{2n}, & \text{if } j \text{ won against } i, \\ \frac{1}{2}, & \text{otherwise.} \end{array} \right\} \tag{1}$$

Note that, whenever $S_{i,j}$ is large, meaning the two players are very similar, the quantity $1 - \frac{S_{i,j}}{2n}$ will be small, thus it is a good proxy for the difference in the winning probabilities.

Spectral Ranking

Next, we move on to summarize existing spectral algorithms, on which our proposed methods are based.

Serial-Rank [Fogel et al., 2014, 2016] proposed a seriation approach to ranking building on [Jonathan et al., 1998], by making use of the similarity between players. The approach is based solely on the available pairwise comparisons, in the spirit of standard spectral clustering techniques that consider the graph Laplacian. Given an ordinal pairwise comparison matrix C, where $C_{i,j} \in \{-1, 0, 1\}$, we construct a similarity matrix $S = \frac{1}{2}(n\mathbf{1}\mathbf{1}^T + CC^T)$ such that $S_{i,j}$ counts the number of agreeing comparisons between i and j with other items. To recover the true ordering in this seriation process, we proceed by setting up the graph Laplacian $\mathcal{L}(S) = \textbf{diag}(S\mathbf{1}) - S$ and obtain the eigenvector of $\mathcal{L}(S)$ corresponding to the second smallest eigenvalue,

$$r_* = \arg\min_r r^\top \mathcal{L}(S) r \quad \text{s.t} \quad r^\top r = 1, \quad r^\top \mathbf{1} = 0. \tag{2}$$

The resulting r_* will be a smooth vector with respect to the similarity graph S since $r^\top \mathcal{L}(S) r = \frac{1}{2}\sum_{i>j} S_{i,j}(r_i - r_j)^2$ measures the smoothness of a vector with respect to graph S [Shuman et al., 2013]. To recover the final ranking, one simply sorts the ranking vector r_* to minimise the upsets.

SVD-Rank [Alexandre et al., 2019, Cucuringu, 2016] considered a simple spectral method for cardinal measurements, arising from the model wherein the pairwise comparisons are modelled as noisy entries of a rank-2 matrix $C = r\mathbf{1}^T - \mathbf{1}r^T$, or a proxy of it. By a simple spectral decomposition

$$r^* = \arg\max_r \left(r^\top CC^\top r\right), \quad \text{s.t.} \quad r^\top r = 1, \quad r^\top \mathbf{1} = 0. \tag{3}$$

one computes the top two singular vectors of the pairwise comparisons matrix C, and extracts the rankings induced by their entries.

The extension SVD-NORM RANK considers the normalisation akin to the connection-Laplacian [Singer et al., 2012], and relies on $\hat{C} = D^{-1}C$, where D is the diagonal matrix with entries equal to the total degree (both out and in degree) of the corresponding node in the pairwise comparison network [Alexandre et al., 2019]. As illustrated by the numerical experiments, SVD-NORM-RANK performs better than SVD-RANK. Furthermore, [Alexandre et al., 2019] provide a detailed theoretical consistency analysis of both algorithms, under a random measurement model in terms of robustness against sampling sparsity of the measurement graph and noise level.

2.2 Kernel Methods, Hilbert Schmidt Independence Criterion and Kernel Canonical Correlation

For any positive definite function $k : \mathcal{X} \times \mathcal{X} \longmapsto \mathbb{R}$, there exits a unique reproducing kernel Hilbert space (RKHS) $\mathcal{H}_k$ of real-valued functions on $\mathcal{X}$. Function $k(\cdot, x)$ is an element of $\mathcal{H}_k$ and represents evaluation at x, i.e. $\langle f, k(\cdot, x)\rangle = f(x), \forall f \in \mathcal{H}_k, \forall x \in \mathcal{X}$.

Hilbert-Schmidt Independence Criterion Now consider random variables x and z taking values in general domains $\mathcal{X}$ and $\mathcal{Z}$. Given kernel functions k and g on $\mathcal{X}$ and $\mathcal{Z}$ respectively, with RKHS $\mathcal{H}_k$ and $\mathcal{H}_g$, the cross-covariance operator is defined as a linear operator $\Sigma_{xz} : \mathcal{H}_g \to \mathcal{H}_k$ such that $\langle f, \Sigma_{xz} h\rangle = Cov[f(x), h(z)]$ for all $h \in \mathcal{H}_g, f \in \mathcal{H}_k$. Hilbert-Schmidt Independence Criterion (HSIC) measuring dependence between x and z is then given by the Hilbert-Schmidt norm of Σ_{xz}. HSIC can also be seen as the maximum mean discrepancy (MMD) [Gretton et al., 2012] between the joint probability measure P_{xz} and the product of their marginals. The empirical estimator of HSIC [Gretton et al., 2005] is given by

$$\widehat{HSIC}_{k,g}(x, z) = \frac{1}{n^2}\text{Tr}(KHGH), \tag{4}$$

where K, G are the kernel matrices computed on paired observations $\{x_i\}_{i=1}^n$ and $\{z_i\}_{i=1}^n$ using kernels k and g respectively, and $H = I - \frac{1}{n}\mathbf{1}\mathbf{1}^\top$ is the centering matrix for data in the feature space.

