

BAYESIAN MODELLING AND ANALYSIS OF
RANKED DATA

STEPHEN RICHARD JOHNSON

Thesis submitted for the degree of
Doctor of Philosophy



*School of Mathematics, Statistics & Physics
Newcastle University
Newcastle upon Tyne
United Kingdom*

January 2019

Acknowledgements

I would like to thank my supervisors, Daniel Henderson and Richard Boys, without whom this thesis would not have been possible. I will be forever grateful not only for their invaluable advice, but also for their enduring support and patience, particularly in times of turmoil. Also, I owe thanks to my mum and dad, along with the rest of my family, for all their love, kindness and encouragement over the course of my postgraduate studies. Finally, to all my friends and colleagues in the School of Mathematics, Statistics and Physics who made my time at Newcastle University so enjoyable.

Abstract

Ranked data are central to many applications in science and social science and arise when *rankers* (individuals) use some criterion to order a set of *entities*. Such rankings are therefore equivalent to permutations of the elements of a set. The majority of models for ranked data rely on a strong assumption of homogeneity, such as all rankers sharing the same view on preferences of the entities. The aim of this thesis is to develop a richer class of models which can reveal any plausible subgroup structure within the data both for rankers and entities.

We begin by looking at the Plackett–Luce model, an extension of the Bradley–Terry model for paired comparisons. First this model is extended to cater for when rankers do not report a full ranking of all entities. For example, they might only report their top five ranked entities after seeing some or all entities. Another issue is that most work in this area assumes that all rankers are equally informed about the entities they are ranking. Often this assumption will be questionable and so we develop a model which allows rankers to have differing reliability. This model, the Weighted Plackett–Luce model, allows for such heterogeneity through a novel two component mixture model defined by augmenting the Plackett–Luce model.

The idea that rankers may be heterogeneous in their beliefs about entities is not new. However, there might be groups of rankers with each group sharing the same view about entities. Generally the number of such groups will not be known and so we investigate the possibility of such group structure by using a Dirichlet process mixture of Weighted Plackett–Luce models. It can also be useful to assess whether some entities are exchangeable, that is, whether there is also entity clustering within each ranker group, an issue that has received little attention in the literature. We extend the model further to explore both ranker and entity clustering by adapting the Nested Dirichlet process. The resulting model is a Weighted Adapted Nested Dirichlet (WAND) process mixture of Plackett–Luce models. Posterior inference is conducted via a simple and efficient Gibbs sampling scheme. The richness of information in the posterior distribution allows for inference about many aspects of the clustering structure both between ranker groups and between entity groups (within ranker groups), in contrast to many other (Bayesian) analyses. The methodology is illustrated using several simulation studies and real data examples.

Finally, we relax the assumption of a known ranking process underpinning these models by looking at the recently developed Extended Plackett–Luce model. This model allows inference for the order in which a homogeneous set of rankers assign entities to ranks. Analysis of this model is challenging but we have found that using Metropolis coupled Markov chain Monte Carlo (MC^3) methods can provide adequate mixing over the high dimensional space of all possible permutations when the number of entities is not small.

Contents

1	Introduction	1
1.1	Introduction	1
1.1.1	Thesis aims	4
1.1.2	Outline of thesis	5
1.2	Bayesian inference	7
1.2.1	Bayes' Theorem	7
1.3	Markov chain Monte Carlo	8
1.3.1	The Metropolis-Hastings algorithm	8
1.3.2	Tuning Metropolis-Hastings algorithms	11
1.3.3	The Gibbs sampler	13
1.3.4	Block updates	14
1.3.5	Convergence	15
1.3.6	Analysing posterior samples	16
1.4	Data augmentation	16
2	Analysis of homogeneous ranked data	19
2.1	Introduction	19
2.2	The Plackett–Luce model	20
2.2.1	Top–M rankings	23
2.2.2	Identifiability issues	24
2.2.3	Rescaling	24
2.2.4	Ties	24
2.2.5	Simulating data from the modified Plackett–Luce model	26

2.3	Bayesian inference	27
2.3.1	Prior specification and latent variables	27
2.3.2	Full conditional distributions	29
2.3.3	MCMC - Gibbs sampling via latent variables	31
2.4	Simulation study	32
2.4.1	Posterior analysis	33
2.5	The Weighted Plackett–Luce model	38
2.5.1	Simulating data from the Weighted Plackett–Luce model	40
2.6	Bayesian inference	40
2.6.1	Prior specification and latent variables	41
2.6.2	Full conditional distributions	42
2.6.3	MCMC - Gibbs sampling via latent variables	43
2.7	Simulation study	45
2.7.1	Posterior analysis	45
2.8	Summary	50
3	Analysis of heterogeneous ranked data	51
3.1	Introduction	51
3.2	Finite mixture models	51
3.3	The Dirichlet process	54
3.3.1	Alternative representations of the Dirichlet process	57
3.3.2	Generating a realisation of a Dirichlet process	59
3.3.3	Generating model parameters consistent with a Dirichlet process prior	61
3.3.4	Generalised Dirichlet process	63
3.4	Dirichlet process mixture models	64
3.4.1	Bayesian inference for DPMM	65
3.4.2	Allowing for uncertainty on the concentration parameter	66
3.5	Uncovering heterogeneity between rankers	67
3.5.1	The model	68

3.5.2	Simulating data from the Dirichlet process mixture of Weighted Plackett–Luce models	68
3.5.3	Prior specification and latent variables	70
3.5.4	Full conditional distributions	71
3.5.5	MCMC using Neal’s Algorithm 8	74
3.5.6	Simulation study – revisiting Dataset 2	75
3.6	Uncovering entity subgroups within a ranker group	81
3.6.1	The model	82
3.6.2	Simulating data from the Weighted Plackett–Luce model with entity clustering	83
3.6.3	Prior and latent variable specification	84
3.6.4	Full conditional distributions	86
3.6.5	MCMC using Neal’s Algorithm 8	88
3.6.6	Simulation study – revisiting Dataset 1	89
3.7	Summary	95
4	The Bayesian WAND	97
4.1	Introduction	97
4.2	Two-way clustering	97
4.3	The Adapted Nested Dirichlet Process (ANDP) prior	99
4.3.1	Generating prior samples (stick-breaking representation)	101
4.3.2	Exploring the ANDP prior	103
4.4	The model	104
4.5	A conditional sampling approach	105
4.5.1	Simulating data from the WAND model	105
4.5.2	Prior specification and latent variables	107
4.5.3	Full conditional distributions	108
4.5.4	Label switching moves	114
4.5.5	Ranker allocations	115
4.5.6	Entity allocations	118
4.5.7	MCMC – a conditional sampler	120

4.5.8	A brief summary	122
4.6	A marginal sampling approach	122
4.6.1	Sampling marginally from the ANDP prior	124
4.6.2	Simulating data from the WAND model	125
4.6.3	Prior specification and latent variables	126
4.6.4	Full conditional distributions	127
4.6.5	MCMC – a marginal sampler	129
4.7	Simulation studies	132
4.7.1	Study 1	132
4.7.2	Study 2	140
4.8	Summary	144
5	Real data analyses	147
5.1	Roskam’s data set	147
5.1.1	Prior sensitivity analysis	150
5.2	NBA study	153
5.2.1	Prior sensitivity analysis	158
5.3	Summary	160
6	The Extended Plackett–Luce model	161
6.1	Introduction	161
6.2	The Extended Plackett–Luce model	162
6.2.1	Simulating data from the Extended Plackett–Luce model	165
6.2.2	Identifiability of the ranking process	166
6.3	Likelihood information about the choice order	170
6.3.1	MM Algorithm	171
6.3.2	Simulation study	172
6.4	Inference – a Bayesian approach	174
6.4.1	Prior specification and latent variables	175
6.4.2	Full conditional distributions for λ , Z	176
6.4.3	Full conditional distribution for σ	177

6.4.4	Metropolis-Hastings proposals for σ	177
6.4.5	Further considerations	183
6.4.6	A Metropolis-within-Gibbs algorithm for the EPL model	185
6.4.7	Simulation study – Metropolis-within-Gibbs	186
6.4.8	Metropolis-Hastings proposals for λ	189
6.4.9	A Metropolis-Hastings algorithm for the EPL model	190
6.4.10	Simulation study – Metropolis-Hastings	191
6.5	Metropolis coupled Markov chain Monte Carlo	193
6.5.1	The advantage of targeting tempered densities	194
6.5.2	Parallel tempering	196
6.5.3	General algorithm outline	197
6.5.4	Tuning a MC ³ sampling scheme	198
6.5.5	Parallel Metropolis coupled Markov chain Monte Carlo	199
6.6	Inference – a Bayesian approach (revisited)	201
6.6.1	A pMC ³ algorithm for the EPL model	201
6.6.2	Simulation study	202
6.7	Summary	205
7	Conclusions	207
7.1	Future work	209
A	Miscellaneous	213
A.1	Derivation of the FCD for DP concentration parameter α	213
B	Datasets	217

List of Figures

2.1	Trace plots of the log complete data likelihood for Datasets 1 and 2 from left to right respectively.	34
2.2	Boxplots summarising the marginal posterior densities for each $\log \lambda_k$ given that $\lambda_{20} = 1$. The densities in each case are shown in white and red for Datasets 1 and 2 respectively. The blue crosses depict the true values from which these data were simulated (log scale).	35
2.3	Trace plots of the log complete data likelihood for Analyses 1 and 2 ($p_i = 0.5, 0.8$) from left to right respectively.	46
2.4	$\Pr(w_i = 1 \mathcal{D})$ – Posterior probability that ranking i is informative under each analysis (Analysis 1: $p_i = 0.5$, Analysis 2: $p_i = 0.8$). Rankings which are random permutations (41–50) are shown in red. Numbers shown denote the Kendall-tau distance between each ranking and $\hat{\mathbf{x}}$	47
2.5	Boxplots summarising the marginal posterior densities for each $\log \lambda_k$ given that $\lambda_{20} = 1$. The boxplots in each case are shown in white and red for Analysis 1 and 2 ($p_i = 0.5, 0.8$) respectively. The blue crosses depict the true values, λ_k , from which these data were simulated.	47
2.6	Boxplots summarising the marginal posterior densities for each $\log \lambda_k$ given that $\lambda_{20} = 1$. The boxplots in each case are shown in white for Analysis 1 under our Weighted Plackett–Luce model and in red for the analysis of Dataset 1 under the standard Plackett–Luce model. The blue crosses depict the true values, λ_k , from which these data were simulated.	48
3.1	Multiple realisations from a Dirichlet process with $G_0 = N(0, 1)$ and $\alpha = 1, 5, 10, 50, 100$ from top to bottom respectively.	56
3.2	Empirical CDF from multiple realisations from a Dirichlet process with $G_0 = N(0, 1)$ and $\alpha = 1, 5, 10, 50, 100$ from top to bottom respectively.	57

3.3	Trace plots of the log complete data likelihood for Analyses 1, 2: $p_i = 0.5, 0.8$ (top left and right respectively) and Analysis 3: $p_i = 1$ (bottom)	76
3.4	Complete linkage dendograms based on the dissimilarities between each pair of rankers for Analyses 1–3 from top to bottom respectively.	78
3.5	$\Pr(w_i = 1 \mathcal{D})$ – Posterior probability that ranking i is informative under each analysis (Analysis 1: $p_i = 0.5$, Analysis 2: $p_i = 0.8$). Rankings which are random permutations (41–50) are shown in red.	80
3.6	Trace plots of the log complete data likelihood for Analyses 1 and 2 (top left, right) and Analyses 3 and 4 (bottom left, right).	92
3.7	$\Pr(w_i = 1 \mathcal{D})$ – Posterior probability that ranking i is informative under each analysis (Analysis 1: $a_\alpha = b_\alpha = 1$, Analysis 2: $a_\alpha = b_\alpha = 3$). Colours distinguish between the different priors on α	92
3.8	$\Pr(N^e = i \mathcal{D})$ – Marginal posterior distribution of the number of entity clusters for each analysis.	93
3.9	Dendograms of entity clustering for Analyses 1, 2 (top left and right) and Analyses 3, 4 (bottom left and right).	94
4.1	Comparison of non-parametric prior distributions for two-way clustering . .	101
4.2	Number of ranker and entity clusters under the Adapted Nested Dirichlet Process prior for various values of concentration parameters α and γ	104
4.3	Plots of the posterior probability $\Pr(w_i = 1 \mathcal{D})$ that ranker i is informative for both scenarios of prior on their ability: $p_i = 0.5$ (left column) and $p_i = 0.9$ (right column). The top row of plots show the comparison between the restricted (*) and full (unrestricted) analyses for Dataset 3. Plots in the middle row are those for the full analyses using Dataset 3, with the corresponding plots using Dataset 4 in the bottom row.	134
4.4	Dendograms for ranker clustering within Dataset 4 under a complete analysis for $p_i = 0.5$ (left plot) and $p_i = 0.9$ (right plot).	136
4.5	Dendograms for entity clustering for Dataset 3 (top) and Dataset 4 (bottom) conditional on a single ranker cluster under both prior specifications for the complete analyses.	138
4.6	Posterior probabilities P_5 for all analyses, P_{10} for all except the top–5 case and P_{15} for the top–15 and complete analyses. The analyses of Dataset 3 and Dataset 4 are shown on the upper and lower row respectively for each prior choice of p	139

4.7	Dendrograms of entity clustering (conditional on 2 ranker clusters) in ranker cluster 1 (left) and ranker cluster 2 (right) for the analysis of Dataset 4 with $p_i = 0.9$	140
4.8	Plot of the posterior probability $\Pr(w_i = 1 \mathcal{D})$ that ranker i is informative (left), colours distinguish between the “true” ranker clusters. Dendrogram (complete linkage) computed using the dissimilarity Δ_{ij} between rankers i and j (right).	142
4.9	Dendrograms showing the dissimilarity between entities within ranker clusters 1 (left) and 2 (right), conditional on two ranker clusters ($N^r = 2$).	143
5.1	Roskam’s dataset: Dendrogram (left) showing the ranker cluster structure along with the posterior probability, $\Pr(w_i = 1 \mathcal{D})$, for each ranker i (right).	149
5.2	Prior and marginal posterior densities for the number of entity clusters within each ranker cluster (conditional on two ranker clusters).	149
5.3	Roskam’s dataset: Dendrograms showing the entity clustering structure within ranker cluster 1 and 2 (left and right respectively) conditional on two ranker clusters.	150
5.4	Roskam’s dataset: Dendrogram (left) showing the cluster structure of the rankers along with the posterior probability $\Pr(w_i = 1 \mathcal{D})$ for each ranking i (right) for $p_i = 0.85, 0.75, 0.65$ (from top to bottom respectively).	151
5.5	Roskam’s dataset: Prior and marginal posterior densities for the number of rankers clusters (left plot) and the number of entity clusters within each ranker cluster, conditional on two ranker clusters, (right plot) for $p_i = 0.85, 0.75, 0.65$	152
5.6	Roskam’s dataset: Dendrograms of entity clustering structure within ranker cluster 1 (left) and ranker cluster 2 (right). These are shown for each prior specification, $p_i = 0.85, 0.75, 0.65$, from top to bottom respectively.	152
5.7	NBA dataset: Dendrogram (left) showing the clustering structure of rankers and highlighting those rankers with $\Pr(w_i = 1 \mathcal{D}) < 0.25$. Plot (right) of the posterior probabilities $\Pr(w_i = 1 \mathcal{D})$ for each ranker, with vertical lines separating the self-certified groups.	154
5.8	Prior and marginal posterior densities for the number of entity clusters within each ranker cluster (conditional on two ranker clusters).	155

5.9	NBA dataset: Dendrograms showing the entity cluster structure within ranker clusters 1 and 2 (left and right respectively) conditional on two ranker clusters.	155
5.10	The probability P_{16} that each entity is in the top–16 under the WAND (\times) model, and the probabilities that each entity is a relevant entity under BARD (\cdot). The vertical line separates out the teams that actually reached the top–16 playoffs.	157
5.11	NBA dataset: Dendrogram (left) showing the clustering structure of rankers and highlighting those rankers with $\Pr(w_i = 1 \mathcal{D}) < 0.25$. Plot (right) of the posterior probabilities $\Pr(w_i = 1 \mathcal{D})$ for each ranker, with vertical lines separating the self-certified groups. The top row shows the results for the “staggered” choice of p_i and the bottom row shows the corresponding results when $p_i = 0.5$ for all rankers.	159
5.12	NBA dataset: Prior and marginal posterior densities for the number of rankers clusters (left plot) and the number of entity clusters within each ranker cluster, conditional on two ranker clusters, (right plot) for “staggered” choice of p_i and $p_i = 0.5$	159
5.13	NBA dataset: Dendrograms showing the entity cluster structure within ranker clusters 1 and 2 (left and right respectively) conditional on two ranker clusters for “staggered” choice of p_i and $p_i = 0.5$, top and bottom respectively.	160
6.1	$\pi(\mathcal{D} \hat{\boldsymbol{\lambda}}_j, \boldsymbol{\sigma}_j)$: maximised log-likelihood given each choice order $\boldsymbol{\sigma}_j$ and the respective MLE $\hat{\boldsymbol{\lambda}}_j$ for $j = 1, \dots, K!$	173
6.2	Trace plots of the log complete data likelihood (left) and the marginal posterior $\pi(\boldsymbol{\sigma} \mathcal{D})$ (right) for chains 1 to 5 from top to bottom respectively (Metropolis-within-Gibbs approach).	187
6.3	Trace plots of the log complete data likelihood (left) and the marginal posterior $\pi(\boldsymbol{\sigma} \mathcal{D})$ (right) for chains 1 to 5 from top to bottom respectively (Metropolis-Hastings approach).	192
6.4	Subset of the marginal posterior $\pi(\boldsymbol{\sigma} \mathcal{D})$ showing the 25 choice orders with highest posterior support from chains 1 to 5 (read from left to right)	193
6.5	$\pi(\theta)$: density plot of an equally weighted two-component normal mixture with component means of ± 2 and standard deviations of 0.05.	194
6.6	Tempered densities $\pi(\theta)^{1/T}$ for $T \in \{1, 2, 4, 8, 16, 32, 64, 128\}$	195

6.7 Trace plot of the log-likelihood (left) and the marginal posterior $\pi(\boldsymbol{\sigma} \mathcal{D})$ (right)	203
6.8 Subset of the marginal posterior $\pi(\boldsymbol{\sigma} \mathcal{D})$ showing the 25 choice orders with highest posterior support (red denotes choice order used to simulate these data)	204
6.9 Boxplots summarising the marginal posterior densities for each $\log \lambda_k$ given that $\lambda_{10} = 1$. The densities in each case are shown in white and red for those obtained under the choice order with the largest posterior support and the true choice order, respectively. The blue crosses depict the true values from which these data were simulated (log scale).	205
6.10 Boxplots summarising the marginal posterior densities for each $\log \lambda_k$ given that $\lambda_{10} = 1$. The densities in each case are shown in white and green for those obtained under the EPL model (for the choice order with the largest posterior support) and those obtained under the standard Plackett–Luce model ($\boldsymbol{\sigma} = (1, \dots, K)$) respectively. The blue crosses depict the true values from which these data were simulated (log scale).	205

List of Tables

1.1	Rank and order vector representations of the same ranking	2
2.1	Ranking types	27
2.2	Aggregate rankings under our analysis of Datasets 1 and 2 along with the corresponding posterior means (denoted $\bar{\lambda}$). The value of λ which was used to simulate these data is also reproduced for ease of comparison.	36
2.3	Aggregate rankings under the Weighted Plackett–Luce model for the analysis of Dataset 2 (for both analyses $p_i = 0.5, 0.8$) along with the corresponding posterior means. To ease comparison, the results from Table 2.2 (standard Plackett–Luce analyses) are also given. The table also contains the relative probability of the aggregate rankings in comparison to a uniform ranking, $r_i = K! \Pr(X = \mathbf{x}_i^{\text{agg}} \bar{\lambda}_i)$	49
3.1	Posterior probabilities of the number of ranker clusters, $\Pr(N^r = i \mathcal{D})$, for each of the three analyses. The expectation and standard deviation of the marginal posterior distribution are also shown along with the prior distribution. The modal values are highlighted in bold.	77
3.2	Aggregate rankings under the infinite mixture of Weighted Plackett–Luce model for the analysis of Dataset 2 (for analyses 1–3; $p_i = 0.5, 0.8, 1$) along with the corresponding posterior means. The results from Table 2.2 (homogeneous standard Plackett–Luce analyses) are also given to facilitate comparison.	81
3.3	Prior probabilities, $\Pr(N^e = j)$, of the number of entity clusters for each analysis (top) and the prior expectations and standard deviations of both the number of entity clusters and the concentration parameter (bottom). Modal values are highlighted in bold.	91

3.4 Marginal posterior means of the skill parameters for each analysis. The aggregate ranking is the same under all analyses.	95
4.1 Posterior distribution of the number of ranker clusters N^r for restricted (*) and full (unrestricted) analyses. Numbers in bold indicate modal values. . .	135
4.2 Posterior distribution of the number of entity clusters, conditional on a single ranker cluster, for restricted (*) and full (unrestricted) analyses. Numbers in bold indicate modal values.	137
4.3 Posterior distribution of the number of entity clusters, conditional on two ranker clusters, for each analysis of Dataset 4 with $p_i = 0.9$. Numbers in bold indicate modal values.	140
4.4 True allocation of entities in to clusters, along with the corresponding true parameter value for each of the entity clusters.	141
4.5 Posterior distribution of the number of entity clusters, conditional on two ranker clusters.	142
4.6 Posterior preference orderings within ranker clusters 1 and 2 (conditional on two ranker clusters) and the overall/aggregate ranking, with mean (and standard deviation) of their skill parameters.	144
5.1 Prior and posterior distribution of the number of ranker clusters (to 2 d.p.).	148
5.2 Roskam's dataset: entity rankings by posterior mean within ranker cluster (conditional on two ranker clusters). Rank 1 corresponds to the entity most preferred within each cluster.	150
5.3 NBA analysis: Posterior preference orderings within ranker clusters 1 and 2 (conditional on two ranker clusters) and the overall/aggregate ranking, with mean (and standard deviation) of their skill parameters. The horizontal lines indicate the MAP entity clustering within ranker clusters. The numbers at the bottom are the number of occurrences in which the MAP clustering was observed (out of 8038 iterations with two rankers clusters). .	156
6.1 Probabilities that each entity is assigned to a specific rank for the standard Plackett–Luce model with $\lambda = (5, 4, 3, 2, 1)$	168
6.2 Cumulative probabilities of each entity being ranked no lower than (left) and no higher than (right) or equal to each position. Entry i, j corresponds to $\Pr(j \in x_{1:i})$ (left) and $\Pr(j \in x_{i:K})$ (right).	169

6.3	Probabilities of each entity being ranked within each position. Entry i, j corresponds to $\Pr(x_i = j)$, that is, the probability entity j receives rank i	170
6.4	A subset of the ranking of choice orders (permutations) based on the value of the log-likelihood evaluated at the corresponding MLE for the skill parameters ($\pi(\mathcal{D} \hat{\lambda}_j, \sigma_j)$).	174
6.5	Permutations (choice orders) used for the initialisation of each chain	186
6.6	Acceptance rates for each of the 5 proposal mechanisms for the choice order σ	188
6.7	Theoretical optimal job allocation across cores with total and relative execution time (assuming job takes a unit time).	200
B.1	Dataset 1 used in standard PL analysis.	218
B.2	Additional 10 uninformative rankings used to form Dataset 2.	219
B.3	Dataset 3 used in simulation study 1 under WAND.	220
B.4	Additional 10 uninformative rankings used to form Dataset 4.	221
B.5	Dataset 5 used in simulation study 2 under WAND. Vertical lines separate the rankings within each different ranker group.	222
B.6	Roskam's psychology data	223
B.7	NBA data	224
B.8	Dataset 6 used in the simulation studies for the Bayesian analysis of the Extended Plackett–Luce model.	225
B.9	Additional 20 rankings used to form Dataset 6	226

Chapter 1

Introduction

1.1 Introduction

Although often unnoticed, rankings appear in many aspects of everyday life. People rank objects all the time, be it based on personal preference or past experiences; perhaps its our favourite movies, online games, sports teams or even which coffee shop we prefer to visit, the list goes on. In the data age that we live, the ability to analyse ranked data is becoming ever more important. For example, large organisations are interested in the preferences of consumers for advertising purposes and online search engines aim to rank their results in an optimal manner. Ranked data are also a common result of experiments which aim to uncover the attitudes or preferences of a cohort to a particular set of items (Vigneau et al., 1999; Yu et al., 2005; Gormley and Murphy, 2006; Vitelli et al., 2018). Sporting events can also give rise to rankings, with this being particularly common in horse/motor racing or round-robin tournaments where the outcome is an ordering of the teams or individual competitors; see, for example, Henery (1981), Stern (1990) and Caron and Doucet (2012).