For a broad family of kernels such as the Gaussian and Matérn family, the population HSIC is 0 if and only if x and z are statistically independent. Based on this fact, nonparametric independence testing [Gretton et al., 2008] can be conducted with null hypothesis $H_0 : P_{xz} = P_x P_z$ vs. the general alternative $H_1 : P_{xz} \neq P_x P_z$ to test for independence. Feature selection methods based on this dependence measure have been proposed in [Song et al., 2012].

Kernel Canonical Correlation Analysis Besides the independence test, the cross covariance operator Σ_{xz} can also be used in the Kernel Canonical Correlation Analysis (KCCA) [Fukumizu et al., 2007], which lies at the core of one of our proposed ranking methods.

Assume we are given two sets of data $\{x_i\}_{i=1}^n$ and $\{z_i\}_{i=1}^n$, which describe the same set of items through possibly two very different ways. We are interested to learn the optimal transformation in the feature spaces of these two views of the items, such that the two resulting embeddings have the maximal correlation under some restrictions in the RKHSs. Mathematically, this corresponds to the following optimisation

$$f_*, g_* = \underset{f\in\mathcal{H}_k, g\in\mathcal{H}_h}{\text{argmax}} \langle f, \hat{\Sigma}_{xz} g\rangle_{\mathcal{H}_h} \quad \text{s.t} \begin{cases} \langle f, (\hat{\Sigma}_{xx} + \epsilon I_n) f\rangle = 1 \\ \langle g, (\hat{\Sigma}_{zz} + \epsilon I_n) g\rangle = 1, \end{cases} \tag{5}$$

where $\hat{\Sigma}_{xz}$ is the empirical estimate of Σ_{xz} and ϵ is small perturbation for numerical stability. The KCCA solution can be obtained via computing the eigenfunctions of the normalised cross-covariance operator Σ_{xz} or via a generalised eigendecomposition route. For further details, we refer the reader to [Fukumizu et al., 2007].

Algorithmic Fairness using HSIC *Algorithmic fairness* is an emerging field motivated by increasing concerns about the lack of fairness, equity and ethics in machine learning practice, and the way it impacts society. There are many definitions of algorithmic fairness, and we will focus on *fairness by statistical parity* as discussed in [Perez et al., 2017, Li et al., 2019], and in particular, parity-fairness in expectation.

Consider a general empirical risk minimisation (ERM) setting where we use x to predict y by constructing a function $f : \mathcal{X} \mapsto \mathcal{Y}$ to minimise empirical risk for some loss function L. We also have in hand some sensitive features z, e.g. gender or ethnicity, and we aim to minimise their influence on the learned function. To this end, we enforce parity-fairness in expectation, requiring that $\mathbb{E}_{x|z}[f(x)|z]$ does not depend on z. Following [Li et al., 2019], this corresponds to including the empirical HSIC as a regularisation term in the objective function

$$f_* = \arg\min_{f\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} L(f(x_i), y_i) + \lambda\frac{1}{n^2} f^\top HGHf, \tag{6}$$

using a linear kernel k on function values $f(x)$, with $\mathcal{H}$ a hypothesis class, and a regularisation parameter λ.

3 Proposed Methodology

In this section, we introduce C-SERIAL-RANK, SVDCOV-RANK and KCCA-RANK. All three lines of algorithms have a different flavour in mixing the information between matches and covariates. C-SERIAL-RANK can control via its hyperparameter, the information contributed from features, thus even if they are noisy, it can still give a robust ranking. On the other hand, SVDCOV-RANK connects r with covariates via a regression fashion, allowing us to predict the rank on completely unseen items while no comparison with existing items are needed. Finally, KCCA-RANK combines the merit of the two methods above by learning two functions f and g separately on the matches and covariates to rank.

3.1 C-Serial-Rank

Recall in Serial-Rank, we first compute a similarity graph based on the matches and compute a non-constant smooth vector r with respect to the graph Laplacian, which itself can be seen as an operator that measures smoothness [Shuman et al., 2013].