This thesis is concerned with preference orderings which arise when *rankers* provide a ranking or ordering for a set of *entities* according to some criterion. Typically, rankers will be individuals although this framework is fairly general and groups, organisations or even sensors could be regarded as rankers, amongst others. There exists almost infinite possibilities for the entities, be it political candidates, world cuisines, universities and so on. Two common representations of ranked data exist which, using the terminology of Marden (1995), we call the *rank* and *order* vector. Both representations portray the same information and so when modelling ranked data one must be careful about which representation is used. Formally, a *rank vector* $\mathbf{y} = (y_1, \dots, y_K)$ of K entities is a list, where the entry y_i indicates the rank given to the i th entity. In contrast, an *order vector*

Rank vector		Order vector	
Entity	y	Entity	x
British	3	Indian	1
Chinese	2	Chinese	2
Italian	4	British	3
Indian	1	Italian	4
Thai	5	Thai	5

Table 1.1: Rank and order vector representations of the same ranking

$\mathbf{x} = (x_1, \dots, x_K)$ of K entities can be thought of as a preference list where the entry x_i contains the label of the entity in position i (with position 1 being most preferred). For example, suppose we ask a ranker to order 5 popular world cuisines, British, Chinese, Italian, Indian and Thai in terms of their preference. If the ranker prefers Indian, then Chinese, then British, then Italian with Thai the least preferred then Table 1.1 shows the corresponding rank and order vectors – note that the rank vector depends on the order in which the entities are listed. We adopt the order vector representation and use the terms *ranking* and *ordering* interchangeably to denote an order vector \mathbf{x} .

Irrespective of the adopted format, rankings are multivariate observations and moreover any particular ranking can be thought of as a permutation of the integers 1 to K ; this perspective can be useful for the development of models for such data. Marden (1995) and more recently Alvo and Yu (2014) provide an overview of the models and statistical literature for ranked data. Many types of models for ranked data exist including parametric, stagewise and distance-based models. Distance-based models rely on the assumption that a modal ranking (of the entities) exists and that rankers are expected to report rankings which are, in some sense, “close” to this modal ordering. Although the distance between two permutations is not well defined, a choice must be made in order to fit this type of model. Two common choices of distance are Kendall’s and Spearman’s distance and these give rise to Mallows’ (1957) ϕ and Mallows’ θ models respectively; details of these models (amongst others) can be found in Flinger and Verducci (1986). Bayesian inference for distance-based models can be problematic, especially when the modal ordering is assumed to be unknown, due to the prohibitive nature of an intractable normalising constant which must be approximated in most cases. Further, distance-based models become increasingly more challenging to fit as the number of entities increases due to the explosion of the size of permutation space. For these reasons we will not consider distance-based models and instead consider parametric ranking models.

Several parametric distributions over the set of permutations have been developed. The so-called stagewise ranking models are examples of parametric rankings models. Stage-

wise models are underpinned by the idea that the ranking process, that is, how a ranker constructs their ordering, can be decomposed into $K - 1$ (dependent) stages. In this thesis we focus predominantly on the popular Plackett–Luce model (Luce, 1959; Plackett, 1975) which assumes the forward order, that is, the assignment of entities to positions in the ranking proceeds sequentially from the most-preferred to the least-preferred item. If the forward ranking process assumption is not plausible then the Reverse Plackett–Luce model provides an alternative choice and both Graves et al. (2003) and Henderson and Kirrane (2018) found this model was more appropriate when modelling NASCAR and Formula 1 races respectively. Further, Mollica and Tardella (2014) proposed the Extended Plackett–Luce model which allows the assumption of an explicit ranking process to be relaxed. Inference for this model is challenging, particularly when the number of entities is not small, as the posterior distribution is extended to also be over permutation space. The current Bayesian solution proposed by Mollica and Tardella (2018) relies on a restricted sample space.

Most models for ranked data treat the information provided by each ranker equally, that is, they assume that each ranker is equally informative. This is a rather strong assumption; it is easy to imagine a situation where some rankers are significantly more informed about the entities in comparison to fellow rankers. Deng et al. (2014) aimed to address this issue and used ranker reliability as part of their BARD (Bayesian Aggregation of Ranked Data) solution. Further, the majority of models for ranked data rely on strong assumptions about the homogeneity of ranked data; the idea that there is an overall consensus view is one such example. More flexible models have been proposed with Gormley and Murphy (2008*a,b*, 2009) and Mollica and Tardella (2014, 2016) considering finite mixtures of Plackett–Luce and related models to allow for different preferences between rankers. This approach was also taken by Vitelli et al. (2018). However they adopted a distance-based model — namely that by Mallows (1957) — rather than the Plackett–Luce model. More flexible infinite mixture models have also been proposed and these approaches allow the number of groups to be inferred rather than be fixed by the analyst; see, for example, Caron et al. (2014). Although these types of models allow for rankers to express different beliefs they also assume that each ranker group can distinguish between each of the entities. However, it is possible that a (homogeneous) group of rankers may not be able to distinguish between some entities, that is, they might believe that some entities are exchangeable. This is an aspect which is often overlooked within the literature and although methods have been proposed (de Leeuw and Mair, 2009; Choulakian, 2016) they often rely on ad-hoc summaries as opposed to a model based approach.

1.1.1 Thesis aims

The main aim of this thesis is to provide flexible models which allow the exploration of (possible) subgroup structure within ranked data. More specifically we aim to identify homogeneous groups of individuals who share similar beliefs along with discovering how some, or indeed all, of these groups may struggle to distinguish between certain entities. Further, we aim to construct models that allow for potential heterogeneity between the abilities of rankers. Emphasis will be placed on efficient inference schemes which enable us to fit our models, under the Bayesian paradigm, in a reasonable amount of computational time. In later sections we increase modelling flexibility further by relaxing the assumption of an explicit ranking process. This is achieved through considering the Extended Plackett–Luce model which has a parameter (representing the ranking process) that is an element of the set of all permutations. Constructing Bayesian inference schemes that can effectively explore large discrete spaces is not straightforward. This problem is even more challenging when such spaces do not exhibit a natural distance measure (permutation space) and so, in this thesis, we aim to provide an effective solution to this issue for reasonably large spaces.

As a starting point, we consider the standard Plackett–Luce model and show how inferences from this model can be affected by even a modest amount of spurious rankings. This result motivates the idea that a suitable model for ranked data should be flexible enough to allow for (potential) heterogeneity between rankers abilities. We propose the Weighted Plackett–Luce model (WPL) which is formed by augmenting the standard Plackett–Luce model with an additional parameter that allows us to handle differing ranker abilities. Taking the WPL model as our building block we then relax the common assumption that the data come from a homogeneous group of rankers, in which each ranker only has fairly minor differences from an overall consensus view. This is achieved by appealing to Bayesian non-parametrics; specifically we propose a Dirichlet process mixture of Weighted Plackett–Luce models. We then consider the notion that a (homogeneous) group of rankers may not be able to distinguish between some entities, that is, they believe some entities are exchangeable. To allow for this we consider an alternative non-parametric prior distribution which allows entities to cluster together. Combining both of these aspects into a single model requires a two-way clustering technique and we focus on the Nested Dirichlet Process (NDP) (Rodriguez et al., 2008). The NDP is not quite suited to clustering entities and so we propose the Adapted Nested Dirichlet process (ANDP) prior. Bayesian inference proceeds under both marginal and conditional approaches, each of which has associated pros and cons. Numerous analyses are preformed on simulated and real data and these show that our model performs well in different scenarios. These studies also highlight the rich (posterior) information available to the analyst as a result of fitting our

model.

Until now each of the proposed models has relied on the underlying assumption of the forward ranking process. In the final chapter we relax the assumption of a known ranking process by looking at the Extended Plackett–Luce model (Mollica and Tardella, 2014). After introducing the Extended Plackett–Luce model, a natural question arises: “is it possible to identify the ranking process given a set of rankings?”. We motivate the identifiability of the ranking process through several examples and consider a maximum likelihood approach to cement this idea further. Constructing suitable (Bayesian) posterior sampling schemes for this model is challenging and, to the best of our knowledge, the only current solution is given by Mollica and Tardella (2018) but this relies on a restricted parameter space. Our aim is to develop MCMC methods capable of exploring the entire parameter space and several sampling algorithms are presented, each of which has varying degrees of success. We found that our final algorithm, which uses Metropolis coupled Markov chain Monte Carlo (MC³), worked well and both the methodology and the MC³ algorithm are described in detail.

1.1.2 Outline of thesis

The remainder of this thesis is organised as follows. In the following sections we provide a brief introduction to Bayesian inference and Markov chain Monte Carlo sampling techniques. A generic Metropolis-Hastings algorithm is outlined and the Gibbs sampler is shown to be a special case. Methods for diagnosing convergence of a Markov chain are discussed and we also consider sensible strategies for handling MCMC output to ensure we obtain a reasonable number of samples from the density of interest. This chapter concludes with a short discussion of data augmentation as this is a particularly useful technique which allows us to use full conditional distributions in closed form when the likelihood is non-standard.

In Chapter 2 we consider the analysis of homogeneous ranked data, that is, we assume that all rankers share similar beliefs about the preference of entities. The Plackett–Luce model (Luce, 1959; Plackett, 1975) and its underpinning assumptions are described in detail. The vanilla Plackett–Luce model is then extended to cater for when rankers do not report a full ranking of all entities (*top* and *partial* rankings) and the associated underlying data generating mechanism is outlined. An efficient MCMC scheme is constructed and we consider a brief simulation study to give a flavour for how posterior inferences can be made. The second half of Chapter 2 is concerned with the notion of ranker reliability. Most work in this area assumes that all rankers are equally informed about the entities they are ranking. Often this assumption will be questionable and so we develop the novel Weighted Plackett–Luce model as this allows us to model rankers with differing reliability through a

two component mixture model. Several simulation studies are considered and we compare posterior inferences from both the standard and Weighted Plackett–Luce models.

Chapter 3 focusses on increasing modelling flexibility so that we can effectively handle heterogeneous ranked data. To do this we appeal to Dirichlet process mixture models which are explored in detail using two well-known representations. Two models are presented. The first considers the idea that rankers may be heterogeneous in their beliefs about entities. This idea is not new and both finite and Dirichlet process mixtures of (standard) Plackett–Luce models are well explored within the literature. We extend this approach slightly by building a model that comprises a Dirichlet process mixture of Weighted Plackett–Luce models. The second model we present is novel and aims to explore whether some entities are exchangeable, that is, whether rankers find it difficult (or impossible) to distinguish between certain groups of entities. This issue has received little attention in the literature and we explore this idea by considering a Dirichlet process mixture over the *skill parameters* where the Weighted Plackett–Luce model is taken to be the ranking distribution. The effectiveness of our model to detect groups of entities is assessed through simulation studies and these conclude the chapter.

In Chapter 4 we combine the aspects of each model presented in the previous chapter, that is, we incorporate both ranker and entity clustering within a single model by appealing to two-way clustering techniques. We focus on the Nested Dirichlet process (Rodriguez et al., 2008) and make a necessary adaptation so that this prior distribution can be used within a ranked data context. The resulting model is a Weighted Adapted Nested Dirichlet (WAND) process mixture of Plackett–Luce models. Both a *conditional* and *marginal* approach to posterior inference are considered with efficient algorithms provided in each case. The methodology is illustrated using several simulation studies and in Chapter 5 we consider two real data examples. For the first real data analysis we use a data set originally collected in 1968 by Roskam and more recently studied by de Leeuw (2006). These data consist of rankings obtained from psychologists within the Psychology Department at the University of Nijmegen (Netherlands). Each psychologist was asked to rank each of 9 sub-areas according to how appropriate they are to their work. The second real data analysis considers a data set taken from Deng et al. (2014) which involved rankings of NBA (National Basketball Association) teams. In their paper, Deng et al. propose a model named “Bayesian Aggregation of Ranked Data” (BARD) and we compare inferences from the WAND model to those from BARD.

In Chapter 6 we relax the assumption of a known ranking process by looking at the recently developed Extended Plackett–Luce model (Mollica and Tardella, 2014). This model contains an additional free parameter (which is a permutation) that allows us to learn about the order in which a homogeneous group of rankers assign entities to

ranks. Constructing a suitable (Bayesian) posterior sampling scheme for this model proved challenging. However we found that using Metropolis coupled Markov chain Monte Carlo (MC³) worked well and both the methodology and the MC³ algorithm are described in detail.

Finally, our conclusions are drawn in Chapter 7 and we also provide some suggested topics for future work.

1.2 Bayesian inference

In this thesis we work almost exclusively within the Bayesian framework (Bernardo and Smith, 1994). In this setting all unknown quantities (parameters, latent variables and so on) are considered to be random variables. A joint probability distribution describes the relationship between the unknown quantities and the (observed) data. The posterior distribution is the (conditional) distribution obtained by conditioning on the observed data and it is this distribution which allows us to make inferences about the unknown quantities (given the data). The posterior is the result of our prior beliefs about the unknown quantities being updated by the observed data through the likelihood function.

1.2.1 Bayes' Theorem

Suppose we have some data $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ and we are interested in learning about a collection of K unknown quantities $\Lambda = (\lambda_1, \dots, \lambda_K)$. The likelihood $L(\Lambda|\mathcal{D}) = \pi(\mathcal{D}|\Lambda)$ is the probability density of the data given the parameters but regarded as a function of the parameters for known data. Note that a “model” is typically specified by a particular likelihood function and this describes how the data are related to the parameters. Given we are working within the Bayesian framework, we must also summarise our (prior) beliefs about the unknown quantities through the choice of a suitably defined prior distribution $\pi(\Lambda)$. The posterior $\pi(\Lambda|\mathcal{D})$ is the density that reflects our updated beliefs about the parameters Λ having observed the data \mathcal{D} and follows from Bayes' Theorem as

$$\pi(\Lambda|\mathcal{D}) = \frac{\pi(\mathcal{D}|\Lambda)\pi(\Lambda)}{\pi(\mathcal{D})}. \quad (1.1)$$

Note that the denominator, $\pi(\mathcal{D}) = \int \pi(\mathcal{D}|\Lambda)\pi(\Lambda)d\Lambda$, is the *marginal likelihood* and is obtained by integrating out (marginalising over) the parameters. Clearly the marginal likelihood does not depend on the parameters Λ and so this is simply a normalising constant which ensures the posterior density integrates to one. It follows that Bayes' Theorem can

be written as

$$\pi(\Lambda|\mathcal{D}) \propto \pi(\mathcal{D}|\Lambda)\pi(\Lambda) \quad (1.2)$$

and so the posterior is proportional to the product of the prior and the likelihood.

Typically, the marginal likelihood $\pi(\mathcal{D})$, and therefore the posterior density $\pi(\Lambda|\mathcal{D})$, is not available in closed form, that is, the posterior is not a standard distribution that can be written down analytically with known marginal distributions and moments. In such cases we must appeal to other methods which allow us to compute the posterior distribution. One of the most popular methods, and the one we shall focus on, is Markov chain Monte Carlo – this is the topic of the next section.

1.3 Markov chain Monte Carlo

Markov chain Monte Carlo (MCMC) is a computational technique that can be used to obtain realisations from the posterior density when it is not available in closed form. In fact, MCMC can be used to obtain realisations from any distribution of interest and so can also be useful for drawing samples from generic high-dimensional distributions. The methodology underpinning MCMC is well studied with countless books and articles published within the literature; see, for example, Chib and Greenberg (1995), Brooks (1998) and Gamerman and Lopes (2006). The general idea is to construct a Markov chain which has stationary distribution $\pi(\cdot)$, which is also known as the *target distribution*. Then, given any initial starting point, providing we run the chain long enough so that it converges (to the target distribution) we can repeatedly update the chain to generate (dependent) samples from the target $\pi(\cdot)$. Clearly within a Bayesian inference setting we wish the target to be the posterior $\pi(\Lambda|\mathcal{D})$. In what follows we discuss a fundamental algorithm, and an associated special case, which allows us to construct Markov chains where the target distribution is the posterior distribution.

1.3.1 The Metropolis-Hastings algorithm

The Metropolis-Hastings algorithm, first proposed by Metropolis et al. (1953) and then further generalised by Hastings (1970), is considered to be the fundamental algorithm used to construct MCMC schemes which target the posterior $\pi(\Lambda|\mathcal{D})$. The notion of a transition kernel, or *proposal density*, is a key idea behind the Metropolis-Hastings algorithm. The (arbitrary) proposal density is denoted $q(\Lambda^*|\Lambda)$ and describes, probabilistically, how to move from a current state Λ to a *proposed* state Λ^* . The (Metropolis-Hastings) algorithm that follows will successively generate a sequence of values $\Lambda^{(1)}, \Lambda^{(2)}, \dots$, which form a Markov chain with target distribution $\pi(\Lambda|\mathcal{D})$.

1. Let the iteration counter be $t = 1$ and initialise the chain to $\Lambda^{(0)} = (\lambda_1^{(0)}, \dots, \lambda_K^{(0)})$ which falls somewhere in the support of $\pi(\Lambda|\mathcal{D})$, that is, so that $\pi(\Lambda^{(0)}|\mathcal{D}) > 0$.
2. Draw Λ^* from the proposal density $q(\Lambda^*|\Lambda^{(t-1)})$.
3. Evaluate the acceptance probability, $p = \min(1, A)$, where

$$A = \frac{\pi(\Lambda^*|\mathcal{D})}{\pi(\Lambda^{(t-1)}|\mathcal{D})} \times \frac{q(\Lambda^{(t-1)}|\Lambda^*)}{q(\Lambda^*|\Lambda^{(t-1)})}.$$
4. Let $\Lambda^{(t)} = \Lambda^*$ with probability p ; otherwise let $\Lambda^{(t)} = \Lambda^{(t-1)}$.
5. Let $t \rightarrow t + 1$ and return to step 2.

Typically we will not be able to evaluate the normalising constant $\pi(\mathcal{D})$ of the posterior density (1.1) and so we might think that we are also unable to evaluate the acceptance rate A . However, as the posterior density features in both the numerator and denominator of the acceptance rate we only need to know the posterior distribution up to a constant of proportionality. It follows that, by using (1.2), we can equivalently express the acceptance probability from Step 3 as $p = \min(1, A)$ where

$$A = \frac{\pi(\mathcal{D}|\Lambda^*)\pi(\Lambda^*)}{\pi(\mathcal{D}|\Lambda^{(t-1)})\pi(\Lambda^{(t-1)})} \times \frac{q(\Lambda^{(t-1)}|\Lambda^*)}{q(\Lambda^*|\Lambda^{(t-1)})}.$$

Note that the choice of proposal density $q(\Lambda^*|\Lambda)$ is completely arbitrary and the Markov chain will target the correct posterior irrespective of the choice made (assuming the support of the proposal distribution is no smaller than the support of the posterior, that is, $q(\cdot|\Lambda) > 0 \forall \Lambda$ where $\pi(\Lambda|\mathcal{D}) > 0$). However, some proposal distributions are better than others in the sense that they lead to a chain which converges rapidly and mixes well, that is, a chain that efficiently explores the support of $\pi(\Lambda|\mathcal{D})$. We now (briefly) describe some of the more common choices of proposal distribution before moving on to discuss the “tuning” of Metropolis-Hastings algorithms in Section 1.3.2.

Random walk Metropolis-Hastings

The random walk Metropolis-Hastings algorithm is where the proposed value is of the form $\Lambda^* = \Lambda + \omega$ where ω is a (vector) of random innovations. Generally ω is chosen to follow a (multivariate) normal distribution with 0 mean and diagonal covariance structure and so $q(\Lambda^*|\Lambda) \sim N_K(\Lambda, \sigma^2 \mathcal{I}_K)$.

Symmetric proposals

A symmetric proposal is any proposal distribution where $q(\Lambda^*|\Lambda) = q(\Lambda|\Lambda^*)$ for all Λ^*, Λ in its support. It follows that, for this type of proposal distribution, the acceptance rate simplifies to $A = \pi(\Lambda^*|\mathcal{D})/\pi(\Lambda|\mathcal{D})$ and so the acceptance probability is independent of the proposal density. Note that the random walk proposal (above) is an example of a symmetric proposal distribution.