With the available covariate information about the items, we construct another similarity matrix using the Gram matrix K, with $K_{i,j} = k(\phi_i, \phi_j)$. The gist of the idea is to merge this extra source of similarity with S, and consider the eigen-decomposition on the resulting Laplacian

$$r_* = \arg\min_r r^T \big(\mathcal{L}(S + \lambda K)\big) r \quad \text{s.t} \quad r^T r = 1 \quad r^T \mathbf{1} = 0, \tag{7}$$

where the trade-off parameter λ controls the information contributed from the features. λ and the kernel hyperparameters can be tuned using cross-validation on held out matches.

Algorithm 1: C-SERIAL-RANK

```
  Input : A set of pairwise comparisons $C_{i,j} \in \{-1, 0, 1\}$, a feature matrix $\Phi$, a kernel function
          $k$, hyperparameter $\lambda$
1 Compute a similarity matrix $S = \frac{1}{2}(n\mathbf{1}\mathbf{1}^T + CC^T)$ and kernel matrix $K$ with $K_{i,j} = k(\phi_i, \phi_j)$
2 Set up the graph Laplacian matrix $\mathcal{L}(\hat{S}) = \mathbf{diag}(\hat{S}\mathbf{1}) - \hat{S}$ where $\hat{S} = S + \lambda K$. Compute the
    eigenvector $r$ of the smallest non-zero eigenvalue of $\mathcal{L}(\hat{S})$.
  Output : A ranking induced by sorting the eigenvector $r$ to minimise the number of upsets.
```

3.2 SVD-Rank with covariates

The previous section provided a simple method that incorporate features into the similarity graph. However, a downside of this extension is that one cannot perform predictions on unseen items using available covariate information. To this end, we introduce SVD-Rank with covariates, for connecting features to the ranking vector r in a regression fashion.

In order to solve SVD-RANK, we convert Eq (3) into a standard eigenvector problem. We project r onto $\{\mathbf{1}\}^\perp$ by setting $r = Zq$, where $Z \in \mathbb{R}^{n \times (n-1)}$ is a matrix whose columns form an orthonormal basis of $\{\mathbf{1}\}^\perp$. Thus Eq (3) becomes

$$q_* = \arg\max_q \left(q^\top Z^\top CC^\top Zq\right), \quad \text{s.t. } q^\top q = 1. \tag{8}$$

By construction, $r_* = Zq_*$ is orthogonal to $\mathbf{1}$, where q_* is the top eigenvector of $Z^\top CC^\top Z$.

Now consider the scenario where we also have access to covariates $\Phi \in \mathbb{R}^{n \times p}$, and we assume the final ranking implied by r varies smoothly as a linear function of these covariates, e.g $r = \Phi\beta$. By considering the same previous line of thought, we set $\Phi\beta = Zq$, and substitute $q = Z^\top \Phi\beta$ into (8) to arrive at

$$\beta_* = \arg\max_\beta \left(\beta^\top \Phi^\top HCC^\top H\Phi\beta\right), \quad \text{s.t. } \beta^\top \Phi^\top H\Phi\beta = 1. \tag{9}$$

It is interesting to note that (9) corresponds to finding the optimal embedding for the cross-covariance matrix $\Phi^\top HC$, in a similar fashion as Principal Component Analysis.

To solve for β, we decompose our feature covariance matrix $\Phi^\top H\Phi \in \mathbb{R}^{p\times p}$ into $LL^\top$, which can be done using Cholesky decomposition or Singular Value decomposition. We obtain again the standard eigenvector problem by setting $\beta = L^{-\top}\gamma$, which leads to

$$\gamma_* = \arg\max_\gamma \left(\gamma^\top \Psi\gamma\right), \quad \text{s.t. } \gamma^\top\gamma = 1, \tag{10}$$

with $\Psi = L^{-1}\Phi^\top HCC^\top H\Phi L^{-\top}$. Finally, we recover the ranking from $r = H\Phi L^{-\top}\gamma_\star$, where $\gamma_\star$ is the top eigenvector of Ψ.

Furthermore, if there is reason to believe that the ranking vector r relates to the features non-linearly, it is straightforward to "kernelise" the above by setting $r = K\alpha$, where K is the kernel matrix of covariates, and obtain a non-parameteric version of SVDCov-Rank, which we denote as SVDKCov-Rank. Substituting $q = Z^\top K\alpha$ into (8), leads to

$$\alpha_* = \arg\max_\alpha \left(\alpha^\top KHCC^\top HK\alpha\right), \quad \text{s.t. } \alpha^\top KHK\alpha = 1. \tag{11}$$

The rest follows closely to the derivation of SVDCov-Rank with details in Algorithm 3.

Note that for large scale ranking problems where the comparison matrix is usually sparse in nature (and so is the corresponding Graph Laplacian), one can compute (via the power iteration method) the dominant eigenvectors in almost linear time in the number of edges of the graph. For sparse matrices, each iteration of the power method is of linear time and the number of iterations depends on the spectral gap. As a consequence of this, spectral ranking methods such as Serial-rank and C-Serial-rank are able to scale to very large problems in practice.

```
Algorithm 2: SVDCov-Rank
   Input  : A set of pairwise comparisons $C_{i,j} \in \{-1,0,1\}$ or $C_{i,j} \in \mathbb{R}$, covariate matrix $\Phi$
 1 Compute $\Psi = L^{-1}\Phi^\top HCC^\top H\Phi L^{-\top}$, where $L$ is the Cholesky decomposition of $\Phi^\top H\Phi$.
 2 Compute the top eigenvector $\gamma$ of $\Psi$ and set $r = H\Phi L^{-\top}\gamma$
   Output : A ranking induced by sorting the $r$ to minimise the number of upsets.
```

```
Algorithm 3: SVDKCov-Rank
   Input  : A set of pairwise comparisons $C_{i,j} \in \{-1,0,1\}$ or $C_{i,j} \in \mathbb{R}$, kernel matrix $K$
 1 Compute $\Psi = L^{-1}KHCC^\top HKL^{-\top}$ where $L$ is the Cholesky decomposition of $KHK$.
 2 Compute the top eigenvector $\gamma$ of $\Psi$ and set $r = HKL^{-\top}\gamma$
   Output : A ranking induced by sorting the $r$ to minimise the number of upsets.
```

3.3 SVDKFair-Rank

We consider a notion of fairness following [Perez et al., 2017], known as fairness by *statistical parity*, which states that the learning process should be as statistically independent of sensitive variables as possible. As we will see, the proposed method is well suited to this notion of fairness and it leads to simple modifications due to an additional regulariser to the objective.

To achieve this, as described in Section 2.2, we incorporate the empirical HSIC as a regulariser to the SVDKCov-Rank objective function Eq (11). The optimisation objective becomes

$$r_* = \arg\max_r \left(r^\top CC^\top r - \frac{\lambda}{n^2} r^\top HGHr\right). \tag{12}$$

Setting $L = (KHK)^{\frac{1}{2}}$ and $r = KL^\top\gamma$, we arrive at the following optimisation

$$\gamma_* = \arg\max_\gamma \left(\gamma^\top L^{-1}KH\Xi HKL^{-\top}\gamma\right) \quad \text{s.t} \quad \gamma^\top\gamma = 1$$

where $\Xi = CC^\top - \frac{\lambda}{n^2}G$, G is the kernel matrix of the sensitive features and λ being the hyperparameter controlling the degree of fairness imposed from the regularisation. To recover the ranking vector, we simply set $r_* = HKL^{-\top}\gamma_*$.

```
Algorithm 4: SVDKFair-Rank
  Input  : A set of pairwise comparisons $C_{i,j} \in \{-1,0,1\}$ or $C_{i,j} \in \mathbb{R}$, kernel matrix $K$ from
           item covariates, kernel matrix $G$ from sensitive features, $\lambda$.
1 Compute $\Psi = L^{-1}KH\left(CC^\top - \frac{\lambda}{n^2}G\right)HKL^{-\top}$, where $L$ is the Cholesky decomposition of
    $KHK$.
2 Compute the top eigenvector $\gamma$ of $\Psi$ and set $r = HKL^{-\top}\gamma$
  Output : A ranking induced by sorting the eigenvector $r$ to minimise the number of upsets.
```

3.4 KCCA-Rank and testing for feature relevance

To utilise both sources of item information from matches and features, we appeal to the KCCA method described in Section 2, and compute the optimal embeddings that maximise the correlation between them in the embedding space. In other words, given $\{c_i\}_{i=1}^n$, the row vectors of the comparison matrix C, and $\{\phi_i\}_{i=1}^n$, the covariate information, we consider the following optimisation problem

$$f_*, g_* = \underset{f\in\mathcal{H}_k, g\in\mathcal{H}_h}{\operatorname{argmax}} \langle f, \hat{\Sigma}_{\phi c} g\rangle_{\mathcal{H}_h} \quad \text{s.t} \begin{cases} \langle f, (\hat{\Sigma}_{\phi\phi} + \epsilon I_n) f\rangle = 1 \\ \langle g, (\hat{\Sigma}_{cc} + \epsilon I_n) g\rangle = 1 \end{cases} \tag{13}$$