Log-normal random walk

A log-normal random walk proposal is particularly useful when the parameters of interest are constrained to be strictly positive. The proposed value is of the form $\Lambda^* = \exp(\log \Lambda + \omega)$ where ω is a random innovation. It follows that, in this case, the proposal distribution is $q(\Lambda^*|\Lambda) \sim \text{LN}_K(\log \Lambda, \sigma^2 \mathcal{I}_K)$ where LN_K denotes the K -dimensional log-normal distribution. The log-normal distribution is not symmetric around its mean so the *proposal ratio* $q(\Lambda|\Lambda^*)/q(\Lambda^*|\Lambda)$ must be computed in order to evaluate the acceptance rate A . It follows that

$$A = \frac{\pi(\Lambda^*|\mathcal{D})}{\pi(\Lambda^{(t-1)}|\mathcal{D})} \times \prod_{k=1}^K \frac{\lambda_k^*}{\lambda_k}.$$

Independence proposals

An independence proposal is a mechanism for generating proposed values Λ^* that are *independent* of the current state of the chain Λ . It follows that $q(\Lambda^*|\Lambda) = q(\Lambda^*)$ and the proposal ratio simplifies to $q(\Lambda)/q(\Lambda^*)$. Further, the acceptance rate in Step 3 of the MH algorithm can be written as

$$A = \frac{\pi(\Lambda^*|\mathcal{D})}{\pi(\Lambda^{(t-1)}|\mathcal{D})} \times \frac{q(\Lambda^{(t-1)})}{q(\Lambda^*)}$$

under this proposal mechanism and so it is clear that we can increase the acceptance probability by choosing $q(\Lambda^*)$ to be as similar as possible to $\pi(\Lambda^*|\mathcal{D})$. Note that if $q(\Lambda^*) = \pi(\Lambda^*|\mathcal{D})$ then $A = 1$ and the proposed value will always be accepted – perhaps not surprising given the proposed value is from the posterior. This idea leads to the Gibbs sampler; further details are provided in Section 1.3.3.

Componentwise updates

In what has been discussed so far we have considered the proposal Λ^* to contain proposed values for all K unknown quantities. However, in practice, it can be difficult to

construct a suitable (K -dimensional) proposal distribution, particularly when the number of (unknown) parameters is large. A solution to this issue is to consider *componentwise* updates, that is, to update each unknown quantity one-at-a-time (conditional on the remaining unknown quantities remaining fixed at their current values). Let $\Lambda_{-k} = (\lambda_1, \dots, \lambda_{k-1}, \lambda_{k+1}, \dots, \lambda_K)$ be the collection of all unknown quantities excluding λ_k then a Metropolis-Hastings algorithm to target the posterior $\pi(\Lambda|\mathcal{D})$ using componentwise updates is as follows.

1. Let the iteration counter be $t = 1$ and initialise the chain to $\Lambda^{(0)} = (\lambda_1^{(0)}, \dots, \lambda_K^{(0)})$ which falls somewhere in the support of $\pi(\Lambda|\mathcal{D})$, that is, so that $\pi(\Lambda^{(0)}|\mathcal{D}) > 0$.
2. Let $\Lambda' = \Lambda^{(t-1)}$ be the current state of the chain, then for $k = 1, \dots, K$
 - (a) draw λ_k^* from the proposal density $q_k(\lambda_k^*|\lambda'_k)$.
 - (b) evaluate the acceptance probability, $p_k = \min(1, A_k)$, where
$$A_k = \frac{\pi(\lambda_k^*, \Lambda'_{-k}|\mathcal{D})}{\pi(\Lambda'|\mathcal{D})} \times \frac{q_k(\lambda'_k|\lambda_k^*)}{q_k(\lambda_k^*|\lambda'_k)}.$$
 - (c) let $\lambda'_k = \lambda_k^*$ with probability p_k .
3. Let $\Lambda^{(t)} = \Lambda'$.
4. Let $t \rightarrow t + 1$ and return to step 2.

Again each (now univariate) proposal distribution $q_k(\cdot)$ can be chosen arbitrarily and so a combination of the previously discussed proposal mechanisms can be used depending on the unknown quantities of interest.

1.3.2 Tuning Metropolis-Hastings algorithms

Key to the implementation of an efficient Metropolis-Hastings algorithm is the choice of proposal distribution(s). An efficient algorithm is one that results in a chain which converges rapidly (to the target distribution) and also mixes well, that is, a chain that efficiently explores the support of $\pi(\Lambda|\mathcal{D})$. For proposal distributions that depend on the current state of the chain (i.e. not independence proposals) it should be clear that the variance of the proposal will determine how the Markov chain explores the sample space. If the variance is too small then the chain will explore the space slowly as, although the proposed values are likely to be accepted, they will only move the chain a small distance from the current state. In contrast, if the variance is too large, then although the proposed values will be a large distance from the current state only relatively few of the proposed

values will be accepted – the chain will therefore remain “stuck” at the same value for many iterations, which is inefficient.

Clearly there exists a trade off between the acceptance probability of proposed values and the distance they allow us to move around the sample space. Given this it would be useful if we could construct proposal distributions so that the proposed values are a reasonable distance from the current state and are also likely to be accepted. Roberts and Rosenthal (2001) suggest that, if the target distribution is Gaussian, the *optimal* acceptance probability (that which maximises the expected squared “jumping” distance) is 0.234. This result was extended to elliptically symmetric targets by Sherlock and Roberts (2009) with Sherlock (2013) later providing a general set of sufficient conditions for which the optimal acceptance probability is 0.234. Further, if the proposal distribution is a normal random walk then it has been suggested by Gelman et al. (1996) amongst others that the variance of ω should be

$$\frac{2.38^2 \text{Var}(\Lambda|\mathcal{D})}{K}.$$

Of course, in general we will not know the posterior variance matrix $\text{Var}(\Lambda|\mathcal{D})$, so an estimate obtained by numerous pilot runs of the algorithm might be used.

Unfortunately there is no hard and fast rule which describes how best to construct suitable proposal distributions in general. The form of the target distribution affects how proposal mechanisms perform and so bespoke proposals are typically required for each scenario. The strategy we suggest is to first choose a proposal distribution which seems sensible *a priori* and then perform numerous iterations of the MH algorithm to calculate the empirical acceptance rate (# proposals accepted/# iterations). If the acceptance rate is too low/high then decrease/increase the variance of the proposal distribution until the acceptance rate is $\simeq 23\%$.

At the beginning of this section we noted that it is only in scenarios where the proposal distribution depends on the current state of the chain that the variance (of the proposal distribution) affects how the Markov chain explores the sample space. By construction independence proposals do not depend on the current state of the chain and so the optimal acceptance probability for this type of proposal is not 0.234. In fact for independence proposals it is advantageous to make the acceptance probability as large as possible, that is, the optimal acceptance probability is one in this case. In other words, we should aim to construct a proposal distribution which is as close to the target (posterior) distribution as possible.

1.3.3 The Gibbs sampler

The Gibbs sampler is a special case of the (componentwise) Metropolis-Hastings algorithm where each of the proposed values λ_k^* are drawn from their corresponding *full conditional distribution*. This technique was first proposed by Geman and Geman (1984) in the context of image processing and only later was it brought to the attention of statisticians by Gelfand and Smith (1990). The full conditional distribution of the k th unknown quantity is $\pi(\lambda_k|\Lambda_{-k}, \mathcal{D})$ and is the conditional distribution of λ_k given all other unknown quantities, and the data. It is often the case that, although the posterior $\pi(\Lambda|\mathcal{D})$ may be intractable, we can obtain the full conditional distribution for each unknown quantity in closed form (and therefore sample from them).

Suppose we are in the scenario where we have a complete set of full conditional distributions, that is, $\pi(\lambda_k|\Lambda_{-k}, \mathcal{D})$ is available in closed form for $k = 1, \dots, K$. Let Λ denote the current state of the Markov chain and recall from Step 2(b) in the (componentwise) MH algorithm that the acceptance rate for each unknown quantity k is

$$A_k = \frac{\pi(\lambda_k^*, \Lambda_{-k}|\mathcal{D})}{\pi(\Lambda|\mathcal{D})} \times \frac{q_k(\lambda_k|\lambda_k^*)}{q_k(\lambda_k^*|\lambda_k)}$$

which we can equivalently express as

$$\begin{aligned} A_k &= \frac{\pi(\lambda_k^*|\Lambda_{-k}, \mathcal{D})\pi(\Lambda_{-k}|\mathcal{D})}{\pi(\lambda_k|\Lambda_{-k}, \mathcal{D})\pi(\Lambda_{-k}|\mathcal{D})} \times \frac{q_k(\lambda_k|\lambda_k^*)}{q_k(\lambda_k^*|\lambda_k)} \\ &= \frac{\pi(\lambda_k^*|\Lambda_{-k}, \mathcal{D})}{\pi(\lambda_k|\Lambda_{-k}, \mathcal{D})} \times \frac{q_k(\lambda_k|\lambda_k^*)}{q_k(\lambda_k^*|\lambda_k)}. \end{aligned}$$

Now if construct an independence proposal where the proposed value for each unknown quantity is drawn from its corresponding full conditional distribution, that is, take $q_k(\lambda_k^*) = \pi(\lambda_k^*|\Lambda_{-k}, \mathcal{D})$ then it follows that

$$A_k = \frac{\pi(\lambda_k^*|\Lambda_{-k}, \mathcal{D})}{\pi(\lambda_k|\Lambda_{-k}, \mathcal{D})} \times \frac{\pi(\lambda_k|\Lambda_{-k}, \mathcal{D})}{\pi(\lambda_k^*|\Lambda_{-k}, \mathcal{D})} = 1$$

and so any proposed value λ_k^* is guaranteed to be accepted (for $k = 1, \dots, K$).

Using this result we obtain the special case of the Metropolis-Hastings algorithm known as the Gibbs sampler. If the full conditionals $\pi(\lambda_k|\Lambda_{-k}, \mathcal{D})$ are available in closed form for $k = 1, \dots, K$ then a Markov chain which targets the posterior $\pi(\Lambda|\mathcal{D})$ is as follows.

1. Let the iteration counter be $t = 1$ and initialise the chain to $\Lambda^{(0)} = (\lambda_1^{(0)}, \dots, \lambda_K^{(0)})$ which falls somewhere in the support of $\pi(\Lambda|\mathcal{D})$, that is, so that $\pi(\Lambda^{(0)}|\mathcal{D}) > 0$.
2. Obtain a new realisation $\Lambda^{(t)} = (\lambda_1^{(t)}, \dots, \lambda_K^{(t)})$ from $\Lambda^{(t-1)}$ by sampling from the full

conditional distributions

$$\begin{aligned}\lambda_1^{(t)} &\sim \pi(\lambda_1 | \lambda_2^{(t-1)}, \lambda_3^{(t-1)}, \dots, \lambda_K^{(t-1)}, \mathcal{D}) \\ \lambda_2^{(t)} &\sim \pi(\lambda_2 | \lambda_1^{(t)}, \lambda_3^{(t-1)}, \dots, \lambda_K^{(t-1)}, \mathcal{D}) \\ &\vdots \\ \lambda_K^{(t)} &\sim \pi(\lambda_K | \lambda_1^{(t)}, \lambda_2^{(t)}, \dots, \lambda_{K-1}^{(t)}, \mathcal{D}).\end{aligned}$$

3. Let $t \rightarrow t + 1$ and return to step 2.

The Gibbs sampler (above) is particularly useful when it is infeasible to directly sample from $\pi(\Lambda | \mathcal{D})$ but sampling from $\pi(\lambda_k | \Lambda_{-k}, \mathcal{D})$ is straightforward. Moreover, unlike the standard Metropolis-Hastings algorithm, we need not construct suitable proposal distributions (which can be difficult in practice) as they are simply the full conditional distributions. The algorithm we have outlined is known as the *fixed sweep* Gibbs sampler and is often used in practice as it is straightforward to implement. Further generalisations such as the *random sweep* Gibbs sampler also exist; see Chapter 5 of Gamerman and Lopes (2006) for full details.

Metropolis-within-Gibbs

Of course, there is no reason why we need restrict ourselves to either implementing a Metropolis-Hastings *or* a Gibbs sampling algorithm. The two sampling methods can be combined and this gives rise to the so called Metropolis-within-Gibbs algorithm. This algorithm is simply the componentwise MH algorithm from Section 1.3.1 where a full conditional distribution is used as the proposal distribution for some unknown quantities with (arbitrary) proposal distributions being used for the remainder. The Metropolis-within-Gibbs algorithm is useful when full conditional distributions are only available in closed form for a subset of the unknown quantities of interest.

1.3.4 Block updates

In practice it is common to update the unknown quantities within an MCMC chain one-at-a-time, that is, by using either the componentwise MH or Gibbs sampling algorithms. Although samplers of this kind are typically easier to implement, using single component updates can give rise to convergence and mixing issues, particularly when some of the unknown quantities have high posterior correlation. Intuitively if two unknown quantities λ_i and λ_j ($i \neq j$) are highly correlated then constructing a proposal for λ_i needs to account for the current value of λ_j . This could be achieved by using a proposal with a

small variance but this will lead to a poorly mixing chain. A practical solution to this problem is to use a *block update* of such highly correlated parameters, that is, update numerous unknown quantities simultaneously. In the example above it may be advantageous to generate the proposed “value” $(\lambda_i^*, \lambda_j^*)$ from a (bivariate) proposal distribution $q(\lambda_i^*, \lambda_j^* | \lambda_i, \lambda_j)$ and either accept or reject the update to both unknown quantities. Of course, this idea generalises straightforwardly to block sizes greater than 2 and further details and discussion can be found in Gamerman and Lopes (2006) and Gelman et al. (2014) amongst others.

1.3.5 Convergence

Recall that the general idea behind MCMC is to construct a Markov chain which has the posterior $\pi(\Lambda | \mathcal{D})$ as its stationary (target) distribution. Therefore, given we are only interested in obtaining posterior realisations, we must ensure that the chain has reached its stationary distribution before using further generated samples. Once a Markov chain has reached its stationary distribution it is said to have *converged*. It is well known that as the number of iterations increases the distribution of the Markov chain tends to the posterior (stationary) distribution, that is, $\Lambda^{(t)} | \mathcal{D} \xrightarrow{d} \Lambda | \mathcal{D}$ as $t \rightarrow \infty$. Obviously we can not perform an infinite number of iterations and so it is useful instead to consider how many iterations are required so that $\Lambda^{(t)} | \mathcal{D} \xrightarrow{d} \Lambda | \mathcal{D}$; this is known as the *burn-in* period. The burn-in period required depends heavily on the form of the posterior distribution and, to a lesser extent, on where the chain is initialised. Clearly this is going to depend on the situation of interest. That said, there are methods for detecting when a Markov chain has not converged. Typically this is done by visual inspection of trace plots showing how the unknown quantities change over the iterations. If the unknown quantities show a clear trend over the iterations then this indicates that the chain has not reached its stationary distribution – in this case the number of iterations (the burn-in period) should be increased. In contrast, if the trace plots show the unknown quantities moving around the support of the distribution in a stable manner then this suggests that the Markov chain has converged. Gelfand and Smith (1990) suggest some additional (informal) checks that can be useful in assessing convergence and some more formal checks have been suggested by Geweke (1992), Raftery and Lewis (1992, 1996) and Gelman (1996) amongst others.

Although these checks are useful it is still possible (and unfortunately fairly easy) to incorrectly assume that a Markov chain has converged, particularly when the stationary (posterior) distribution is multi-modal. It may be the case that, although the unknown quantities show signs of stationarity, they are in fact “trapped” in a local mode and are therefore not exploring the support of the posterior distribution. In an attempt to avoid misdiagnosing convergence it is useful to run multiple Markov chains simultaneously –

each of which should be initialised from a different starting value. If the trace plots from each chain fail to overlap then this is indicative that the chains have not yet converged to the target distribution and a longer burn-in period is required.

1.3.6 Analysing posterior samples

Once the Markov chain has converged to its stationary distribution it follows that any generated samples will be from the posterior $\pi(\Lambda|\mathcal{D})$ by construction. However, as we noted when first introducing MCMC at the beginning of this section, the samples generated from the Markov chain will be dependent and successive draws are said to be *autocorrelated*. If successive values are highly correlated then the amount of information (about the posterior distribution) contained within consecutive samples is much less than if these values were independent. An autocorrelation plot can be useful for assessing the amount of dependence between consecutive samples. The R package *coda* (Plummer et al., 2006) provides a useful function for generating an autocorrelation plot which is simply the autocorrelation function at different lag times (in addition to many other MCMC convergence diagnostics). If the generated samples are highly autocorrelated then it can be useful to *thin* the MCMC output which is done by only considering every i th iterate.

When we have obtained a reasonable number of (almost) un-autocorrelated posterior realisations it is straightforward to compute estimates of posterior summary statistics such as the marginal means and variances of the unknown quantities of interest. Further, we can easily obtain plots of marginal (or joint) posterior distributions by using a kernel density estimate.

1.4 Data augmentation

We conclude this chapter with a brief overview of data augmentation (Tanner and Wong, 1987) and highlight the main advantages of using such an approach. We begin with the standard framework. Suppose the form of our likelihood is non-standard or indeed intractable. Applying Bayes' Theorem will often result in our posterior distribution, $\pi(\Lambda|\mathcal{D})$, taking a non-standard form. This is somewhat inconvenient as we wish to sample from this distribution. Of course, we could appeal to the Metropolis-Hastings algorithm outlined in Section 1.3.1 to obtain posterior realisations. However appealing to data augmentation might allow us to do better.

The general idea behind data augmentation is to introduce some latent variables Z so that the joint posterior density of these variables along with the unknown quantities of interest (Λ) is of a convenient form, that is, $\pi(\Lambda, Z|\mathcal{D})$ is a well-known probability distribution.

The joint posterior distribution of Λ and Z is

$$\pi(\Lambda, Z | \mathcal{D}) \propto \pi(\mathcal{D} | \Lambda, Z) \pi(Z | \Lambda) \pi(\Lambda),$$

whence the posterior density of interest is

$$\pi(\Lambda | \mathcal{D}) \propto \int_Z \pi(\mathcal{D} | \Lambda, Z) \pi(Z | \Lambda) \pi(\Lambda) dZ,$$

that is, the marginal distribution of our augmented posterior. It follows that if we can generate samples from the augmented posterior distribution $\pi(\Lambda, Z | \mathcal{D})$ then we can trivially obtain the posterior distribution over the unknown quantities of interest Λ .

Unfortunately in practice it is often difficult to construct latent variables Z so that the joint posterior density $\pi(\Lambda, Z | \mathcal{D})$ is a well-known probability distribution. However, in some scenarios it can be reasonably straightforward to introduce latent variables which result in the full conditional distributions $\pi(\Lambda | \mathcal{D}, Z)$ and $\pi(Z | \mathcal{D}, \Lambda)$ being of standard form. In fact, it is often the case that the latent variables are introduced by defining their full conditional distribution $\pi(Z | \mathcal{D}, \Lambda)$. If all the full conditional distributions are known then, from the results in Section 1.3.3, it should be clear that we can use a Gibbs sampler to obtain realisations from the joint posterior, that is, repeatedly

- update Λ given \mathcal{D} and Z by sampling from $\pi(\Lambda | \mathcal{D}, Z)$
- update Z given \mathcal{D} and Λ by sampling from $\pi(Z | \mathcal{D}, \Lambda)$.

It follows that a judicious choice of latent variables might allow us to avoid the need to implement a MH algorithm (which needs us to construct and tune proposal distributions) and instead use a more straightforward Gibbs sampling approach if the full conditional distributions are available.

Chapter 2

Analysis of homogeneous ranked data

2.1 Introduction

We consider the popular Plackett–Luce model (Luce, 1959; Plackett, 1975) which is an extension to multiple comparison (ranked) data of the model for paired comparisons proposed by Bradley and Terry (1952). This model relies on strong assumptions about the homogeneity of the ranked data, such as the idea that all rankers share an agreed overall consensus view regarding the preference of the entities. This assumption is perhaps not well justified in many real world scenarios. In this chapter, we begin by assuming that all rankers share similar views and develop more flexible models which allow for heterogeneity between rankers’ beliefs in Chapters 3 and 4. Data typically consist of complete and partial rankings (to be defined in Section 2.2) and we shall detail how the Plackett–Luce model can be modified to allow for a much richer class of rankings such as top- M and top- M partial rankings (defined in Section 2.2.1). Throughout the majority of the literature it is assumed that any particular ranker is no more (or less) likely to share a similar preference (of the entities) to the view expressed by the (assumed) overall consensus group. In Section 2.5 we question this assumption and propose that some individuals may be significantly more informed about the entities they are ranking, and so their opinion should hold more weighting. Ranker reliability is introduced into the model by means of a latent binary indicator within the Plackett–Luce likelihood.

2.2 The Plackett–Luce model

We assume our data (rankings) are observations from the Plackett–Luce model (Luce, 1959; Plackett, 1975). We define the set of all entities to be $\mathcal{K} = \{1, \dots, K\}$ with $K = |\mathcal{K}|$. Each entity has a “skill rating” $\lambda_k > 0$ for $k = 1, \dots, K$. Individual rankings need not contain every entity and so we let $n_i \leq K$ be the number of entities contained within ranking i . Thus, a typical observation from this model is $\mathbf{x}_i = (x_{i1}, \dots, x_{in_i})$, where x_{ij} is the entity that has rank j in ranking i . The probability of such an observation is

$$\begin{aligned}\Pr(\mathbf{X}_i = \mathbf{x}_i | \boldsymbol{\lambda}) &= \prod_{j=1}^{n_i} \frac{\lambda_{x_{ij}}}{\sum_{m=j}^{n_i} \lambda_{x_{im}}} \\ &= \prod_{j=1}^{n_i-1} \frac{\lambda_{x_{ij}}}{\sum_{m=j}^{n_i} \lambda_{x_{im}}}.\end{aligned}\tag{2.1}$$

The Plackett–Luce probability above is said to be a *multistage* model (Marden, 1995) due to the way it is constructed. This multistage construction is naturally highlighted if we consider a simple example. Suppose we have a complete ranking of $K = 4$ entities, namely $\mathbf{x} = (1, 2, 3, 4)$, and skill parameters $\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \lambda_3, \lambda_4)$. The probability of this ranking under the Plackett–Luce model is

$$\Pr\{\mathbf{x} = (1, 2, 3, 4) | \boldsymbol{\lambda}\} = \frac{\lambda_1}{\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4} \times \frac{\lambda_2}{\lambda_2 + \lambda_3 + \lambda_4} \times \frac{\lambda_3}{\lambda_3 + \lambda_4} \times \frac{\lambda_4}{\lambda_4}.$$

From this we observe that the probability of a particular ranking is constructed as the product of the individual (conditional) probabilities that each entity is ranked within their respective position. Let $|\mathbf{c}|_n$ be an operator that takes an arbitrary length vector \mathbf{c} and normalises (but maintains proportionality) the values such that $\sum_i c_i = 1$. It should then be clear that the probability of any entity being allocated rank 1 is given by the corresponding entry within $|\boldsymbol{\lambda}|_n$. The conditional probability for an entity being allocated rank 2 is given by the corresponding entry in $|\{\boldsymbol{\lambda}\} \setminus \{\lambda_{x_1}\}|_n$, that is, the normalised values given that the entity assigned rank 1 is no longer available for selection. This iterative conditioning (on positions which have already been assigned) continues until only a single entity remains, at which point this entity is ranked last. It is often useful to think of this construction in terms of a race containing K horses. Naturally the horse that crosses the line first is awarded rank/position 1, that is, the strongest or most preferred entity within a ranking context. Rank 2 is awarded to the horse which would have won the race if the horse that finished first did not take part. Similarly rank 3 is awarded to the horse which would have won given neither the winner nor the horse awarded rank 2 featured in the race. This process continues until rank K is the only remaining rank to be assigned, at this point the race only contains a single horse which will win by definition.