where ϵ are small perturbation to ensure numerical stability. The resulting vectors $f_*(\phi_{1:n})$ and $g_*(c_{1:n})$ can both be used as the ranking vector, and we choose the one that minimises the number of upsets. However, one should note that g_* cannot be used for predicting unseen items, by construction. The rest follows similarly to the derivation of KCCA, and can be solved using a generalised eigenvector decomposition. The details can be seen in Algorithm 5.

```
Algorithm 5: KCCA-Rank
  Input  : Pre-computed kernel matrix $K$ from item covariates, kernel matrix $G$ from matches
           outcome, small perturbation $\epsilon$
1 Compute $\widetilde{K} = HKH$ and $\widetilde{G} = HGH$
2 Compute $K^* = \frac{1}{n}\widetilde{K}\left[\widetilde{K} + \epsilon I_n\right]$ and $G^* = \frac{1}{n}\widetilde{G}\left[\widetilde{G} + \epsilon I_n\right]$
3 Compute the top generalised eigenvector of the problem
      $$\begin{bmatrix} 0 & \frac{1}{n}\widetilde{K}\widetilde{G} \\ \frac{1}{n}\widetilde{G}\widetilde{K} & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \lambda \begin{bmatrix} K^* & 0 \\ 0 & G^* \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \tag{14}$$
4 Compute the ranking vectors $r_1 = KH\alpha_*$ and $r_2 = GH\beta_*$
  Output : A ranking induced by sorting $r_1$ or $r_2$, whichever one minimises the number of upsets.
```

Feature relevance and selection test Following the above setup, by treating the matches and the features as two sets of information about the same items, we can perform the kernel independence test [Gretton et al., 2008] mentioned in Section 2 to test for statistical independence. This is to ensure the features we collected are informative for our ranking problem. Furthermore, using HSIC, one can conduct backward (or forward) feature selection using dependence maximisation to obtain a subset that is most relevant to the ranking problem [Song et al., 2012].

4 Experiments

In this section, we compare the performance of C-Serial-Rank (cs in the figure legends for brevity), SVDCov-Rank (svdc), SVDKCov-Rank (svdk) and KCCA-Rank (kcca) with that of a number of other benchmark algorithms from the literature, including Serial-Rank (ser), SVD-Norm-Rank (svdn). The Adjusted Inductive Pairwise Ranking (ipr) will be compared against when we conduct experiments on real data. Finally, we conclude by presenting numerical results for our SVDKFair-Rank algorithm.

Before performing experiments on real data, we applied the *Backward Elimination Using HSIC* algorithm (BAHSIC) [Song et al., 2012] to select the most informative subset of features to train on.

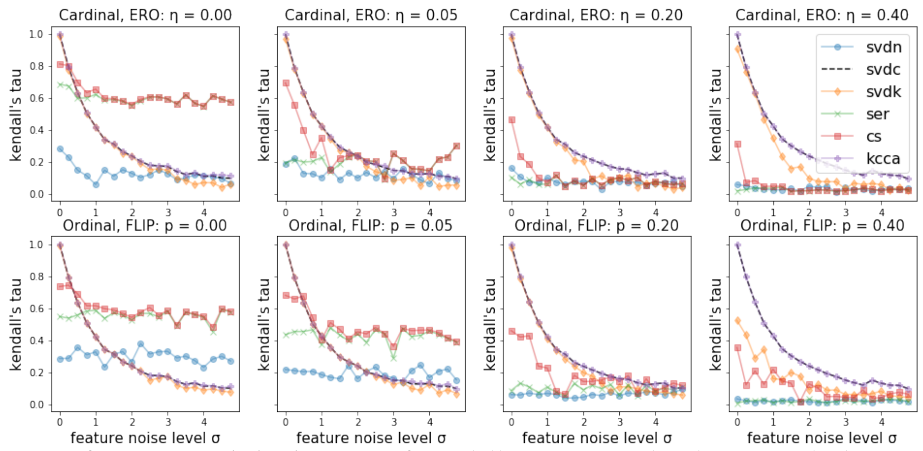


Figure 1: Performance statistics in terms of Kendall's tau score (the closer to 1 the better) for our methods under different synthetic noise models. The network has 1000 nodes and a sparsity of 0.008. Results are averaged over 20 iterations.

For both BAHSIC and our algorithms, we used the Radial basis function kernel (RBF) $K(x, x') = \exp(-\frac{||x-x'||^2}{2d^2})$ and select its length scale d using 10 fold cross-validation. The same procedure is performed for the λ trade-off parameter in CS. In practice, we should pick the kernel that reflects the nature of the nonlinearity in the data.

4.1 Simulation Studies

This section details the outcomes of synthetic numerical experiments, under two noise model for *ordinal* and *cardinal* data respectively. For *ordinal* comparisons, we consider the *FLIP* model where for each observation we flip the outcome with probability p. For *cardinal* data, we use the adjusted Erdös-Rényi model (*ERO*) from [Cucuringu, 2016], where with probability η, we replace a comparison with $\max_{i,j}(r_i - r_j)U$ for $U \sim U[-1, 1]$, given the true underlying ranking vector r.

Besides the noise model on the observed matches, we are interested to investigate the algorithmic performance when noise is added between the features and r. With this objective in mind, we set up our skill function $f(x) = \sin 3\pi x - 1.5x^2$, and the ranking vector as $r_i = f(x_i) + \sigma N(0, 1)$, with σ being the feature noise level. We sample 1000 players $\{x_i\}_{i=1}^{1000}$ from $U[0, 1]$ to compute r. The corresponding comparison matrix will be given by $C_{i,j} = r_i - r_j$ for the cardinal case, and $C_{i,j} = \text{sign}(r_i - r_j)$ for ordinal matches. We measure the *Kendall's Tau* score between the recovered ranking vector r_* and the ground truth r for different noise levels. Our experiments are conducted on a highly sparse yet connected graph (with sparsity = 0.008).

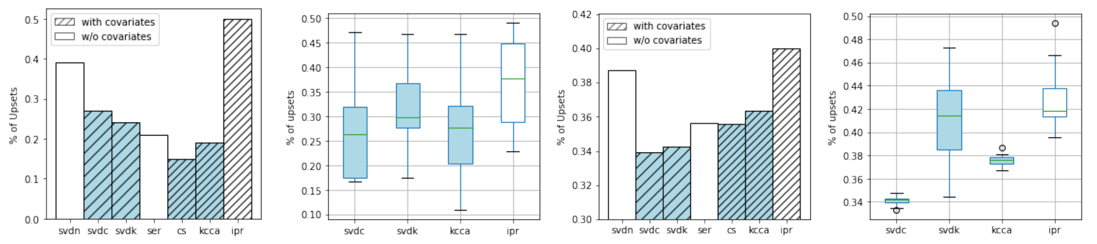
Different noise models correspond to different challenges in ranking with covariates. While σ regulates how informative features are for the true ranking, η/p regulate how noisy comparisons are given the true ranking. In other words, high σ means features are less relevant and we should focus on C, while high η/p means C is less informative and we need side information from the features. Figure 1 demonstrates how the proposed algorithms perform at varying levels of noise. Both CS and SER, on average, outperform the rest in the match noiseless case and when the feature noise level σ is high; this shows how SER can balance the trade-off between matches and covariates. On the other hand, when we start adding noise to the comparison graph (*ERO* or *FLIP*), covariate based methods such as KCCA, SVDC and SVDK were consistently the best performers, demonstrating their merits even with extreme feature noises.

4.2 Experiments on real data

We apply all algorithms to a variety of real data sets corresponding to a range of different comparison graphs, and measure outcome by their proportion of upsets in the final ranking, computed with respect to the given pairwise measurements. We will conduct two main types of experiments, described as follows.

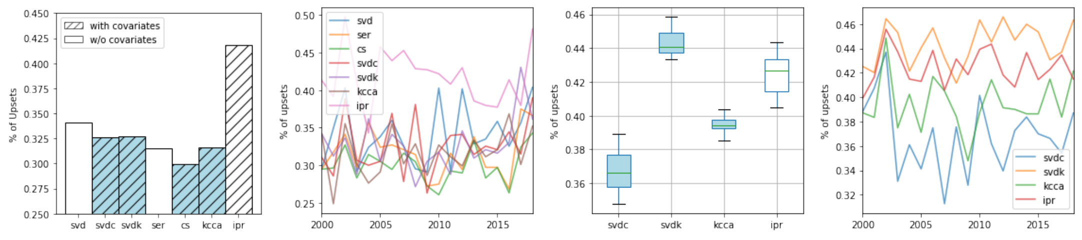
Rank Inference Given a set of pairwise comparisons with corresponding item features, we are interested in finding the optimal ranking that best describes the given set of comparisons.

Rank Prediction In addition to the given pairwise comparisons and item attributes on already *seen* items, we are interested in predicting the rank of *unseen* items where only side information about them is available. In our experiments, we perform a 70/30 train-test split to our items, and train the model on the induced subgraph from the training set. We then predict the ranking for the unseen items using their features. The random splitting is repeated 20 times in order to obtain confidence bounds. Since SVDN, SER and CS do not connect the features with the ranking vector directly, they cannot be used for prediction.



(a) Inference on seen lizards (b) Prediction on unseen lizards (c) Inference on seen Pokémon (d) Prediction on unseen Pokémon

Figure 2: Percentage of upsets (the lower the better) in the Flatlizard competition and Pokémon Battle network, under the rank inference and rank prediction setup, respectively. Proposed algorithms are colored in blue.