As the Plackett–Luce probability is constructed using the method outlined above it is said to follow the so-called “*forward ranking process*” (Mollica and Tardella, 2014). Although this ranking process is intuitive, it results in a significant limitation for the Plackett–Luce model. When choosing to model rankings under the Plackett–Luce probability we also inherently make the assumption that each ranking is formed using the forward ranking process. This assumption is one which is often overlooked and is perhaps not well justified within a real world scenario. We might imagine a scenario where a particular ranker is more confident (or equivalently more certain) about the ranks of entities which are their most and least preferred and therefore chooses to allocate these ranks first. Thus in this scenario a particular ranker might allocate entities to rank 1, then to rank K , then to rank $K - 1$ and so on. This is not the forward ranking process. A necessary consequence is that, in this case, the true underlying ranking distribution will not follow the Plackett–Luce model. Indeed, as we shall see in Chapter 6, the choice of the (forward ranking) Plackett–Luce model in such a scenario typically results in a poor approximation to the true underlying ranking distribution and, as a result, potentially misleading inferences.

If the forward ranking process assumption is not plausible we could instead choose to model rankings using the Reverse Plackett–Luce model. This model was suggested by Marden (1995) and uses a similar multistage construction as the standard Plackett–Luce model but on the reverse rankings. The probability of a particular ranking under this model is

$$\Pr(\mathbf{X}_i = \mathbf{x}_i | \boldsymbol{\lambda}) = \prod_{j=1}^K \frac{\lambda_{x_{iK-j+1}}}{\sum_{m=1}^{K-j+1} \lambda_{x_{im}}}.$$

Therefore, for the complete ranking $\mathbf{x} = (1, 2, 3, 4)$, the probability is

$$\Pr\{\mathbf{x} = (1, 2, 3, 4) | \boldsymbol{\lambda}\} = \frac{\lambda_4}{\lambda_4 + \lambda_3 + \lambda_2 + \lambda_1} \times \frac{\lambda_3}{\lambda_3 + \lambda_2 + \lambda_1} \times \frac{\lambda_2}{\lambda_2 + \lambda_1} \times \frac{\lambda_1}{\lambda_1}.$$

Perhaps not too surprisingly this model is said to follow the so-called “*backward ranking process*” (Mollica and Tardella, 2014), that is, each ranking is formed by first allocating the entity which is least preferred (rank K) followed by that which is second least preferred (rank $K - 1$) and so on. In terms of our horse race scenario, the horse which finishes first is that which receives rank K , then the horse that would have won conditional on the “winning” horse not being in the race receives rank $K - 1$, the final remaining horse is the most preferred and receives rank 1. Graves et al. (2003) used this model in the context of NASCAR races and more recently Henderson and Kirrane (2018) used the Reverse Plackett–Luce model when analysing Formula 1 races. Both these analyses found the assumption of the backward ranking process resulted in better model fit than assuming the forward ranking process. It is perhaps difficult to justify *a priori* which assumption is more plausible. Indeed, a ranker could choose to allocate their ranks in any of the

$K!$ orderings included within \mathcal{S}_K , the set of all $K!$ permutations of K elements. Mollica and Tardella (2014) explored this idea and developed the Extended Plackett–Luce model. This model relaxes assumptions on any explicit ranking process and the “choice ordering” is instead inferred from the rankings themselves. In what follows we assume the forward ranking process and so use the standard form of the Plackett–Luce model. We shall revisit the Extended Plackett–Luce model in Chapter 6.

A further limitation of the Plackett–Luce model is that it only defines a probability for certain types of ranking. The model requires each ranker to report a position for each of the entities they consider. This allows for two types of ranking: (a) *Complete rankings*, which occur when a ranker considers, and assigns a rank to, all possible entities and (b) *Partial rankings*, which occur when a ranker considers a subset of all the entities but still reports a rank for each entity considered, and so $n_i < K$ in this scenario. A paired comparison, that is, an ordered list of two entities is equivalent to a partial ranking with $n_i = 2$. This should come as no surprise given that the Plackett–Luce model is a generalisation (to multiple comparison data) of the Bradley–Terry model which is defined for paired comparison data (Bradley and Terry, 1952). We will also consider a special case known commonly as *top- M rankings*. Here individuals report a rank only for those entities which they classify as being positioned 1 to M (where they have considered more than M entities). The Plackett–Luce model does not adequately capture the information within data of this type. For example, if we naively chose to model *top- M* rankings (with $n_i = M$) using the Plackett–Luce model in (2.1) then the entity that is assigned rank M would be treated as if it was ranked last. Furthermore the model would also behave as if all the entities that did not appear within the ranking were not considered by the ranker, that is, the model would treat this as a partial ranking. However, in the case of a *top- M* ranking, we have the additional information that, although they do not receive a particular rank, the entities not featuring in the ranking are considered to have at least rank $M + 1$. Note that the definition of a partial ranking varies within the literature with some authors defining a partial ranking to be what we consider a top- M ranking. We therefore make it clear that, within this thesis, the terms partial and top rankings are used to refer to ranking types as defined above.

At first it appears that allowing for *top- M* rankings may be problematic as we wish to marginalise over all possible (unknown) positions of the unranked entities. However, the Plackett–Luce model (more specifically the distribution it induces over rankings) is internally consistent, that is, the probability of a particular ranking is independent of the subset of entities from which the ranking was formed; see Hunter (2004) for a proof outline. It follows that, under the Plackett–Luce model, it is trivial to consistently combine incomplete (top, partial) rankings. The following section details how the Plackett–Luce probability can be extended to use all the information contained within *top- M* rankings.

2.2.1 Top-M rankings

In real world scenarios we often encounter a much broader class of rankings which can be classified as *top- M* and *top- M partial rankings*. A *top- M* ranking is obtained when a particular individual considers all K entities but only assigns entities to ranks 1 to M (their most preferred M entities) and leaves the remaining entities “unranked”. A *top- M partial* ranking is a special case of a *top- M* ranking which is obtained when an individual only considers a subset of all the entities (hence $n_i < K$) and again proceeds to only assign entities to ranks 1 to M (leaving the remaining ones they considered unranked).

A modification to the Plackett–Luce probability is required to allow for these additional ranking types. Caron et al. (2014) detail a modification to allow for *top- M* rankings and their result can be trivially extended to allow for *top- M* partial rankings. Recall the set of all entities is defined by $\mathcal{K} = \{1, \dots, K\}$. Now suppose ranker i considers $K_i \leq K$ entities, and denote the set of these entities as $\mathcal{K}_i \subseteq \mathcal{K}$. Also let $\mathcal{U}_i = \mathcal{K}_i \setminus \{\mathbf{x}_i\}$ be the collection of “unranked” entities (for ranker i), by which we mean the entities they considered but did not feature in their ranking. From the definition of \mathcal{U}_i it is clear that any entity $k \in \mathcal{U}_i$ is considered to be ranked at least $(n_i + 1)$ th. The probability of a particular ranking under the (now modified) Plackett–Luce model is

$$\Pr(\mathbf{X}_i = \mathbf{x}_i | \boldsymbol{\lambda}) = \prod_{j=1}^{n_i} \frac{\lambda_{x_{ij}}}{\sum_{m=j}^{n_i} \lambda_{x_{im}} + \sum_{m \in \mathcal{U}_i} \lambda_m}. \quad (2.2)$$

Note that

$$\Pr(\mathbf{X}_i = \mathbf{x}_i | \boldsymbol{\lambda}) \neq \prod_{j=1}^{n_i-1} \frac{\lambda_{x_{ij}}}{\sum_{m=j}^{n_i} \lambda_{x_{im}} + \sum_{m \in \mathcal{U}_i} \lambda_m}$$

unless $\mathcal{U}_i = \emptyset$, the empty/null set. Therefore in a situation where we have only (a) *Complete* or (b) *Partial* rankings (hence $\mathcal{U}_i = \emptyset \forall i$) the modified Plackett–Luce probability simplifies and we recover (2.1). Henceforth we shall maintain full generality and proceed assuming that at least one ranker provides a top or top partial ranking, that is, there exists an i such that $\mathcal{U}_i \neq \emptyset$.

We also note that, unlike the standard Plackett–Luce model, simply analysing the reverse rankings is no longer equivalent to analysing the standard (forward) rankings under the Reverse Plackett–Luce model. Indeed analysing the reverse rankings under model 2.2 would be particularly challenging given that we do not have explicit ranks for those entities judged to be ranked in at least position $M + 1$. This results in those entities effectively being tied for “first” place (under the reverse rankings). Of course, data of this form could be analysed by implementing methods for handling ties within ranked data – these

methods are discussed in Section 2.2.4.

2.2.2 Identifiability issues

The Plackett–Luce model suffers from a fundamental problem of parameter identifiability: the probability (2.2) is invariant to strictly positive scalar multiplication of the skill parameters. More formally, if we let $\lambda_k^* = C\lambda_k$ for $k = 1, \dots, K$ with $C > 0$ then (due to the normalisation within the construction of the probability) we have $\Pr(\mathbf{X} = \mathbf{x}|\boldsymbol{\lambda}) = \Pr(\mathbf{X} = \mathbf{x}|\boldsymbol{\lambda}^*)$ for any ranking \mathbf{x} . Numerous approaches can be taken to overcome this issue. For example we could choose to constrain $\boldsymbol{\lambda}$ so that it lies on the $K - 1$ dimensional simplex. Alternatively, we could take an approach similar to that of a corner constraint and fix $\lambda_1 \equiv 1$. Both these methods reduce the number of free parameters to $K - 1$. Caron and Doucet (2012) noted that the identifiability issue can also result in poor mixing within MCMC chains. Resolving the mixing (and identifiability) issue is of course desirable and it turns out that, within the Bayesian solution we consider, this can be easily achieved through a suitable rescaling strategy within the inference scheme. We now turn to dealing with the rescaling issue.

2.2.3 Rescaling

Let us consider $\Lambda^\dagger = \sum_{k=1}^K \lambda_k$, the sum of all the K skill parameters. As discussed in Section 2.2.2, the Plackett–Luce likelihood is invariant to scalar multiplication of the parameters, and hence Λ^\dagger is not likelihood identifiable. Indeed, if we let $\lambda_k^* = \lambda_k/\Lambda^\dagger$ for $k = 1, \dots, K$, we have that $\pi(\boldsymbol{\lambda}^*, \Lambda^\dagger | \mathcal{D}) = \pi(\boldsymbol{\lambda}^* | \mathcal{D}) \pi(\Lambda^\dagger)$.

Caron and Doucet (2012) noted that without the addition of a rescaling step, MCMC schemes for Plackett–Luce models can suffer from poor mixing. The idea is to rescale the parameters so that the posterior distribution of Λ^\dagger is the same as its prior distribution. This is achieved by performing an appropriate rescaling step at each iteration of the MCMC scheme. Of course, the rescaling required will be situation dependent as the prior distribution on Λ^\dagger is induced by the prior choice for each of the skill parameters, λ_k .

2.2.4 Ties

Ties within ranked data can present particular modelling challenges. Most models for ranked data, including the Plackett–Luce model, are built upon the assumption that only a single entity can be assigned to a particular rank, that is, multiple entities can not be “tied” for the same position. It is however possible to overcome this issue and indeed numerous methods for incorporating ties within an analysis under the Plackett–Luce model

have been discussed within the literature. Perhaps the most intuitive method is to evaluate the exact contribution the tied entities make to the likelihood, that is, average over the PL probabilities of all possible rankings formed by permuting the ranks of the tied entities. Unfortunately this approach can significantly increase the amount computation required even in scenarios where relatively few entities are tied. For example, when six entities are tied for a particular position the likelihood of such a ranking contains an additional $6! = 720$ terms. Furthermore if we envisage a scenario in which numerous entities are tied for numerous different ranks then it is clear that the computation of the likelihood will soon become infeasible. An alternative approach discussed by Breslow et al. (1974) uses an approximation to the exact likelihood by assuming that each of the tied entities (within a given rank) is preferred to all other entities ranked either in the same position or lower. This approach significantly reduces the computational burden (compared to calculating the likelihood exactly). However it does come at the cost of a non-exact likelihood function. Baker and McHale (2015) discuss a further (more exact) likelihood approximation where they consider the likelihood of all possible rankings (formed by permuting the tied entities). However the skill parameters for the tied entities are defined to be the mean of the respective tied entity skill parameters, with some additional random noise. For example, if there are t tied entities then the likelihood under the $t!$ possible rankings would be evaluated subject to $\lambda_i = \hat{\mu} + \epsilon_i$ ($i = 1, \dots, t$) where $\hat{\mu} = \sum_{i=1}^t \lambda_i / t$. Full details of this schematic and details of inference on λ_i are discussed in Appendix A of Baker and McHale (2015).

Our solution to the problem of ties is based on our MCMC solution to the inference problem. Essentially, at each MCMC iteration, we simulate a ranking without ties from a uniform distribution over all rankings consistent with the rankings with ties. For example, suppose we have a ranking $\mathbf{x} = (1, \bar{2}, \bar{3}, 4, 5)$ where entities 2 and 3 are tied for second position (indicated by the bar). To incorporate this ranking into our analysis we would let $\mathbf{x} = (1, 2, 3, 4, 5)$ with probability 0.5 and otherwise let $\mathbf{x} = (1, 3, 2, 4, 5)$ at each iteration of our MCMC scheme. This schematic can be trivially extended to incorporate rankings with ties involving more than two entities, for example rankings such as $\mathbf{x} = (\bar{1}, \bar{2}, \bar{3}, 4, 5)$. Furthermore we can allow for the possibility of more than a single set of tied entities within a single ranking, that is, a ranking of the form $\mathbf{x} = (\bar{1}, \bar{2}, \bar{3}, \bar{4}, \bar{5})$. In all scenarios we sample, uniformly at random, from the discrete distribution over all possible rankings formed by permutations of the tied entities. This method for incorporating ties within a ranked data analysis is described further by Glickman and Hennessy (2015).

2.2.5 Simulating data from the modified Plackett–Luce model

Having defined our modified Plackett–Luce probability (2.2) we are now in a position to describe the process which allows us to simulate data under this model. We begin by specifying “true” values for the skill parameters, namely $\lambda_k > 0$ for $k = 1, \dots, K$. These values could be simulated from an appropriate distribution if desired. In Section 2.2.2 we discussed how the Plackett–Luce probability is invariant to (strictly positive) scalar multiplication of the skill parameters; we must therefore be cautious when specifying values for our skill parameters. It is important to remind ourselves that the skill parameters for each entity can only be compared relative to one another. Hence the choice of $\boldsymbol{\lambda} = (3, 2, 5)$ specifies the equivalent distribution over rankings as the choice of $\boldsymbol{\lambda} = (0.3, 0.2, 0.5)$ and indeed the same distribution as $C\boldsymbol{\lambda}$ for any $C > 0$.

We describe the data generating process using the well-known exponential latent variable representation of the Plackett–Luce model (Diaconis (1988), Marden (1995)). In this representation we introduce latent variables ν_j , which are interpreted as the (latent) arrival time of entity j (in a homogeneous Poisson process). These variables follow independent exponential distributions with rate parameter λ_j . The latent arrival times are then trivially converted into rankings by assigning rank 1 to the entity that arrived first, rank 2 to the entity that arrived second and so on. Formally, the complete ranking \mathbf{x} is generated via the following process.

1. Sample $\nu_j \stackrel{indep}{\sim} \text{Exp}(\lambda_j)$ for $j = 1, \dots, K$.
2. Set $x_j = \underset{q \in S_j}{\operatorname{argmin}} \nu_q$ where $S_j = \mathcal{K} \setminus \{x_1, \dots, x_{j-1}\}$ for $j = 1, \dots, K$.

We note that the process described above is only designed to generate complete rankings. Once we have obtained complete rankings it is possible to convert these to partial, top- M or top- M partial rankings as required. Partial rankings are formed by removing (from the complete ranking) those entities which were not considered by the ranker and preserving the preference order of those entities which remain. Top- M rankings are trivially obtained by only considering the first M positions of the complete ranking. Top- M partial rankings are obtained from the complete ranking using a two step process. We begin by first obtaining the corresponding partial ranking (as above), and then the ranking required is given by taking the first M positions within the (newly formed) partial ranking. For example, suppose we generate the complete ranking $\mathbf{x} = (1, 7, 3, 5, 2, 6, 8, 4)$ using the process outlined above. Table 2.1 shows the corresponding top-5, partial, and top-5 partial rankings formed from this complete ranking. The partial rankings were formed assuming that entities 2 and 8 were not considered. The table also provides the number of entities within each ranking, n_i , and the total number of entities each ranker considered, K_i .

Ranking type	Rank								n_i	K_i	\mathcal{U}_i	$\mathcal{K} \setminus \mathcal{K}_i$
	1	2	3	4	5	6	7	8				
Complete	1	7	3	5	2	6	8	4	8	8	\emptyset	\emptyset
Top-5	1	7	3	5	2				5	8	{4, 6, 8}	\emptyset
Partial	1	7	3	5	6	4			6	6	\emptyset	{2, 8}
Top-5 partial	1	7	3	5	6				5	6	{4}	{2, 8}

Table 2.1: Ranking types

The sets \mathcal{U}_i and $\mathcal{K} \setminus \mathcal{K}_i$ containing the unranked entities and those entities that were not considered by ranker i respectively are also provided.

2.3 Bayesian inference

We now describe how Bayesian inference can be performed using rankings assumed to follow the modified Plackett–Luce model (2.2). In order to make meaningful inferences these data must contain numerous rankings (n). To formulate our likelihood concisely we let $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^n$ be the collection of all such rankings. The likelihood under the Plackett–Luce model is then given as the product of the respective probabilities of each ranking, and hence takes the form

$$\begin{aligned}\pi(\mathcal{D}|\boldsymbol{\lambda}) &= \prod_{i=1}^n \Pr(\mathbf{X}_i = \mathbf{x}_i | \boldsymbol{\lambda}) \\ &= \prod_{i=1}^n \prod_{j=1}^{n_i} \frac{\lambda_{x_{ij}}}{\sum_{m=j}^{n_i} \lambda_{x_{im}} + \sum_{m \in \mathcal{U}_i} \lambda_m}.\end{aligned}\quad (2.3)$$

Inference could proceed using a maximum likelihood approach such as the MM algorithm (Hunter, 2004), to maximise this likelihood and obtain an estimate for our skill parameters $\hat{\boldsymbol{\lambda}}$. Here however we adopt the Bayesian approach to inference and therefore define a suitable prior distribution along with an appropriate posterior sampling scheme.

2.3.1 Prior specification and latent variables

The choice of suitable prior distributions is a problem well discussed within the Bayesian literature (Bernardo and Smith, 1994). Here our choice of prior distribution is mainly motivated by mathematical convenience. However, we believe our choice is sufficiently flexible to allow for informative prior beliefs to be portrayed if desired. The skill parameters λ_k are required to be strictly positive and so it seems sensible to choose independent Gamma prior distributions, namely $\lambda_k \stackrel{\text{indep}}{\sim} \text{Ga}(a_k, b_k)$ for $k = 1, \dots, K$. It has been shown that the

rate parameters b_k are not likelihood identifiable (Caron and Doucet, 2012) and so we let $b_k = b = 1$ as our skill parameters are invariant to (strictly positive) scalar multiplication. This is done widely within the literature. Therefore our prior distribution for the skill parameters is $\lambda_k \stackrel{\text{indep}}{\sim} \text{Ga}(a_k, 1)$ for $k = 1, \dots, K$ and our complete model specification is

$$\begin{aligned} \mathbf{X}_i | \boldsymbol{\lambda} &\stackrel{\text{indep}}{\sim} \text{PL}(\boldsymbol{\lambda}) & i = 1, \dots, n, \\ \lambda_k &\stackrel{\text{indep}}{\sim} \text{Ga}(a_k, 1) & k = 1, \dots, K, \end{aligned}$$

where $\mathbf{X} | \boldsymbol{\lambda} \sim \text{PL}(\boldsymbol{\lambda})$ denotes that ranking $\mathbf{X} = \mathbf{x}$ follows the Plackett–Luce model with probability defined in (2.2).

Under this prior choice the form of the posterior distribution is highly non-standard and so we adopt the sampling-based approach of Markov chain Monte Carlo (MCMC). It can be desirable to implement a Gibbs sampler (if possible) rather than a Metropolis-Hastings sampler, particularly if this benefits in increased sampling efficiency. Conditional on an independent Gamma prior specification, Caron and Doucet (2012) showed that by appealing to data augmentation techniques it is possible to facilitate a conjugate update for the skill parameters. Our sample space is augmented by introducing appropriate latent variables (collectively denoted Z) which are interpreted as the hypothetical (exponential) inter-event times of the entities in the homogeneous Poisson processes referred to in Section 2.2.5. These latent variables are defined through their full conditional distribution and are given by

$$z_{ij} | \mathcal{D}, \boldsymbol{\lambda} \stackrel{\text{indep}}{\sim} \text{Exp} \left(\sum_{m=j}^{n_i} \lambda_{x_{im}} + \sum_{m \in \mathcal{U}_i} \lambda_m \right), \quad (2.4)$$

for $i = 1, \dots, n$ and $j = 1, \dots, n_i$.