(a) Inference on seen NFL teams (Upsets averaged over 2000-2018) (b) Yearly % of upsets (Inference) (c) Prediction averaged over years (d) Averaged yearly % upsets (Prediction)

Figure 3: Comparison of algorithms for the NFL Competition data set (2000 - 2018); both rank inference and rank prediction are studied. Proposed algorithms are colored in blue.

FlatLizard Competition The data is collected at Augrabies Falls National Park (South Africa) in September-October 2002 [Whiting et al., 2009], on the contest performance and background attributes of 77 male flat lizards (*Platysaurus broadleyi*). The results of exactly 100 ordinal contests were recorded (sparsity = 0.017), along with 18 physical measurements made on each lizard, such as *weight* and *head size*. After BAHSIC, we retain 9 of such features.

Figure 2a and 2b showcase the performance of the algorithms. In the inference setting (2a), focusing on CS, SVDC and SVDK, we see a clear improvement in comparison to their covariate-free respective versions. In particular, CS performed the best, followed by KCCA. For the prediction setting (2b), all our methods overperform the benchmark method (ipr), with SVDC and KCCA achieving the best recovery scores. Overall, we observe that the features are related to the rankings in a nonlinear fashion, and that our kernelised algorithms are able to capture such available side information.

Pokémon Battle Our next data set comes from a Kaggle competition. Approximately 45,000 battles between 800 Pokémon are recorded in ordinal format (with sparsity equal to 0.071). We also have available 25 features for each Pokemon, such as their *type*, *attack* and *defense*. The battles are generated by a custom algorithm written by the Kaggle host that closely reflects the game mechanics. After BAHSIC, we retain 6 of such features.

The results are presented in Figure 2c and 2d. It is no surprise that SVDC performed the best in both *inference* and *prediction* tasks, since features such as *attack* and *defense* are linearly related to the strength of a Pokémon.

National Football League 2000 - 2018 Our final real data set comes from sports, and contains the outcome of National Football League (NFL) matches during the regular season, for the years 2000 - 2018. In addition, 256 matches per year between 32 teams, along with 18 performance metrics, such as *yards per game* and *number of fumbles* are recorded. After the BAHSIC test, 6 features are retained.

For the inference task, as illustrated in Figure 3a, CS achieves the smallest fraction of upsets, followed by KCCA and SER. For prediction, as shown in Figure 3c, SVDC was the best performer, followed by KCCA. This is indeed no surprise, as the features we collected are again linearly related to the strength of the teams.

4.3 Fair Ranking Experiment

We demonstrate the fair ranking algorithm derived in Section 3.3 using the *Communities and Crime* [Redmond et al., 2002] data set. We will create a pseudo ranking data set out of the original input, by comparing the capita violent crime rate between each community, eg. community i is preferred over community j if i has a lower crime rate etc. Socio-economical features, such as *median family income* and *family size*, are used as predictive features. Furthermore, we also treat *race* as the sensitive variable for our fair ranking problem.

Figure 4 illustrates the connection between the sensitive feature with the ranking vector of SVDKFAIR-RANK and SVDKCOV-RANK. We observe that, by adding the HSIC regulariser, the number of upsets increases slightly. However, the right plot of Figure 4 shows that SVDKCOV-RANK has learned a very strong correlation between the ranking values and the sensitive variable, while SVDKFAIR-RANK does not have such an undesirable behaviour. In practice, our fair ranking method can mitigate the unfairness of the problem by trading it off for the number of upsets (possibly computed on biased comparisons), by tuning a suitable hyperparameter λ.

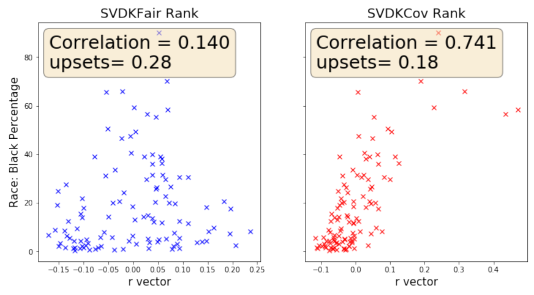


Figure 4: Illustration of the relationship between the sensitive variable *race* and the recovered ranking vector r, for the SVDKFAIR-RANK and SVDKCOV-RANK algorithms.

5 Conclusions and future directions

We considered the problems of ranking given a subset of noisy pairwise comparisons in light of item covariate information. We proposed three lines of spectral ranking methods to solve the ranking with covariate task, each with different strengths and weaknesses demonstrated in the experiments. Furthermore, we extended SVDKCOV-RANK to a fair ranking setting using the statistical parity formalism. In addition, a feature selection method for the ranking problem using dependence maximisation is also demonstrated.