Aside

If we first let Λ denote the collection of all skill parameters then using the results from Section 1.4 we can verify that we do indeed obtain the desired posterior of our skill parameters when we integrate out these latent variables from the joint posterior as follows

$$\begin{aligned} \pi(\Lambda | \mathcal{D}) &= \int_Z \pi(\Lambda, Z | \mathcal{D}) dZ \\ &\propto \int_Z \pi(Z | \Lambda, \mathcal{D}) \pi(\Lambda | \mathcal{D}) \pi(\Lambda) dZ \end{aligned}$$

$$\begin{aligned}
 &= \pi(\Lambda) \underbrace{\int_0^\infty \int_0^\infty \cdots \int_0^\infty}_{\sum_{i=1}^n n_i} \left[\prod_{i=1}^n \prod_{j=1}^{n_i} \frac{\lambda_{x_{ij}}}{\sum_{m=j}^{n_i} \lambda_{x_{im}} + \sum_{m \in \mathcal{U}_i} \lambda_m} \left(\sum_{m=j}^{n_i} \lambda_{x_{im}} + \sum_{m \in \mathcal{U}_i} \lambda_m \right) \right. \\
 &\quad \times \exp \left\{ - \left(\sum_{m=j}^{n_i} \lambda_{x_{im}} + \sum_{m \in \mathcal{U}_i} \lambda_m \right) z_{ij} \right\} \left. dZ \right] \\
 &= \pi(\Lambda) \prod_{i=1}^n \prod_{j=1}^{n_i} \lambda_{x_{ij}} \times \prod_{i=1}^n \prod_{j=1}^{n_i} \int_0^\infty \exp \left\{ - \left(\sum_{m=j}^{n_i} \lambda_{x_{im}} + \sum_{m \in \mathcal{U}_i} \lambda_m \right) z_{ij} \right\} dz_{ij} \\
 &= \pi(\Lambda) \prod_{i=1}^n \prod_{j=1}^{n_i} \lambda_{x_{ij}} \times \left[\prod_{i=1}^n \prod_{j=1}^{n_i} - \left(\sum_{m=j}^{n_i} \lambda_{x_{im}} + \sum_{m \in \mathcal{U}_i} \lambda_m \right)^{-1} \right. \\
 &\quad \times \exp \left\{ - \left(\sum_{m=j}^{n_i} \lambda_{x_{im}} + \sum_{m \in \mathcal{U}_i} \lambda_m \right) z_{ij} \right\} \left. \right]_0^\infty \\
 &= \pi(\Lambda) \prod_{i=1}^n \prod_{j=1}^{n_i} \frac{\lambda_{x_{ij}}}{\sum_{m=j}^{n_i} \lambda_{x_{im}} + \sum_{m \in \mathcal{U}_i} \lambda_m} \left(\prod_{i=1}^n \prod_{j=1}^{n_i} e^0 - e^{-\infty} \right) \\
 &= \pi(\Lambda) \pi(\mathcal{D}|\Lambda).
 \end{aligned}$$

As we shall see in the section that follows, we are also able to obtain the full conditional distribution for Λ in closed form under this latent variable specification.

2.3.2 Full conditional distributions

Our posterior distribution is formed by applying Bayes' Theorem. As we have augmented our sample space, the resulting posterior distribution is a joint distribution containing the latent random variables Z and the skill parameters $\boldsymbol{\lambda}$. Before starting our derivation it is useful to first construct the density of all stochastic quantities in the model; this is given by

$$\begin{aligned}
 \pi(\boldsymbol{\lambda}, \mathcal{D}, Z) &= \pi(Z|\mathcal{D}, \boldsymbol{\lambda}) \pi(\mathcal{D}|\boldsymbol{\lambda}) \pi(\boldsymbol{\lambda}) \\
 &= \prod_{i=1}^n \prod_{j=1}^{n_i} \left(\sum_{m=j}^{n_i} \lambda_{x_{im}} + \sum_{m \in \mathcal{U}_i} \lambda_m \right) \exp \left\{ -z_{ij} \left(\sum_{m=j}^{n_i} \lambda_{x_{im}} + \sum_{m \in \mathcal{U}_i} \lambda_m \right) \right\} \\
 &\quad \times \prod_{i=1}^n \prod_{j=1}^{n_i} \frac{\lambda_{x_{ij}}}{\sum_{m=j}^{n_i} \lambda_{x_{im}} + \sum_{m \in \mathcal{U}_i} \lambda_m} \times \prod_{k=1}^K \frac{\lambda_k^{a_k-1} e^{-\lambda_k}}{\Gamma(a_k)} \\
 &= \prod_{k=1}^K \frac{\lambda_k^{a_k-1} e^{-\lambda_k}}{\Gamma(a_k)} \times \prod_{i=1}^n \prod_{j=1}^{n_i} \lambda_{x_{ij}} \exp \left\{ -z_{ij} \left(\sum_{m=j}^{n_i} \lambda_{x_{im}} + \sum_{m \in \mathcal{U}_i} \lambda_m \right) \right\}. \quad (2.5)
 \end{aligned}$$

We are now able to obtain the full conditional distributions (FCDs) by constructing the conditional distribution of each unknown quantity given all other stochastic quantities and the data. If we begin with the latent variables Z it should be clear that

$$\pi(Z|\mathcal{D}, \boldsymbol{\lambda}) \propto \prod_{i=1}^n \prod_{j=1}^{n_i} \exp \left\{ -z_{ij} \left(\sum_{m=j}^{n_i} \lambda_{x_{im}} + \sum_{m \in \mathcal{U}_i} \lambda_m \right) \right\},$$

and so the full conditional distribution of the z_{ij} is as in (2.4) (by construction). The full conditional distribution for the remaining random quantities, $\boldsymbol{\lambda}$, is

$$\begin{aligned} \pi(\boldsymbol{\lambda}|\mathcal{D}, Z) &\propto \prod_{k=1}^K \lambda_k^{a_k-1} e^{-\lambda_k} \prod_{i=1}^n \prod_{j=1}^{n_i} \lambda_{x_{ij}} \exp \left\{ -z_{ij} \left(\sum_{m=j}^{n_i} \lambda_{x_{im}} + \sum_{m \in \mathcal{U}_i} \lambda_m \right) \right\} \\ &= \prod_{k=1}^K \lambda_k^{a_k+q_k-1} \exp \left\{ - \left(1 + \sum_{i=1}^n \sum_{j=1}^{n_i} \delta_{ij}(k) z_{ij} \right) \lambda_k \right\} \end{aligned}$$

where

$$q_k = \sum_{i=1}^n \mathbb{I}(k \in \{\mathbf{x}_i\}),$$

and

$$\delta_{ij}(k) = \mathbb{I}(k \in \{x_{ij}, \dots, x_{in_i}\} \cup \mathcal{U}_i), \quad (2.6)$$

are the number of times entity k appears within a ranking and an indicator variable over the event that entity k receives a rank no better than j in ranking i , respectively. It then follows that

$$\lambda_k | \mathcal{D}, Z \stackrel{\text{indep}}{\sim} \text{Ga} \left(a_k + q_k, 1 + \sum_{i=1}^n \sum_{j=1}^{n_i} \delta_{ij}(k) z_{ij} \right),$$

for $k = 1, \dots, K$.

As we now have a complete set of full conditional distributions we are in a position to construct a sampling scheme to generate realisations from our posterior distribution. We note that this is a straightforward modification of the Gibbs sampler of Caron and Doucet (2012) where here the definition of $\delta_{ij}(k)$ in (2.6) has changed so that we can deal with top- M rankings. This sampler is also somewhat similar to that detailed in Caron et al. (2014) but here we consider a (fixed) finite number of entities $K < \infty$.

2.3.3 MCMC - Gibbs sampling via latent variables

In Section 2.3.2 we derived a complete set of full conditional distributions assuming the prior and latent variable specification as in Section 2.3.1. We can now construct a Markov chain Monte Carlo scheme to generate realisations from our posterior distribution: this is a Gibbs sampler. The algorithm outline is as follows.

1. Initialise the iteration counter to $t = 1$.

Initialise the state of the chain, one option is as follows

- For $k = 1, \dots, K$, sample $\lambda_k^{(0)} \stackrel{\text{indep}}{\sim} \text{Ga}(a_k, 1)$.
- For $i = 1, \dots, n$, $j = 1, \dots, n_i$, sample $z_{ij}^{(0)} \stackrel{\text{indep}}{\sim} \text{Exp}\left(\sum_{m=j}^{n_i} \lambda_{x_{im}}^{(0)} + \sum_{m \in \mathcal{U}_i} \lambda_m^{(0)}\right)$.

2. Obtain new realisations of $\boldsymbol{\lambda}^{(t)}$, $Z^{(t)}$ from $\boldsymbol{\lambda}^{(t-1)}$, $Z^{(t-1)}$ as follows:

- For $k = 1, \dots, K$, sample

$$\lambda_k^{(t)} | \mathcal{D}, Z^{(t-1)} \stackrel{\text{indep}}{\sim} \text{Ga}\left(a_k + q_k, 1 + \sum_{i=1}^n \sum_{j=1}^{n_i} \delta_{ij}(k) z_{ij}^{(t-1)}\right).$$

- For $i = 1, \dots, n$, $j = 1, \dots, n_i$, sample

$$z_{ij}^{(t)} | \mathcal{D}, \boldsymbol{\lambda}^{(t)} \stackrel{\text{indep}}{\sim} \text{Exp}\left(\sum_{m=j}^{n_i} \lambda_{x_{im}}^{(t)} + \sum_{m \in \mathcal{U}_i} \lambda_m^{(t)}\right).$$

3. Rescale:

- Sample $\Lambda^\dagger \sim \text{Ga}\left(\sum_{k=1}^K a_k, 1\right)$.
- Calculate $\Sigma = \sum_{k=1}^K \lambda_k^{(t)}$.
- For $k = 1, \dots, K$, let $\lambda_k^{(t)} \rightarrow \lambda_k^{(t)} \Lambda^\dagger / \Sigma$.

4. Set $t = t + 1$ and return to step 2.

With computational efficiency in mind, we note that q_k and $\delta_{ij}(k)$ depend only on the data and so remain constant throughout the Markov chain Monte Carlo scheme. These values can therefore be computed at step 1 and reused within each iteration. We also note that $q_k = n$ for all k if our data consists entirely of complete rankings.

The rescaling in Step 3 follows from the discussion in Section 2.2.3 where it was noted that, without the addition of a rescaling step, MCMC schemes for Plackett–Luce models can suffer from poor mixing (Caron and Doucet, 2012). The idea was to rescale the parameters so that the posterior distribution of the sum of all the K skill parameters is the same as its prior distribution (as the data are not informative about this sum). Let $\Lambda^\dagger = \sum_{k=1}^K \lambda_k$ be the sum of all the K skill parameters. Then as $\lambda_k \stackrel{\text{indep}}{\sim} \text{Ga}(a_k, 1)$ for $k = 1, \dots, K$ *a priori* it follows that the (induced) prior for Λ^\dagger is a $\text{Ga}(\sum_{k=1}^K a_k, 1)$ distribution. The posterior for Λ^\dagger can therefore be kept the same as the prior by drawing a realisation of Λ^\dagger from a $\text{Ga}(\sum_{k=1}^K a_k, 1)$ distribution and then multiplying the current (posterior) λ values by a factor of Λ^\dagger / Σ , where $\Sigma = \sum_{k=1}^K \lambda_k^{(t)}$ denotes the current (posterior) sum of the K skill parameters.

2.4 Simulation study

In this study we perform Bayesian inference on data which are simulated (generated) under the Plackett–Luce model. The benefit of performing inference on data simulated from the true model is that we know the parameter values from which these data were generated. We can therefore assess how our model performs under these conditions before performing inference on a real world scenario. Here we consider two datasets, both of which contain (only) complete rankings of $K = 20$ entities. The set of all entities is therefore given by $\mathcal{K} = \{1, \dots, 20\}$. Our first dataset (Dataset 1) contains $n = 40$ rankings which were simulated using the process outlined in Section 2.2.5 subject to the “true” parameter values

$$\lambda_1 = 20, \quad \lambda_k = \lambda_{k-1} - 1, \quad \text{for } k = 2, \dots, K,$$

that is, $\boldsymbol{\lambda} = (20, 19, \dots, 1)$. Under this parameter specification entities that are indexed by smaller numbers are more preferred; these entities are therefore more likely to feature towards the beginning of a ranking (be assigned a low numbered rank) in comparison to those entities indexed by large numbers. This becomes clear if we consider the notion of an *optimal* ranking. The optimal ranking, denoted $\hat{\mathbf{x}}$, is defined as the ranking such that the Plackett–Luce probability is maximised (conditional on some *fixed* skill parameters). Mathematically such a ranking is given by

$$\hat{\mathbf{x}}|\boldsymbol{\lambda} = \underset{\mathbf{x} \in \mathcal{S}_K}{\operatorname{argmax}} \Pr(\mathbf{X} = \mathbf{x}|\boldsymbol{\lambda}), \tag{2.7}$$

whence it should be clear that the optimal ranking is $\hat{\mathbf{x}} = (1, 2, \dots, 20)$ under our current choice of skill parameters.

The second dataset (Dataset 2) is comprised of $n = 50$ rankings, the first 40 of which are those rankings from within Dataset 1 and the additional 10 rankings (numbered 41–50) are random permutations of the K entities. Dataset 2 is therefore an extension of Dataset 1 and we make it clear that the rankings which are common amongst these datasets maintain the same labels. The random permutations shall be referred to as *uninformative* or *spam* rankings. The uninformative rankings are generated under the usual data generating process (as in Section 2.2.5) with $\lambda_k = c$ for $k = 1, \dots, K$, where c is an arbitrary positive constant. We note that this method of simulation is equivalent to sampling an element uniformly at random from the set of all permutations \mathcal{S}_K . The purpose of analysing a dataset such as this is to investigate how our posterior distribution (and therefore our inference) is affected by these spam rankings. In some sense this is a sensitivity analysis to determine how robust the Plackett–Luce model is to the addition of spurious rankings. The rankings used within this study can be found within the appendices; Table B.1 contains Dataset 1 and the additional 10 rankings which are included within Dataset 2 are provided in Table B.2.

Before we can perform Bayesian inference on these data we must first specify suitable prior distributions. In order to maintain conjugacy (and hence use the Gibbs sampler outlined in Section 2.3.3) we choose independent gamma prior distributions for each of our skill parameters as discussed in Section 2.3.1. In this scenario we also know the true parameter values from which these data were simulated, however, we choose to perform inference assuming we have no prior knowledge regarding the strength of each entity. We therefore desire a prior specification such that each ranking is equally likely *a priori*. This is achieved by choosing $\lambda_k \stackrel{\text{iid}}{\sim} \text{Ga}(a, 1)$, that is, setting $a_k = a$ for $k = 1, \dots, K$. Without loss of generality we take $a = 1$.

2.4.1 Posterior analysis

Before we begin our investigation into the posterior distribution we shall first give some computational details. Our MCMC algorithm was initialised using a random draw from the prior distribution. We then proceeded to perform 11K iterations; the first 1K of which were discarded as a burn-in period. This left us with 10K (almost) un-autocorrelated samples from our posterior distribution. The computational time required to perform inference on these data is approximately 1 and 1.3 seconds for Datasets 1 and 2 respectively. This inference scheme is implemented in C and computation is performed on a single thread of an Intel Core i7-4790S CPU (3.20GHz clock speed).

Our posterior distribution is of high dimension, namely $(n \times K) + K$ in these analyses. Assessing convergence and mixing of each individual parameter is therefore problematic; especially as it is easy to see that the dimension of our parameter space will increase

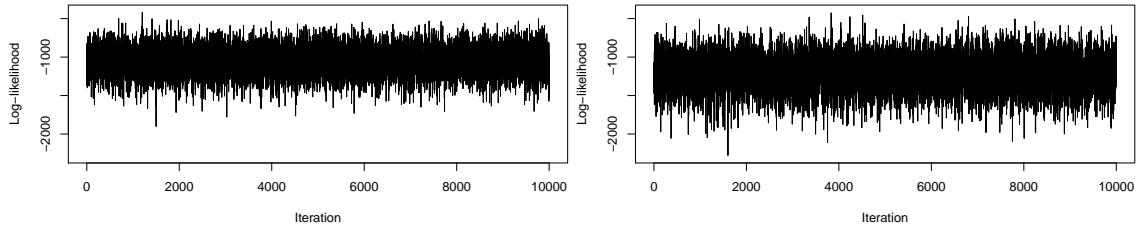


Figure 2.1: Trace plots of the log complete data likelihood for Datasets 1 and 2 from left to right respectively.

significantly for larger datasets. Consequently it is desirable to obtain a method for conveniently assessing the convergence and mixing of a Markov chain for high dimensional sample spaces such as this. As opposed to considering each random variable in turn we instead propose to consider an overall summary of our random variables, namely the complete data likelihood, $\pi(Z, \mathcal{D}|\boldsymbol{\lambda}) = \pi(Z|\mathcal{D}, \boldsymbol{\lambda})\pi(\mathcal{D}|\boldsymbol{\lambda})$. Gelman et al. (2014) advocate this approach to assessing convergence (especially when implementing mixture models which we consider in Chapter 3). Figure 2.1 depicts trace plots of the log complete data likelihood (after burn-in) for the analyses of both Datasets 1 and 2. We observe that our chains appear to be mixing well and furthermore each chain appears to be sampling from its stationary distribution. Convergence (to the stationary distribution) was also verified by initialising numerous chains at different starting values and checking the posterior distributions are equivalent (up to stochastic noise) in all cases.

Given that we are satisfied that our MCMC scheme is generating realisations from the posterior distribution (for both analyses) we can now begin our investigation into the inferences on our skill parameters $\boldsymbol{\lambda}$. In order to ease comparison we perform (offline) rescaling by letting $\lambda_k \rightarrow \lambda_k/\lambda_{20}$ for $k = 1, \dots, 20$ at each iteration of our MCMC output. As λ_{20} now takes its true value we can compare our posterior marginals (for the remaining skill parameters) relative to the true values chosen at the beginning of this study, $\boldsymbol{\lambda} = (20, 19, \dots, 1)$. Figure 2.2 depicts boxplots of the marginal posterior distribution for each $\log \lambda_k$. The distributions corresponding to the analyses of Datasets 1 and 2 are shown in white and red respectively. The blue crosses denote the true values from which the (informative) rankings were simulated. We also make it clear that, due to our rescaling, λ_{20} is constant and therefore omitted from the plot. Furthermore, outliers, defined as those observations further into the tail than 1.5 times the inter quartile range (IQR) from either the upper or lower quartiles, have also been omitted.

Under the analysis of Dataset 1 we observe that the posterior marginal distributions typically have significant support for the true parameter values. This is not particularly surprising given these data were simulated from the true model. There are of course some exceptions; the marginal posterior distributions for entities 3, 5 and 16 show significant

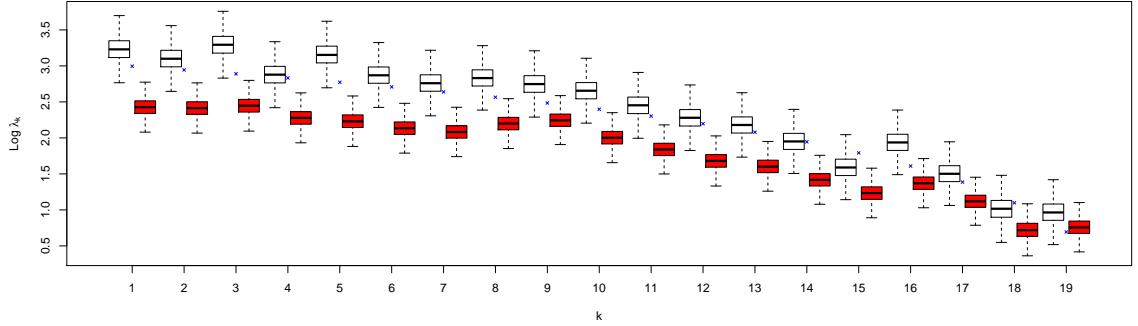


Figure 2.2: Boxplots summarising the marginal posterior densities for each $\log \lambda_k$ given that $\lambda_{20} = 1$. The densities in each case are shown in white and red for Datasets 1 and 2 respectively. The blue crosses depict the true values from which these data were simulated (log scale).

support for larger values of λ than the values from which these data were generated. In other words the analysis suggests these entities are “stronger” (or more preferred) than we know they actually are. We believe this is a feature of these data and had we analysed a larger dataset consisting of say $n = 1000$ rankings we would expect our marginal posteriors to be somewhat more focussed around the true values. From the analysis of Dataset 1 we can conclude that the Plackett–Luce model (and our associated sampling scheme) are capable of making reasonable inferences from a set of ranked data.

We now consider the analysis of Dataset 2, it is interesting to see how the introduction of an additional 10 uninformative rankings has a significant effect on our marginal posterior distributions; see Figure 2.2. In this analysis the marginal posterior distributions often show little support for the true values from which the 40 informative rankings were simulated. However it is worth nothing that, for this analysis at least, the model is still able to detect a similar (downward) trend in the preference of the entities as to that under the analysis of Dataset 1. The trend however does appear less significant and the uninformative rankings seem to have induced a “flattening” effect, that is, the (relative) differences between our marginal posterior distributions are less compelling. The result of this is that there is more (posterior) uncertainty on the preference order of the entities. This is perhaps not surprising given the uninformative rankings each express a random preference and therefore our model takes this into account by increasing uncertainty.

Often the aim/purpose of analysing ranked data is to obtain a so-called *aggregate* ranking. An aggregate ranking is a single ranking that summarises the preferences across all rankings contained within a particular dataset; to this extent it could be interpreted as an “average” ranking. There are numerous ways in which to obtain such a ranking. Here we choose to form our aggregate ranking by ordering the entities based upon their marginal posterior mean. The aggregate ranking, which we denote x^{agg} , is therefore equivalent to

$\hat{\mathbf{x}}$	$\boldsymbol{\lambda}$	Dataset 1		Dataset 2	
		$\mathbf{x}_1^{\text{agg}}$	$\bar{\boldsymbol{\lambda}}_1$	$\mathbf{x}_2^{\text{agg}}$	$\bar{\boldsymbol{\lambda}}_2$
1	20.00	3	27.47	3	11.67
2	19.00	1	25.83	1	11.44
3	18.00	5	23.90	2	11.29
4	17.00	2	22.66	4	9.84
5	16.00	4	18.11	9	9.54
6	15.00	6	17.98	5	9.41
7	14.00	8	17.31	8	9.11
8	13.00	7	16.12	6	8.55
9	12.00	9	15.91	7	8.10
10	11.00	10	14.50	10	7.48
11	10.00	11	11.84	11	6.36
12	9.00	12	9.95	12	5.42
13	8.00	13	9.00	13	5.02
14	7.00	14	7.17	14	4.17
15	6.00	16	7.08	16	3.98
16	5.00	15	4.99	15	3.47
17	4.00	17	4.58	17	3.10
18	3.00	18	2.81	19	2.16
19	2.00	19	2.68	18	2.08
20	1.00	20	1.00	20	1.00

Table 2.2: Aggregate rankings under our analysis of Datasets 1 and 2 along with the corresponding posterior means (denoted $\bar{\boldsymbol{\lambda}}$). The value of $\boldsymbol{\lambda}$ which was used to simulate these data is also reproduced for ease of comparison.

the “optimal” ranking given $\bar{\boldsymbol{\lambda}}$ where $\bar{\boldsymbol{\lambda}} = (\bar{\lambda}_1, \dots, \bar{\lambda}_K)$ is the parameter vector containing the marginal posterior means for each entity. Formally $\mathbf{x}^{\text{agg}} = \hat{\mathbf{x}}|\bar{\boldsymbol{\lambda}}$ where $\hat{\mathbf{x}}|\boldsymbol{\lambda}$ is as in (2.7).