We have presented an extensive set of numerical experiments on three real data sets, which altogether contain 21 distinct comparison graphs, showcasing the competitiveness of our approach compared to other state-of-the-art algorithms from the literature. Moreover, we demonstrated how our algorithms are able to predict the ranking of new items, which previous spectral methods cannot handle, while the existing method (IPR) generally has poor performance. In particular, C-SERIAL-RANK and

SVDCov-Rank were shown to perform particularly well on many of the inference tasks, often achieving the best performance in terms of the number of upsets induced by the recovered ranking. When it comes to prediction, SVDCov-Rank was the best overall performer, followed by KCCA-Rank. The latter outperforms the former when there exists a strong non-linearity between the features and the ranking problem.

There are several avenues for future work. An interesting direction would be to incorporate more general type of side information into the ranking problem, e.g., prior information on partially observed player relationships. Yet another relevant direction is the extraction of partial rankings using the side covariates. In many real world scenarios, aiming for a global ordering of the items is unrealistic, and one is often interested in uncovering partial orderings that reflect accurately various subsets/clusters of a less homogeneous population of items/players. Leveraging features for the discovery of latent structures behind cyclical and inconsistent preferences is also an interesting direction to explore.

References

Jonathan E Atkins, Erik G Boman, and Bruce Hendrickson. A spectral algorithm for seriation and the consecutive ones problem. *SIAM Journal on Computing*, 28(1):297–310, 1998.

Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

M. Cucuringu. Sync-Rank: Robust Ranking, Constrained Ranking and Rank Aggregation via Eigenvector and Semidefinite Programming Synchronization. *IEEE Transactions on Network Science and Engineering*, 3(1):58–79, 2016.

Alexandre d'Aspremont, Mihai Cucuringu, and Tyagi Hemant. Ranking and synchronization from pairwise measurements via SVD. *arXiv:1906.02746*, 2019.

F. Fogel, A. d'Aspremont, and M. Vojnovic. Serialrank: Spectral ranking using seriation. In *Advances in Neural Information Processing Systems 27*, pages 900–908, 2014.

Fajwel Fogel, Alexandre d'Aspremont, and Milan Vojnovic. Spectral ranking using seriation. *Journal of Machine Learning Research*, 17(88):1–45, 2016.

Kenji Fukumizu, Francis R Bach, and Arthur Gretton. Statistical consistency of kernel canonical correlation analysis. *Journal of Machine Learning Research*, 8(Feb):361–383, 2007.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(Mar):723–773, 2012.

Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Schölkopf. Measuring statistical dependence with Hilbert-Schmidt norms. In *International conference on algorithmic learning theory*, pages 63–77. Springer, 2005.

Arthur Gretton, Kenji Fukumizu, Choon H Teo, Le Song, Bernhard Schölkopf, and Alex J Smola. A kernel statistical test of independence. In *Advances in neural information processing systems*, pages 585–592, 2008.

Daniel Kahneman and Amos Tversky. On the interpretation of intuitive probability: A reply to jonathan cohen. 1979.

Zhu Li, Adrian Perez-Suay, Gustau Camps-Valls, and Dino Sejdinovic. Kernel dependence regularizers and gaussian processes with applications to algorithmic fairness. 2019.

R.. Ducan Luce. *Individual Choice Behavior a Theoretical Analysis*. john Wiley and Sons, 1959.

Sahand Negahban, Sewoong Oh, and Devavrat Shah. Rank centrality: Ranking from pairwise comparisons. *Operations Research*, 65(1):266–287, 2016.

UN Niranjan and Arun Rajkumar. Inductive pairwise ranking: going beyond the n log (n) barrier. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Adrián Pérez-Suay, Valero Laparra, Gonzalo Mateo-García, Jordi Muñoz-Marí, Luis Gómez-Chova, and Gustau Camps-Valls. Fair kernel learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 339–355. Springer, 2017.

Michael Redmond and Alok Baveja. A data-driven software tool for enabling cooperative information sharing among police departments. *European Journal of Operational Research*, 141(3):660–678, 2002.

David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE signal processing magazine*, 30(3):83–98, 2013.

Amit Singer and H-T Wu. Vector diffusion maps and the connection laplacian. *Communications on pure and applied mathematics*, 65(8):1067–1144, 2012.

Le Song, Alex Smola, Arthur Gretton, Justin Bedo, and Karsten Borgwardt. Feature selection via dependence maximization. *Journal of Machine Learning Research*, 13(May):1393–1434, 2012.

Martin J Whiting, Jonathan K Webb, and J Scott Keogh. Flat lizard female mimics use sexual deception in visual but not chemical signals. *Proceedings of the Royal Society B: Biological Sciences*, 276(1662):1585–1591, 2009.

Dingdong Yi, Xinran Li, and Jun S Liu. Bayesian aggregation of rank data with covariates and heterogeneous rankers. *arXiv preprint arXiv:1607.06051*, 2016.