Table 2.2 provides the aggregate rankings for the analyses of Datasets 1 and 2 (denoted $\mathbf{x}_1^{\text{agg}}$ and $\mathbf{x}_2^{\text{agg}}$ respectively) along with the corresponding marginal posterior means ($\bar{\lambda}_1$ and $\bar{\lambda}_2$). To ease comparison the true values from which these data were simulated along with the “optimal” ranking based upon the true values ($\hat{\mathbf{x}}$) are also given. We observe how the aggregate rankings under both analyses are coherent with the optimal ranking; particularly for the entities which are ranked at least tenth. The Kendall-tau distance, $K_\tau(a, b)$, is a measure of distance which is defined for any two orderings, a and b . The value of the Kendall-tau distance is equivalent to the number of adjacent (bubble sort) swaps which must be performed to b such that it becomes aligned with a . It is therefore a useful distance to use when comparing rankings (Marden, 1995). We can compute the Kendall-tau distance between the optimal ranking and the aggregate rankings under each analysis giving $K_\tau(\hat{\mathbf{x}}, \mathbf{x}_1^{\text{agg}}) = 6$ and $K_\tau(\hat{\mathbf{x}}, \mathbf{x}_2^{\text{agg}}) = 10$. Therefore we conclude that the

analysis of Dataset 1 results in an aggregate ranking that is, in some sense, more similar to the true preference ordering in comparison to the equivalent summaries based upon Dataset 2.

Another interesting feature of the posterior distributions is that although the aggregate rankings are somewhat similar the marginal posterior means of the entities are significantly different within each analysis; see Table 2.2 and the boxplots shown in Figure 2.2. In other words, although $\bar{\lambda}_1$ and $\bar{\lambda}_2$ define different distributions over rankings, the modal ranking is similar under each parameter vector. However, as the marginal posterior means (of the skill parameters) are significantly less dispersed under the analysis of Dataset 2, this suggests an increased level of uncertainty about an entity's position. To summarise this uncertainty we look at the probability of the modal ranking (x^{agg}), calculated using the posterior means of the skill parameters ($\bar{\lambda}$), relative to the same probability calculated under the uniform distribution. The probability of any (complete) ranking x under the uniform distribution is $1/K!$ and so the quantity of interest is $r = K! \Pr(X = x^{\text{agg}} | \bar{\lambda})$. The idea is that large values of r indicate that the modal ranking has much larger (posterior) support than a uniform ranking and so we can conclude that the ranking distribution (defined by $\bar{\lambda}$) is, in some sense, more concentrated around the modal ranking. In contrast, small values of r suggest that the ranking distribution is not as concentrated around the modal ranking and so there is more variation within the ranking distribution. In other words, the differences between the probabilities of different rankings are much smaller. Small values of r therefore indicate increased levels of uncertainty about the position of entities within the ranking. Note that $r = 1$ corresponds to the uniform distribution over rankings and so in this case each ranking is equally likely. For the analyses considered here we obtain $r_1 = 103625$ and $r_2 = 12364$ for Datasets 1 and 2, and so the probability of the aggregate ranking under the analysis of Dataset 1 is over one hundred thousand times larger than the probability of a uniform ranking, with this reducing to around twelve thousand times for the analysis of Dataset 2. It follows that, although the aggregate rankings under both analyses are similar there is much more posterior support for the aggregate ranking under the analysis of Dataset 1. This again highlights how the uninformative rankings in Dataset 2 have weakened our inferences on the skill parameters.

To conclude, this study has shown how outlying rankings can have a significant effect on our posterior distribution. This is a feature of our model which is not particularly desirable. A more robust model would be one which was more flexible and had the capability to allow for potential heterogeneity between the strength of particular rankings. In the next section we shall detail an extension of the Plackett–Luce model which allows us to account for such potential heterogeneity.

2.5 The Weighted Plackett–Luce model

We have alluded to and indeed shown through our simulation study in Section 2.4 that contaminating our data with random permutations can have a significant effect on posterior beliefs. This is a feature of the Plackett–Luce model which is not particularly desirable. Ideally we would like a model where our posterior inferences are not significantly affected by a few spurious rankings/observations. In this section we describe a novel extension to the standard Plackett–Luce model which aims to allow for potential heterogeneity between the amount of information contained within particular rankings.

Before outlining such a model it is natural to recast this problem in terms of ranker reliability. From the form of the Plackett–Luce likelihood (2.3) it is clear how each ranking makes an equally weighted contribution to the overall likelihood. In other words the model considers each ranker to be equally informative/reliable. This is a rather strong assumption. It is easy to conceive of a scenario in which some rankers are significantly more informed about the entities they are ranking compared to other rankers. In the remainder of this section we propose an extension to the Plackett–Luce model so that rankings no longer make an equal contribution to the overall likelihood. The resulting model is one which enables some rankings to have a larger influence over our parameter inferences than others, and so allows for potential heterogeneity between the ability of rankers.

We choose to model this potential heterogeneity between rankings via a mixture model with two components: one for “*informative rankings*” and the other for “*uninformative rankings*”. The mixture model is defined using latent binary indicator variables $W_i \in \{0, 1\}$ for $i = 1, \dots, n$. We let $W_i = 0$ if ranking i is uninformative and $W_i = 1$ otherwise. The probability of a particular ranking (conditional on the skill parameters and the latent binary indicator variable) under this “Weighted Plackett–Luce model” is

$$\Pr(\mathbf{X}_i = \mathbf{x}_i | \boldsymbol{\lambda}, W_i = w_i) = \prod_{j=1}^{n_i} \frac{\lambda_{x_{ij}}^{w_i}}{\sum_{m=j}^{n_i} \lambda_{x_{im}}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_m^{w_i}}, \quad (2.8)$$

whence for an informative ranking ($w_i = 1$) we recover (2.2), the standard Plackett–Luce probability. However, for an uninformative ranking ($w_i = 0$) we have

$$\begin{aligned} \Pr(\mathbf{X}_i = \mathbf{x}_i | \boldsymbol{\lambda}, W_i = 0) &= \prod_{j=1}^{n_i} \frac{\lambda_{x_{ij}}^0}{\sum_{m=j}^{n_i} \lambda_{x_{im}}^0 + \sum_{m \in \mathcal{U}_i} \lambda_m^0}, \\ &= \frac{(K_i - n_i)!}{K_i!}, \\ &= \frac{1}{P(K_i, n_i)}, \end{aligned} \quad (2.9)$$

that is, the reciprocal of the number of ordered permutations of n_i entities from a set of size K_i . The implication of a ranking being deemed uninformative under this model results in its contribution to the likelihood being constant regardless of the values of the skill parameters – this should be clear as $\Pr(\mathbf{X}_i = \mathbf{x}_i | \boldsymbol{\lambda}, W_i = 0)$ does not depend on $\boldsymbol{\lambda}$. Therefore $W_i = 0$ corresponds to there being no information in ranking i : essentially ranker i has picked \mathbf{x}_i uniformly at random from all possible rankings of n_i out of K_i entities. We write this probability model using the notation $\mathbf{X}_i | \boldsymbol{\lambda}, w_i \sim \text{PLW}(\boldsymbol{\lambda}, w_i)$.

At first glance taking w_i to be binary may appear to be fairly restrictive as, with $w_i = 0$, the Weighted Plackett–Luce model assumes that the entire ranking \mathbf{x}_i is completely uninformative. To allow more flexibility we did consider allowing each ranker to have a different binary variable in each position of their ranking by introducing position dependent binary indicators w_{ij} for $i = 1, \dots, n$, $j = 1, \dots, n_i$. However, we judged that this would introduce far too many parameters and lead to identifiability issues, particularly when we consider clustering rankers and entities in Chapters 3 and 4. We also considered allowing the weight parameters w_i to be continuous, such as in the unit interval. However this too is problematic as it not only renders the weights uninterpretable (in terms of ranker reliability) but also introduces problems of identifiability. To see this problem it is useful to consider a simple example. We begin by noting that the Weighted Plackett–Luce probability is simply the standard Plackett–Luce probability evaluated at the skill parameters $\boldsymbol{\lambda}^{w_i}$, and so the probabilities of different entity rankings for ranker i are described by the $\text{PL}(\boldsymbol{\lambda}^{w_i})$ distribution. Now consider two rankers i and j and let the skill parameter vector be $\boldsymbol{\lambda}$. Suppose these rankers have weight parameters $w_i = 0.8$ and $w_j = 0.5$ and so here the ranking distributions are $\text{PL}(\boldsymbol{\lambda}^{w_i=0.8})$ and $\text{PL}(\boldsymbol{\lambda}^{w_j=0.5})$. However equivalent ranking distributions (and hence the same weighted Plackett–Luce likelihood) could be obtained by using $\boldsymbol{\lambda}^* = \boldsymbol{\lambda}^{0.8}$, with $w_i = 1$ and $w_j = 0.5/0.8 = 0.625$. This simple example shows an identifiability problem for $(\boldsymbol{\lambda}, \mathbf{w})$. Also, the value of w_i is not meaningful as ranker i would be classed as fairly informative in the $\boldsymbol{\lambda}$ setting but extremely informative in the $\boldsymbol{\lambda}^*$ setting. These issues do not occur if we choose w_i to be binary. Also this choice has the benefit that $w_i = 1$ recovers the standard Plackett–Luce distribution and $w_i = 0$ is meaningful in that it represents a uniform ranking distribution.

It is also clear that the Weighted Plackett–Luce (WPL) model (2.8) differs from the model proposed by Benter (1994). The WPL model considers ranker-specific weights (w_i , for $i = 1, \dots, n$) to allow for potential heterogeneity between rankers' abilities, whereas the Benter model considers *position dependent* "weight" parameters (w_j , for $j = 1, \dots, K$) that are common across all rankers and represent the "importance" of each stage in the ranking process. In theory it would also be possible to introduce both sets of weight parameters and hence consider a Weighted Benter model. However this is likely to give rise to identifiability issues and so this model is not considered any further within this thesis.

2.5.1 Simulating data from the Weighted Plackett–Luce model

Our underlying probability model has changed as a result of introducing binary indicator variables to reflect the (latent) ability of rankers. It follows that the data generating mechanism must also be adapted to allow for this additional (potential) heterogeneity. We generalise the well-known exponential latent variable representation of the standard Plackett–Luce model and, conditional on the skill parameters $\boldsymbol{\lambda}$ and the indicator variable $w \in \{0, 1\}$, the corresponding latent arrival times under the Weighted Plackett–Luce model are

$$\nu_j \stackrel{\text{indep}}{\sim} \text{Exp}(\lambda_j^w), \quad (2.10)$$

for $j = 1, \dots, K$.

A complete ranking \mathbf{x} can then be simulated from under the Weighted Plackett–Luce model as outlined in Section 2.2.5 where ν_j is instead drawn from (2.10) in step 1. We also note that, although Dataset 2 from the previous simulation study (Section 2.4) was simulated from under the standard Plackett–Luce model, these data follow the underlying ranking distribution defined by the Weighted Plackett–Luce model with $\boldsymbol{\lambda} = (20, 19, \dots, 1)$ and $w_i = 1$ for $i = 1, \dots, 40$; $w_i = 0$ for $i = 41, \dots, 50$.

2.6 Bayesian inference

Bayesian inference for the parameters in the Weighted Plackett–Luce model is achieved in a similar manner to that for the standard Plackett–Luce model (see Section 2.3). However, here we assume that the latent binary indicators are unknown (note that we essentially assumed $w_i = 1 \forall i$ for the standard Plackett–Luce model). It follows that we have additional random quantities in the model, namely $\mathbf{w} = \{w_i\}_{i=1}^n$, which are also to be inferred from the data. As with the standard Plackett–Luce model, the likelihood is formed by taking the product of the respective probabilities for each of the n rankings, and so

$$\pi(\mathcal{D}|\boldsymbol{\lambda}, \mathbf{w}) = \prod_{i=1}^n \prod_{j=1}^{n_i} \frac{\lambda_{x_{ij}}^{w_i}}{\sum_{m=j}^{n_i} \lambda_{x_{im}}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_m^{w_i}}, \quad (2.11)$$

where $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^n$ denotes the collection of all rankings.

Again, as with the standard Plackett–Luce model, a maximum likelihood approach could be implemented if desired. For this model we would of course have to make appropriate adaptations to the optimisation schemes within the literature to allow for our additional latent indicator variables. However here we proceed within the Bayesian framework, define a suitable prior distribution and apply Bayes' Theorem to obtain our posterior distribution.

2.6.1 Prior specification and latent variables

We elect to take the same prior specification for our skill parameters as in Section 2.3.1 for the reasons discussed therein. However, for this model we must also specify a suitable prior distribution over our latent binary indicator variables \mathbf{w} . As $w_i \in \{0, 1\}$ we choose $w_i \stackrel{\text{indep}}{\sim} \text{Bern}(p_i)$ where $p_i \in (0, 1]$ is the probability that ranking i is informative *a priori*. Note that we omit the choice of $p_i = 0$ as this implies that $\Pr(w_i = 0 | \mathcal{D}) = 1$; whence the likelihood of ranking i is constant irrespective of $\boldsymbol{\lambda}$. The contribution to the likelihood from ranking i can therefore be absorbed into the constant of proportionality when applying Bayes' Theorem. It follows that, if we truly believe a ranking is uninformative (with probability 1) it is sufficient to simply omit this ranking from our analysis. If defining probabilities p_i *a priori* is not desired then a hierarchical model structure could be constructed; given that p_i is a probability, a Beta distribution would be a sensible choice however this is not considered here. Our complete model specification is therefore

$$\begin{aligned} \mathbf{X}_i | \boldsymbol{\lambda}, \mathbf{w} &\stackrel{\text{indep}}{\sim} \text{PLW}(\boldsymbol{\lambda}, w_i) & i = 1, \dots, n, \\ \lambda_k &\stackrel{\text{indep}}{\sim} \text{Ga}(a_k, 1) & k = 1, \dots, K, \\ w_i &\stackrel{\text{indep}}{\sim} \text{Bern}(p_i) & i = 1, \dots, n. \end{aligned}$$

As with the standard Plackett–Luce model, it is possible to augment our sample space so that a conjugate update for our skill parameters $\boldsymbol{\lambda}$ can be achieved. The form of the likelihood has changed since we introduced our latent ranker weights and we must therefore also modify our latent variable specification. The appropriate latent variables to introduce for this model are

$$z_{ij} | \mathcal{D}, \boldsymbol{\lambda}, \mathbf{w} \stackrel{\text{indep}}{\sim} \text{Exp} \left(\sum_{m=j}^{n_i} \lambda_{x_{im}}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_m^{w_i} \right), \quad (2.12)$$

for $i = 1, \dots, n$ and $j = 1, \dots, n_i$.

2.6.2 Full conditional distributions

Again we proceed to derive the full conditional distributions for each of our random variables by first constructing the density of all stochastic quantities. This is given by

$$\begin{aligned}
 \pi(\boldsymbol{\lambda}, \mathcal{D}, Z, \mathbf{w}) &= \pi(Z|\mathcal{D}, \boldsymbol{\lambda}, \mathbf{w})\pi(\mathcal{D}|\boldsymbol{\lambda}, \mathbf{w})\pi(\boldsymbol{\lambda})\pi(\mathbf{w}) \\
 &= \prod_{i=1}^n \prod_{j=1}^{n_i} \left(\sum_{m=j}^{n_i} \lambda_{x_{im}}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_m^{w_i} \right) \exp \left\{ -z_{ij} \left(\sum_{m=j}^{n_i} \lambda_{x_{im}}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_m^{w_i} \right) \right\} \\
 &\quad \times \prod_{i=1}^n \prod_{j=1}^{n_i} \frac{\lambda_{x_{ij}}^{w_i}}{\sum_{m=j}^{n_i} \lambda_{x_{im}}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_m^{w_i}} \times \prod_{k=1}^K \frac{\lambda_k^{a_k-1} e^{-\lambda_k}}{\Gamma(a_k)} \times \prod_{i=1}^n p_i^{w_i} (1-p_i)^{1-w_i} \\
 &= \prod_{k=1}^K \frac{\lambda_k^{a_k-1} e^{-\lambda_k}}{\Gamma(a_k)} \times \prod_{i=1}^n p_i^{w_i} (1-p_i)^{1-w_i} \\
 &\quad \times \prod_{i=1}^n \prod_{j=1}^{n_i} \lambda_{x_{ij}}^{w_i} \exp \left\{ -z_{ij} \left(\sum_{m=j}^{n_i} \lambda_{x_{im}}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_m^{w_i} \right) \right\}. \quad (2.13)
 \end{aligned}$$

Unsurprisingly, given that the latent variables introduced are defined through their full conditional distributions, we observe from (2.13) that the FCD for the z_{ij} is as in (2.12) for $i = 1, \dots, n$, $j = 1, \dots, n_i$.

The full conditional distribution for $\boldsymbol{\lambda}$ is

$$\begin{aligned}
 \pi(\boldsymbol{\lambda}|\mathcal{D}, Z, \mathbf{w}) &\propto \prod_{k=1}^K \lambda_k^{a_k-1} e^{-\lambda_k} \prod_{i=1}^n \prod_{j=1}^{n_i} \lambda_{x_{ij}}^{w_i} \exp \left\{ -z_{ij} \left(\sum_{m=j}^{n_i} \lambda_{x_{im}}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_m^{w_i} \right) \right\} \\
 &= \prod_{k=1}^K \lambda_k^{a_k + \tilde{q}_k - 1} \exp \left\{ - \left(1 + \sum_{i=1}^n w_i \sum_{j=1}^{n_i} \delta_{ij}(k) z_{ij} \right) \lambda_k \right\},
 \end{aligned}$$

where

$$\tilde{q}_k = \sum_{i=1}^n w_i \mathbb{I}(k \in \{\mathbf{x}_i\}),$$

is the number of times entity k appears in an informative ranking. As in our previous analysis $\delta_{ij}(k)$ is an indicator variable over the event that entity k receives a rank no better than j in ranking i and is given by (2.6). From this we can obtain the FCDs for our skill parameters as

$$\lambda_k | \mathcal{D}, Z, \mathbf{w} \stackrel{\text{indep}}{\sim} \text{Ga} \left(a_k + \tilde{q}_k, 1 + \sum_{i=1}^n w_i \sum_{j=1}^{n_i} \delta_{ij}(k) z_{ij} \right), \quad \text{for } k = 1, \dots, K.$$

The Weighted Plackett–Luce model also contains the additional binary indicator variables. These variables follow a discrete distribution with two components. Let $\mathbf{w}_{-i} = (w_1, \dots, w_{i-1}, w_{i+1}, \dots, w_n)$ denote the vector which contains all the indicator values except that associated with ranker i . It follows that

$$\begin{aligned}\Pr(w_i = 1 | \mathcal{D}, \boldsymbol{\lambda}, Z, \mathbf{w}_{-i}) &\propto \Pr(w_i = 1) \pi(Z | w_i = 1, \mathcal{D}, \boldsymbol{\lambda}, \mathbf{w}_{-i}) \Pr(\mathcal{D} | w_i = 1, \boldsymbol{\lambda}, \mathbf{w}_{-i}) \\ &= p_i \prod_{i=1}^n \prod_{j=1}^{n_i} \lambda_{x_{ij}} \exp \left\{ -z_{ij} \left(\sum_{m=j}^{n_i} \lambda_{x_{im}} + \sum_{m \in \mathcal{U}_i} \lambda_m \right) \right\} \\ &\propto p_i \prod_{j=1}^{n_i} \lambda_{x_{ij}} \exp \left\{ -z_{ij} \left(\sum_{m=j}^{n_i} \lambda_{x_{im}} + \sum_{m \in \mathcal{U}_i} \lambda_m \right) \right\},\end{aligned}$$

and

$$\begin{aligned}\Pr(w_i = 0 | \mathcal{D}, \boldsymbol{\lambda}, Z, \mathbf{w}_{-i}) &\propto \Pr(w_i = 0) \pi(Z | w_i = 0, \mathcal{D}, \boldsymbol{\lambda}, \mathbf{w}_{-i}) \Pr(\mathcal{D} | w_i = 0, \boldsymbol{\lambda}, \mathbf{w}_{-i}) \\ &= (1 - p_i) \prod_{i=1}^n \prod_{j=1}^{n_i} \lambda_{x_{ij}}^0 \exp \left\{ -z_{ij} \left(\sum_{m=j}^{n_i} \lambda_{x_{im}}^0 + \sum_{m \in \mathcal{U}_i} \lambda_m^0 \right) \right\} \\ &\propto (1 - p_i) \prod_{j=1}^{n_i} \exp \left\{ -z_{ij} \left(\sum_{m=j}^{n_i} 1 + \sum_{m \in \mathcal{U}_i} 1 \right) \right\} \\ &= (1 - p_i) \prod_{j=1}^{n_i} \exp \{ -z_{ij}(K_i - j + 1) \}.\end{aligned}$$

Hence the (discrete) full conditional distribution for w_i is

$$w_i | \mathcal{D}, \boldsymbol{\lambda}, Z \stackrel{\text{indep}}{\sim} \text{Bern}(\rho_i),$$

where

$$\rho_i = \frac{\Pr(w_i = 1 | \mathcal{D}, \boldsymbol{\lambda}, Z, \mathbf{w}_{-i})}{\Pr(w_i = 1 | \mathcal{D}, \boldsymbol{\lambda}, Z, \mathbf{w}_{-i}) + \Pr(w_i = 0 | \mathcal{D}, \boldsymbol{\lambda}, Z, \mathbf{w}_{-i})}, \quad (2.14)$$

is the probability that ranking i is informative (given the other quantities).

2.6.3 MCMC - Gibbs sampling via latent variables

In the previous section we derived a complete set of full conditional distributions for each random quantity within our model. As with the standard Plackett–Luce model we are able to implement a Gibbs sampler to generate realisations from our posterior distribution. For brevity and clarity of reading we omit the iteration counter t when outlining this algorithm. However, the updates proceed in the same manner as the algorithm outlined

in Section 2.3.3. The algorithm proceeds as follows.

1. Initialise the state of the chain. One possibility is as follows.

- For $k = 1, \dots, K$, sample $\lambda_k \stackrel{\text{indep}}{\sim} \text{Ga}(a_k, 1)$.
- For $i = 1, \dots, n$, sample $w_i \stackrel{\text{indep}}{\sim} \text{Bern}(p_i)$.
- For $i = 1, \dots, n$, $j = 1, \dots, n_i$, sample

$$z_{ij} | \boldsymbol{\lambda}, \mathbf{w} \stackrel{\text{indep}}{\sim} \text{Exp} \left(\sum_{m=j}^{n_i} \lambda_{x_{im}}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_m^{w_i} \right).$$

2. Update the state of the chain by repeatedly performing the following steps.

- For $k = 1, \dots, K$, sample

$$\lambda_k | \mathcal{D}, Z, \mathbf{w} \stackrel{\text{indep}}{\sim} \text{Ga} \left(a_k + \tilde{q}_k, 1 + \sum_{i=1}^n w_i \sum_{j=1}^{n_i} \delta_{ij}(k) z_{ij} \right).$$

- For $i = 1, \dots, n$, $j = 1, \dots, n_i$, sample

$$z_{ij} | \mathcal{D}, \boldsymbol{\lambda}, \mathbf{w} \stackrel{\text{indep}}{\sim} \text{Exp} \left(\sum_{m=j}^{n_i} \lambda_{x_{im}}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_m^{w_i} \right).$$

- For $i = 1, \dots, n$, sample

$$w_i | \mathcal{D}, \boldsymbol{\lambda}, Z \stackrel{\text{indep}}{\sim} \text{Bern}(\rho_i),$$

where ρ_i is given by (2.14).

- Rescale:

- Sample $\Lambda^\dagger \sim \text{Ga} \left(\sum_{k=1}^K a_k, 1 \right)$.
- Calculate $\Sigma = \sum_{k=1}^K \lambda_k$.
- For $k = 1, \dots, K$, let $\lambda_k \rightarrow \lambda_k \Lambda^\dagger / \Sigma$.

We note here that, unlike in the standard Plackett–Luce analysis, \tilde{q}_k is no longer a function of the data alone. The value now depends on the random variables \mathbf{w} and so the computation of \tilde{q}_k is required at each iteration of our MCMC scheme; specifically after new realisations of w_i have been drawn. On the other hand, the indicators $\delta_{ij}(k)$ remain only a function of the data and therefore can be computed at step 1 and used throughout the MCMC scheme.

2.7 Simulation study

In this study we perform Bayesian inference assuming our rankings follow the Weighted Plackett–Luce model. Primarily we focus on whether this model is able to (correctly) identify uninformative rankings contained within a given dataset. In our previous study (Section 2.4) we observed how our posterior beliefs were significantly altered by the contamination of our dataset with random permutations. With this in mind it will be interesting to see how our Weighted Plackett–Luce model performs given it has the flexibility to down-weight the influence of particular rankings on our parameter inference.

Given these aims we revisit a dataset from our previous simulation study on the standard Plackett–Luce model; namely Dataset 2 from Section 2.4. Recall that Dataset 2 contains $n = 50$ complete rankings of $K = 20$ entities. Forty of the rankings (1–40) are informative and thus follow the Weighted Plackett–Luce model subject to $w_i = 1$. The remaining rankings (41–50) are uninformative rankings, that is, random permutations of the K entities. These are considered to follow the Weighted Plackett–Luce model subject to $w_i = 0$. We also remind the reader that the optimal ranking, that is, the ranking which maximises the Plackett–Luce probability, is $\hat{\mathbf{x}} = (1, 2, \dots, 20)$ under the parameters used to simulate these data. Note that, strictly speaking, these data were not generated from the data generating process for the Weighted Plackett–Luce model. However as discussed within Section 2.5.1 the process used to generate Dataset 2 is equivalent to using the data generating process for the Weighted Plackett–Luce model subject to the choice of $\lambda = (20, 19, \dots, 1)$, $w_i = 1$ for $i = 1, \dots, 40$ and $w_i = 0$ for $i = 41, \dots, 50$.

Within this study we choose to use the prior specification as outlined in Section 2.6.1, that is, independent Gamma prior distributions on our skill parameters and independent Bernoulli distributions on our latent ranker weights. Furthermore we also wish to make the same assumption regarding our skill parameters as in our previous analysis; namely that each ranking is equally likely *a priori*. Given this we let $a_k = a = 1$ for $k = 1, \dots, K$, which gives $\lambda_k \stackrel{\text{indep}}{\sim} \text{Ga}(1, 1)$. We shall now consider two separate analyses of these data, each of which has an alternative prior specification on our latent binary indicator variables. In Analysis 1 we choose to assume that each ranking is equally likely to be informative as it is uninformative; hence $p_i = 0.5$. We note that this choice is not in line with these data as the true proportion of informative rankings within this dataset is $40/50 = 0.8$. Thus we let $p_i = 0.8$ *a priori* in Analysis 2.

2.7.1 Posterior analysis

Realisations from our posterior distribution (for both analyses) are obtained by implementing the Gibbs sampler detailed in Section 2.6.3. For each analysis the Markov chain

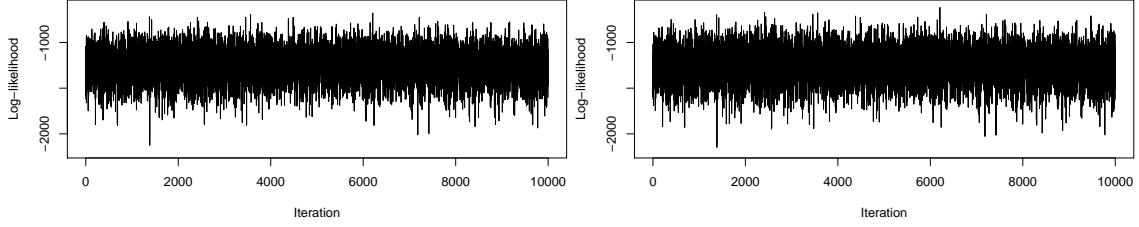


Figure 2.3: Trace plots of the log complete data likelihood for Analyses 1 and 2 ($p_i = 0.5, 0.8$) from left to right respectively.

is initialised at a random draw from the prior distribution. Each chain runs for 11K iterations; the first 1K of which are discarded as burn-in. This results in 10K (almost) un-autocorrelated realisations from our posterior distribution. As for previous analyses our inference scheme is implemented in C and computation is performed on a single thread of an Intel Core i7-4790S CPU (3.20GHz clock speed). The computational time required to perform inference on these data is approximately 1.9 seconds for both analyses. The mixing of our MCMC chains is assessed by inspecting the trace plots of the log complete data likelihood; see Figure 2.3. From this we observe that our chains appear to be mixing well over the sample space of the random quantities. Convergence has been verified by initialising each chain at numerous starting values and checking the posterior realisations are equivalent (up to stochastic noise) in all cases.

We begin our investigation into the posterior distribution by summarising the marginal posterior distributions for each of our ranker weights w_i . Figure 2.4 depicts the posterior probability that each ranker is informative, that is, $\Pr(w_i = 1|\mathcal{D})$. This probability is obtained by taking the posterior mean of ρ_i and not simply the posterior expectation of w_i . Our probability is therefore a result of a Rao–Blackwellized estimator which typically provide us with a better estimate than simply taking the (posterior) mean of w_i (Casella and Robert, 1996). The Kendall-tau distance, $K_\tau(\hat{\mathbf{x}}, \mathbf{x}_i)$, between each of our simulated rankings and the optimal ranking is also given in Figure 2.4. We observe that, with the exception of ranking 42, all the uninformative rankings (41–50) receive a lower posterior probability of being informative than specified *a priori* under each respective analysis. As expected these rankings are also those which typically have a larger distance from the optimal ranking. It is encouraging to see that those rankings which are informative (1–40) almost always obtain large posterior probabilities of being informative, particularly under Analysis 2 where $p_i = 0.8$. That said, the posterior probabilities for (informative) ranker 8 are not close to 1 in either analysis. Closer inspection of this ranking reveals that it is somewhat atypical of this dataset: entities 2 and 4 both appear within the bottom 5 positions and entities 13, 11 and 10 all feature within the top 5 positions. These features are somewhat at odds with the true parameter values from which these data were

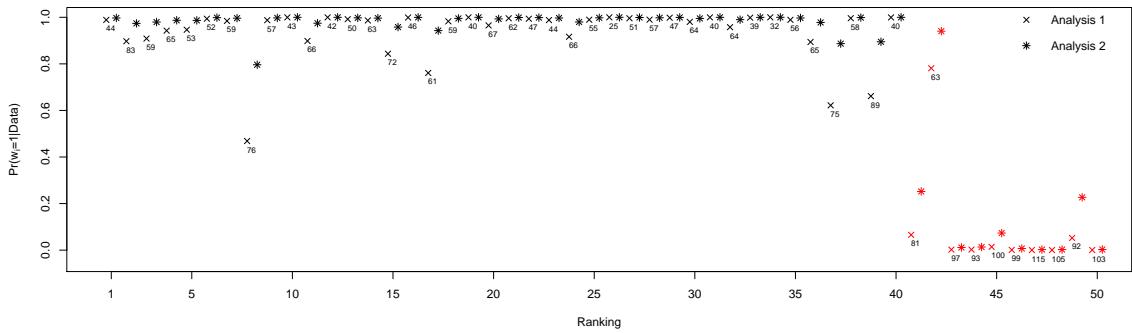


Figure 2.4: $\Pr(w_i = 1 | \mathcal{D})$ – Posterior probability that ranking i is informative under each analysis (Analysis 1: $p_i = 0.5$, Analysis 2: $p_i = 0.8$). Rankings which are random permutations (41–50) are shown in red. Numbers shown denote the Kendall-tau distance between each ranking and \hat{x} .

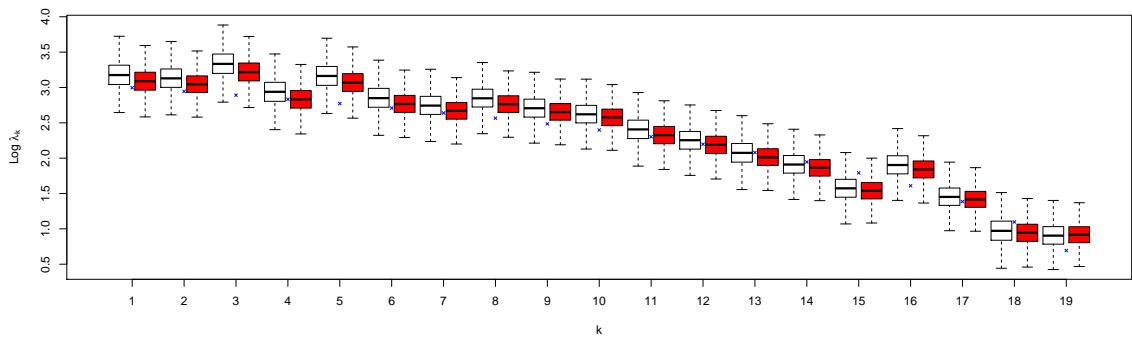


Figure 2.5: Boxplots summarising the marginal posterior densities for each $\log \lambda_k$ given that $\lambda_{20} = 1$. The boxplots in each case are shown in white and red for Analysis 1 and 2 ($p_i = 0.5, 0.8$) respectively. The blue crosses depict the true values, λ_k , from which these data were simulated.

simulated. Therefore it appears that the Weighted Plackett–Luce model is able to correctly identify those rankings which are somewhat spurious and down-weight their contribution to the likelihood appropriately.

Figure 2.5 depicts boxplots of the marginal posterior distribution of $\log \lambda_k$ for Analyses 1 and 2 (shown in white and red respectively). As in our analyses under the standard Plackett–Luce model we have rescaled our posterior realisations so that λ_{20} takes its true value, that is, set $\lambda_k \rightarrow \lambda_k/\lambda_{20}$. It is clear that the posterior marginal distributions are comparable across the two analyses; this does not come as a surprise given that the posterior probabilities of each ranker being informative are similar under both prior specifications. Furthermore the preference ordering of the entities has also been identified by the model; this is clearly seen through the downward trend in Figure 2.5 as k increases.

Interestingly when we compare the marginal posterior distributions of the skill parameters under the Weighted Plackett–Luce model to those under the analysis of Dataset 1 assuming the standard Plackett–Luce model we see significant similarities; see Figure 2.6.

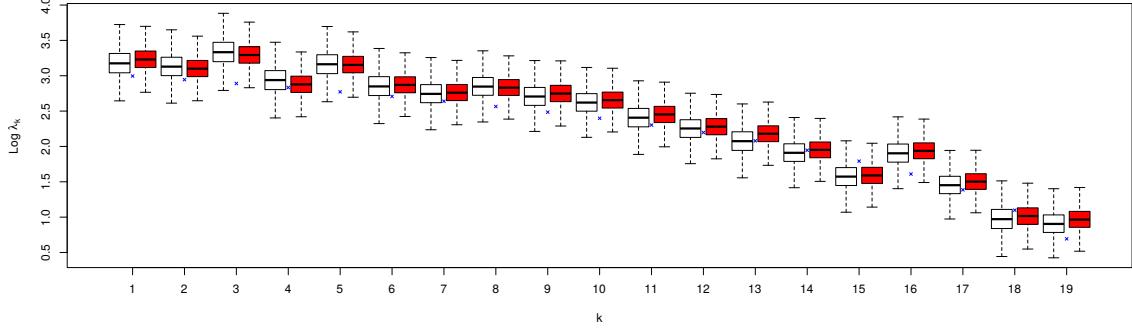


Figure 2.6: Boxplots summarising the marginal posterior densities for each $\log \lambda_k$ given that $\lambda_{20} = 1$. The boxplots in each case are shown in white for Analysis 1 under our Weighted Plackett–Luce model and in red for the analysis of Dataset 1 under the standard Plackett–Luce model. The blue crosses depict the true values, λ_k , from which these data were simulated.

It is evident that by adopting the Weighted Plackett–Luce model our posterior distribution is significantly less affected by the incorporation of uninformative rankings within our dataset. This becomes somewhat more clear if we consider the posterior aggregate rankings. Table 2.3 provides the aggregate rankings for the analyses of Dataset 2 under both prior specifications for our Weighted Plackett–Luce model; denoted $\mathbf{x}_3^{\text{agg}}$ and $\mathbf{x}_4^{\text{agg}}$ respectively. The corresponding marginal posterior means, $\bar{\lambda}_3$ and $\bar{\lambda}_4$, upon which these aggregates are formed are also given. To ease comparison the aggregate rankings ($\mathbf{x}_1^{\text{agg}}, \mathbf{x}_2^{\text{agg}}$) and the corresponding marginal posterior means ($\bar{\lambda}_1, \bar{\lambda}_2$) for the analyses of Datasets 1 and 2 under the standard Plackett–Luce model are also reproduced here. The true values from which these data were simulated along with the “optimal” ranking based upon the true values also feature in Table 2.3. Considering our two separate analyses under the Weighted Plackett–Luce model we observe how the aggregate rankings are equivalent under both prior specifications. Furthermore this aggregate ranking is also equivalent to that formed from the analysis of Dataset 1 under the standard Plackett–Luce model, that is, the analysis containing no spam rankings. This is particularly interesting as it shows how the effect of the spam rankings on our posterior beliefs has been negated by introducing our binary indicators into the Plackett–Luce probability.

In Section 2.4.1 we noted that it can be useful to look at the relative probability of the aggregate ranking in comparison to a uniform ranking ($r = K! \Pr(X = \mathbf{x}^{\text{agg}} | \bar{\lambda})$) for each analysis. This quantity provides insight into how concentrated the ranking distribution is around the (posterior) modal ranking. Table 2.3 provides the r_i for each analysis considered here, along with those for the analyses of Datasets 1 and 2 under the standard Plackett–Luce model. We observe that the values of the r_i under each of the Weighted Plackett–Luce analyses are similar to that for the analysis of Dataset 1 under the standard PL model. Again this highlights how adopting the Weighted Plackett–Luce model

$\hat{\boldsymbol{x}}$	$\boldsymbol{\lambda}$	PL _W				PL			
		$p_i = 0.5$		$p_i = 0.8$		Dataset 1		Dataset 2	
		$\boldsymbol{x}_3^{\text{agg}}$	$\bar{\boldsymbol{\lambda}}_3$	$\boldsymbol{x}_4^{\text{agg}}$	$\bar{\boldsymbol{\lambda}}_4$	$\boldsymbol{x}_1^{\text{agg}}$	$\bar{\boldsymbol{\lambda}}_1$	$\boldsymbol{x}_2^{\text{agg}}$	$\bar{\boldsymbol{\lambda}}_2$
1	20.00	3	28.83	3	25.55	3	27.47	3	11.67
2	19.00	1	24.60	1	22.41	1	25.83	1	11.44
3	18.00	5	24.19	5	21.96	5	23.90	2	11.29
4	17.00	2	23.40	2	21.43	2	22.66	4	9.84
5	16.00	4	19.35	4	17.32	4	18.11	9	9.54
6	15.00	6	17.73	6	16.19	6	17.98	5	9.41
7	14.00	8	17.62	8	16.17	8	17.31	8	9.11
8	13.00	7	15.93	7	14.70	7	16.12	6	8.55
9	12.00	9	15.31	9	14.46	9	15.91	7	8.10
10	11.00	10	14.06	10	13.40	10	14.50	10	7.48
11	10.00	11	11.37	11	10.44	11	11.84	11	6.36
12	9.00	12	9.71	12	9.08	12	9.95	12	5.42
13	8.00	13	8.16	13	7.63	13	9.00	13	5.02
14	7.00	14	6.91	14	6.57	14	7.17	14	4.17
15	6.00	16	6.88	16	6.42	16	7.08	16	3.98
16	5.00	15	4.93	15	4.74	15	4.99	15	3.47
17	4.00	17	4.38	17	4.19	17	4.58	17	3.10
18	3.00	18	2.71	18	2.62	18	2.81	19	2.16
19	2.00	19	2.52	19	2.54	19	2.68	18	2.08
20	1.00	20	1.00	20	1.00	20	1.00	20	1.00

r_i	129768	93809	103625	12364
-------	--------	-------	--------	-------

Table 2.3: Aggregate rankings under the Weighted Plackett–Luce model for the analysis of Dataset 2 (for both analyses $p_i = 0.5, 0.8$) along with the corresponding posterior means. To ease comparison, the results from Table 2.2 (standard Plackett–Luce analyses) are also given. The table also contains the relative probability of the aggregate rankings in comparison to a uniform ranking, $r_i = K! \Pr(X = \boldsymbol{x}_i^{\text{agg}} | \bar{\boldsymbol{\lambda}}_i)$.

results in our posterior distribution being significantly less affected by the inclusion of uninformative rankings within the dataset.

2.8 Summary

This chapter has outlined the Plackett–Luce model and discussed the underlying assumptions along with some of its limitations. Identifiability issues, and several possible solutions, were described with our preferred method of resolving this being to employ a suitable rescaling strategy within our posterior sampling algorithm. The Plackett–Luce probability was extended to deal with a much richer class of rankings (top and top partial rankings). By appealing to data augmentation techniques, an efficient Gibbs sampling strategy was made possible (subject to certain prior specifications) and a detailed outline of the algorithm to sample from the posterior distribution was given.

Numerous simulation studies were considered which revealed how, under the standard Plackett–Luce model, our (posterior) inferences can be substantially different if the data contain unusual (spam) rankings. This is an undesirable feature of the model and therefore in Section 2.5 we proposed the Weighted Plackett–Luce model which allows for the notion of ranker reliability through a latent binary indicator. We saw through simulation studies how inferences under the Weighted Plackett–Luce model are more robust to the addition of spam rankings. Moreover our model was able to correctly identify those rankings that were in some sense unusual.

Although the Weighted Plackett–Luce model allows for a certain level of ranker heterogeneity – namely that a small group of rankers that appear to have an alternative preference can be down weighted – the model is not sufficient to effectively handle a scenario where numerous groups of rankers express different preferences. It follows that even when modelling rankings using the Weighted Plackett–Luce model, we must still make the underlying assumption that all rankers share similar beliefs/preferences about the entities they are ranking. We believe however that this assumption is perhaps implausible in real world scenarios and in the next chapter we build more flexible models which allow for the assumption of homogeneous beliefs to be relaxed.

Chapter 3

Analysis of heterogeneous ranked data

3.1 Introduction

Until now we have assumed that a single parameter vector λ is sufficient to summarise the beliefs of all rankers contributing to a dataset. Further we have also made the assumption that each skill parameter λ_k ($k = 1, \dots, K$) is unique. We now suppose that there may be groups of rankers, each group with their own beliefs about the true preference ordering of the entities. To implement this structure each ranker has their own parameter vector λ_i . However, all rankers within the same ranker group share the same beliefs about the entities, and so all rankers within the group have the same skill parameter values. We further propose that particular ranker groups may not be able to distinguish between certain entities, that is, there may be group structure in the entities in the ranker groups. To allow for this we require the possibility that values within each parameter vector λ_i are not necessarily unique. We appeal to Bayesian non-parametric clustering methods to implement this structure, specifically by using Dirichlet processes. First however, we review methods for finite mixtures.

3.2 Finite mixture models

A common approach when analysing heterogeneous data is to appeal to mixture models. This rich class of models allow us to infer subgroups contained within data without (prior) information on subgroup membership of individual observations. A subgroup can be thought of as a *cluster* of individual observations which form a “homogeneous” group. Individual observations within each subgroup are assumed to follow the same underlying

distribution.

The simplest models within this class are *finite mixture models*; see, for example, Everitt and Hand (1981) and Lindsay (1995). Note that, for ease of notation and exposition, we write $f(\cdot|\cdot)$ for a density (or probability) function (depending on whether the quantity is continuous or discrete) but simply refer to these functions as densities. In finite mixture models the parametric density that defines the model, denoted $f(x)$, is comprised of a fixed (finite) number N of mixture components. More formally we say a density $f(x)$ is an N -component mixture if it takes the form

$$f(x|\psi, \lambda) = \sum_{c=1}^N \psi_c f_c(x|\lambda_c) \quad (3.1)$$

where $f_1(x), \dots, f_N(x)$ are component densities and the ψ_c are *mixture weights* for each component. Each component density is also parameterised by a *unique* value λ_c . In order for this density to be well defined the following constraints must hold: (a) the component densities $f_c(x|\lambda_c)$ must all be valid density functions, that is, we require $f_c(x|\lambda_c) \geq 0$ for all x and $\int f_c(x|\lambda_c) dx = 1$ for $c = 1, \dots, N$, and (b) the mixture weights ψ_c must lie on the $(N - 1)$ -dimensional simplex, that is, $\psi_c \geq 0$ for $c = 1, \dots, N$ and $\sum_c \psi_c = 1$. Assuming these conditions hold this mixture distribution is defined for any choice of component densities be they continuous or discrete. In practice however these densities are often chosen from the same family.

If we have n observations, denoted $\mathbf{x} = (x_1, \dots, x_n)$, the (observed data) likelihood is

$$\pi(\mathbf{x}|\psi, \lambda) = \prod_{i=1}^n \left\{ \sum_{c=1}^N \psi_c f_c(x_i|\lambda_c) \right\}, \quad (3.2)$$

which, in general, is very complicated. It is however possible to make the form of the likelihood substantially more straightforward by appealing to data augmentation methods – specifically by introducing latent component/cluster indicator variables which we now discuss.

A common approach when implementing mixture models is to introduce latent cluster indicator variables, here denoted $\mathbf{c} = (c_1, \dots, c_n)$, where $c_i = j$ denotes that observation i belongs to component/cluster j . Conditional on the latent cluster indicator variables, the model is simplified significantly as the conditional density for observation x_i is simply $f_{c_i}(x_i|\lambda_{c_i})$. These random (unobserved) variables follow a *categorical* distribution defined as $\Pr(c_i = c) = \psi_c$ for $i = 1, \dots, n, c = 1, \dots, N$ and denoted $c_i|\psi \sim \text{Cat}(\psi)$. Therefore the joint (complete data) likelihood of the data \mathbf{x} and the latent cluster indicator variables \mathbf{c}

is

$$\pi(\mathbf{x}, \mathbf{c} | \boldsymbol{\psi}, \boldsymbol{\lambda}) = \prod_{i=1}^n \psi_{ci} f_{ci}(x_i | \lambda_{ci}),$$

as, given the parameters $\boldsymbol{\lambda}, \boldsymbol{\psi}$, the pairs (x_i, c_i) are independent. This form of the likelihood is substantially more straightforward than (3.2) and is the reason latent cluster indicators are typically introduced when fitting mixture models.

It follows that Bayesian implementations of finite mixture models, given the latent indicators, are generally of the form

$$\begin{aligned} X_i | \boldsymbol{\lambda}, c_i, \boldsymbol{\psi} &\sim f_{ci}(x_i | \lambda_{ci}) \\ c_i | \boldsymbol{\psi} &\sim \text{Cat}(\boldsymbol{\psi}) \\ \boldsymbol{\psi} &\sim \text{Dir}(\boldsymbol{\alpha}) \end{aligned} \tag{3.3}$$

where $i = 1, \dots, n$, and $\text{Dir}(\boldsymbol{\alpha})$ denotes the Dirichlet distribution with concentration parameters $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_N)$ where $\alpha_i > 0$. Note that in the model definition above the mixture components (and the latent cluster indicators) are exchangeable, that is, they can be arbitrarily relabelled while maintaining the equivalent model specification (Stephens, 2000). Therefore, within an inference context, it is perhaps not sensible to favour a particular mixture component *a priori*. This is achieved by choosing the concentration parameters to be $\alpha_i = \alpha = 1$ which gives $\boldsymbol{\psi} \sim \text{Dir}(\mathbf{1})$, that is, the mixture component weights $\boldsymbol{\psi}$ follow a uniform distribution over the $(N - 1)$ -dimensional simplex.

Naturally we could choose to form an N -component mixture of Plackett–Luce models by letting the component distributions be of Plackett–Luce form, with $X_i | \Lambda, c_i \sim \text{PL}(\boldsymbol{\lambda}_{ci})$ where $\boldsymbol{\lambda}_c = (\lambda_{c1}, \dots, \lambda_{cK})$ is the parameter vector associated with component (cluster) c and $\Lambda = \{\boldsymbol{\lambda}_c\}_{c=1}^N$ is the collection of all such parameter vectors. Indeed Gormley and Murphy (2008a,b, 2009) and Mollica and Tardella (2014) propose finite mixtures of Plackett–Luce and related models to allow for differing preferences between rankers. This approach was also taken by Vitelli et al. (2018) but instead they chose a distance based model, namely that of Mallows (1957). Of course, this approach could be trivially extended to form an N -component mixture of Weighted Plackett–Luce models by letting $X_i | \Lambda, c_i \sim \text{PLW}(\boldsymbol{\lambda}_{ci}, \mathbf{w})$. Under this setting the ranker weights \mathbf{w} would be common across all components with only the parameter vector $\boldsymbol{\lambda}$ being cluster specific. In some sense the models described in Chapter 2 could be considered to be a trivial case within the (finite) mixture model framework, with $N = 1$ mixture components, that is, a single homogeneous subgroup which contains the entire population of rankers.

Although finite mixture models give the flexibility to model heterogeneous data, specifying an appropriate form of such a model is a non-trivial task. One of the main issues that

arises when fitting finite mixture models is the constraint that a fixed number of mixture components must be chosen *a priori*. This requires the analyst to decide how many subgroups are contained within a population before performing their analysis. In an attempt to overcome this issue many choose instead to fit numerous models, each with differing numbers of components, and then appeal to model selection techniques (such as Akaike information criterion (AIC) or Bayesian information criterion (BIC)) to determine which model best fits the data. This solution however comes at the cost of performing numerous analyses. The analyst is still also required to choose the (different) number of components to consider. Ideally the mixture model would be defined so that the number of components is not fixed *a priori* and instead allows the number of components to be inferred using, for example, reversible jump methods (Richardson and Green, 1997). Alternatively we can appeal to a more flexible class of models, namely *infinite mixture models*. As the name suggests infinite mixture models contain an “infinite” number of components and thus the underlying density $f(x|\psi, \lambda)$ can be thought of as the limiting case as $N \rightarrow \infty$ of a finite mixture (3.1). Note that an “infinite” number of components only exists in theory and in practice the number of non-empty components can be at most the number of observations. Given the form of a finite mixture model (3.3) it is clear that we require an infinite dimensional Dirichlet distribution in order to define an infinite mixture model. The generalised version (to infinite dimension) of the Dirichlet distribution is the *Dirichlet process* – this the topic of the next section.

3.3 The Dirichlet process

In the previous section we discussed the need to increase modelling flexibility and relax the requirement of a fixed number of components *a priori* by appealing to infinite mixture models. Such models allow for full generality and furthermore allow the number of mixture components to be inferred from the data. By their nature, infinite mixture models induce an infinite dimensional parameter space and thus fall within the area of Bayesian non-parametrics (Hjort et al., 2010).

The Dirichlet process is a conjugate prior for infinite dimensional categorical distributions – a generalisation (to infinite dimension) of the result that the Dirichlet distribution is a conjugate prior for the categorical (multinomial) distribution. We now provide an overview and describe the common representations of the Dirichlet process before considering infinite mixture models. The overview provided here is somewhat brief and we refer the reader to either Ferguson (1973) and Antoniak (1974) or the more recent book by Hjort et al. (2010) for further details on the underlying measure theory for Dirichlet processes.

We use the notation $G \sim \text{DP}(\alpha, G_0)$ to denote that a distribution G follows a Dirichlet

process, where α and G_0 denote the concentration parameter and base distribution respectively. As the name suggests G_0 is a distribution itself and can be a continuous or discrete distribution. A realisation from a Dirichlet process however is almost surely a *discrete* distribution regardless of the form of G_0 . The realisations (distributions) are drawn from around the base distribution in a conceptually similar way to how realisations from the Normal distribution are drawn from around the mean. Further the expectation of a Dirichlet process is also the base distribution, that is, $E(G) = G_0$. The concentration parameter controls the deviation of realisations from the base distribution, and in that sense behaves similarly to a (inverse) standard deviation, and so, not surprisingly, $G \rightarrow G_0$ as $\alpha \rightarrow \infty$, that is, realisations get increasingly similar to the base distribution as the concentration parameter increases. As $\alpha \rightarrow 0$, the realisations G are discrete distributions concentrated at a single point mass. It follows that the probability of two distinct components (of G) being equal, $\Pr(\lambda_i = \lambda_j)$ for $i \neq j$, tends to 0 as $\alpha \rightarrow \infty$ (assuming G_0 is continuous) whereas $\Pr(\lambda_i = \lambda_j) \rightarrow 1$ as $\alpha \rightarrow 0$. Figure 3.1 illustrates this by depicting (on each row) three independent realisations from a Dirichlet process with $\alpha = 1, 5, 10, 50, 100$ from top to bottom respectively. The base distribution chosen in this case is $G_0 = N(0, 1)$. We observe that for the smaller values of α our (discrete distribution) realisations G are defined over fewer atoms with some having large mass. It follows that (theoretically) if $\alpha = 0$ we would have a single point mass at some value $\lambda \in \mathbb{R}$. For larger values of α we observe that the realisations (distributions) have increasingly more unique atoms, each having relatively small weight. In theory if $\alpha = \infty$ then this distribution would be defined over infinitely many atoms each of which has mass 0, that is, the distribution would be $G_0 = N(0, 1)$. The convergence of G to the base distribution as α increases is perhaps more easily seen through Figure 3.2 which shows the empirical cumulative distribution function (CDF) for each of the respective realisations shown within Figure 3.1. It is clear that as α increases, the CDF of these realisations becomes more like that of a $N(0, 1)$ distribution, that is, $G \rightarrow G_0$ as $\alpha \rightarrow \infty$.

As the Dirichlet process is a *distribution over distributions*, its structure is somewhat difficult to visualise. Sethuraman (1994) has shown that each Dirichlet process has a corresponding stick-breaking representation: writing $G \sim DP(\alpha, G_0)$ is equivalent to

$$\begin{aligned}
 G(\cdot) &= \sum_{j=1}^{\infty} \psi_j \delta_{\lambda_j}(\cdot) \\
 \psi_j &= v_j \prod_{\ell < j} (1 - v_\ell) \\
 v_j &\stackrel{\text{iid}}{\sim} \text{Beta}(1, \alpha) \\
 \lambda_j &\stackrel{\text{iid}}{\sim} G_0
 \end{aligned} \tag{3.4}$$

where $\delta_x(\cdot)$ denotes the Dirac probability measure concentrated at x and $j \in \mathbb{N}$. The construction of the weights dictates that $\sum_j \psi_j = 1$ and so it follows that the stick-breaking representation defines a discrete distribution G with *atoms* λ_j that have respective probabilities (*weights*) ψ_j . We also note that the weights are stochastically decreasing, that is, $E(\psi_j) \rightarrow 0$ as $j \rightarrow \infty$.

Although the stick-breaking representation provides the most intuitive insight in to how a Dirichlet process is defined, there are alternative representations. Within the next section we consider two common alternatives to the stick-breaking representation – the Chinese restaurant process and the (related) Pólya urn scheme.

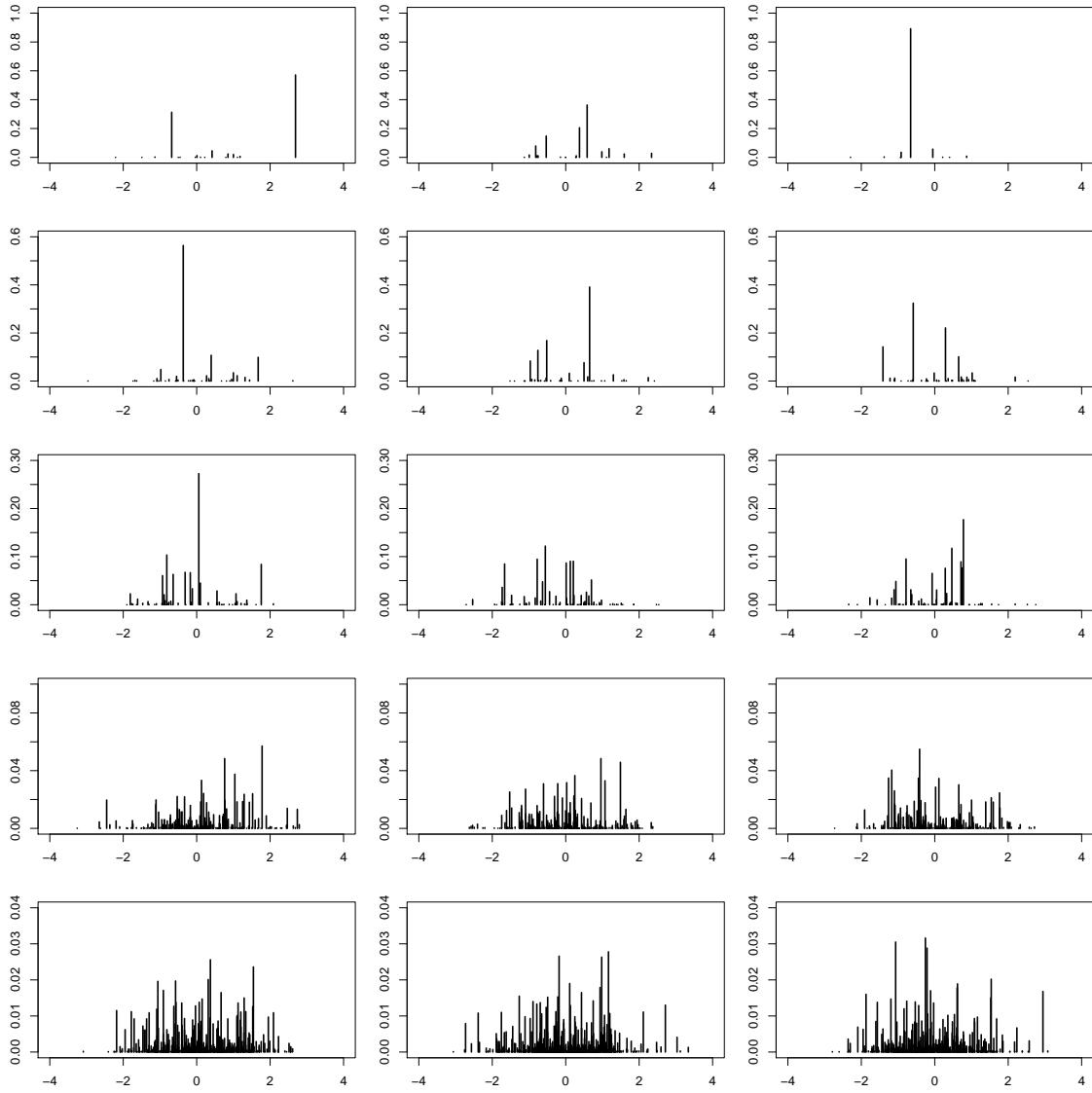


Figure 3.1: Multiple realisations from a Dirichlet process with $G_0 = N(0, 1)$ and $\alpha = 1, 5, 10, 50, 100$ from top to bottom respectively.

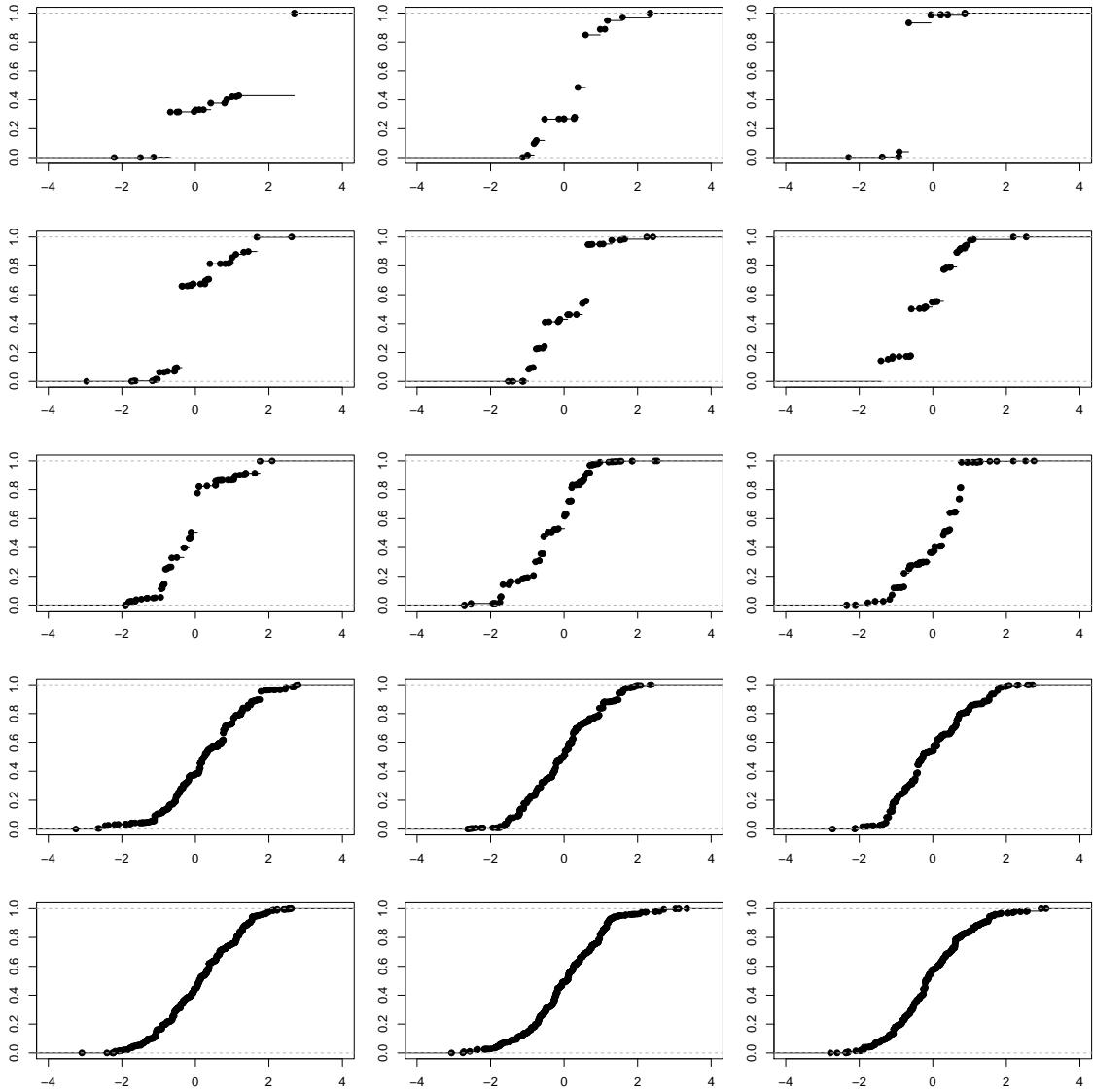


Figure 3.2: Empirical CDF from multiple realisations from a Dirichlet process with $G_0 = N(0, 1)$ and $\alpha = 1, 5, 10, 50, 100$ from top to bottom respectively.

3.3.1 Alternative representations of the Dirichlet process

Chinese restaurant process representation

An alternative way of visualizing a Dirichlet process is via the Chinese restaurant process (Aldous, 1985). This analogy, attributed to Jim Pitman and Lester Dubins, describes the distribution over the cluster allocations (more formally the distribution over partitions) induced by the Dirichlet process and proceeds as follows. Suppose there is a Chinese restaurant which contains an infinite number of tables, each of which have infinite seating

capacity. The first customer to enter the restaurant sits at table number 1. The next customer then has a choice: (a) they sit at an occupied table with probability proportional to the number of people currently at that table or (b) they sit at a new (unoccupied) table with probability proportional to α . This process continues until all n customers have been seated. At this point $N^c \leq n$ tables will be occupied, and the individuals at each table are interpreted as being clustered together. It follows that N^c is the number of unique clusters. From this metaphor it should be clear that a customer has higher probability of sitting at a table with a large number of customers rather than one with only a few customers. This is a feature of the Dirichlet process which is often summed up by the phrase “*the rich get richer*”. Drawing independent realisations from G_0 and assigning a value to each table results in a discrete distribution with probabilities proportional to the number of people seated at each respective table. This process is exchangeable, that is to say, the order in which the n customers arrive does not affect the final probability distribution.

Pólya urn representation

Another way to visualize a Dirichlet process (and the associated Chinese restaurant process) is via a Pólya urn scheme (Blackwell and MacQueen, 1973). For this analogy it is useful to consider $\alpha \in \mathbb{N}$ although the probabilities of observations being assigned to each cluster (to be defined) hold for any $\alpha \in \mathbb{R}_{>0}$. Suppose we have an urn filled with α white balls. A realisation from the Dirichlet process is obtained by repeatedly drawing balls from the urn subject to the following rules. If the ball drawn is white we generate an additional *uniquely* coloured ball before proceeding to place both the white ball we selected and the new coloured ball back into the urn. If the ball drawn is *not* white we generate an additional ball of the same colour as the one drawn and return both back into the urn. This process continues until n balls have been drawn. Once the n th ball has been drawn (and the appropriate action taken) the white balls are discarded from the urn. At this point there will be $N^c \leq n$ uniquely coloured balls within the urn and the distribution over these colours is equivalent to the distribution over the tables within the Chinese restaurant process. Drawing independent realisations from G_0 and assigning a value to each unique colour results in a discrete distribution with probabilities proportional to the number of each coloured ball. Again this process is also exchangeable, that is, if n people each select one ball the order in which the people are arranged does not affect the final probability distribution.

The probability of assigning the i th person/ball to table/ball colour j under both of these alternative representations is

$$\Pr(c_i = j | c_1, \dots, c_{i-1}) = \frac{n_{ij}^c}{\alpha + i - 1}, \quad \text{for } j = 1, \dots, N^c,$$

$$\Pr(c_i = N^c + 1 | c_1, \dots, c_{i-1}) = \frac{\alpha}{\alpha + i - 1},$$

where n_{ij}^c denotes the current number of observations assigned to cluster j (at iteration i), and N^c is the number of unique clusters that currently exist ($N^c = 0$ when $i = 1$).

3.3.2 Generating a realisation of a Dirichlet process

In this section we describe how to obtain realisations of a Dirichlet process, that is, how to obtain a realisation of the discrete distribution G where $G|\alpha, G_0 \sim \text{DP}(\alpha, G_0)$. The stick-breaking representation, although straightforward, does come with inherent problems. In practice it is not possible to sample an infinite number of weights for each of the corresponding atoms to define G as in (3.4). Unfortunately appealing to alternative representations of the Dirichlet process such as the Chinese restaurant/Pólya urn schemes yields similar issues. These processes are designed to allow for direct sampling from G , as opposed to generating a realisation of G itself. Perhaps, given a particular realisation of cluster allocations \mathbf{c} and corresponding cluster parameters $(\lambda_j^\dagger \stackrel{\text{indep}}{\sim} G_0 \text{ for } j = 1, \dots, N^c)$ from G , we might think that a realisation of the discrete distribution G is one with atoms λ_j^\dagger whose weights are proportional to the number of observations within cluster j , that is, with $\Pr(\lambda = \lambda_j^\dagger) \propto \#(c_i = j)$ for $i = 1, \dots, n$ and $j = 1, \dots, N^c$. However, this is not a true realisation of G as such a distribution is only defined over those atoms which are currently assigned to one of the n observations. The remaining infinite number of atoms (and corresponding weights) remain undefined until an observation is assigned to them. Given this, a true realisation of G can only be generated using this method if we consider the limit as $n \rightarrow \infty$, which is clearly infeasible. With no obvious solution to this problem we instead return our focus to the stick-breaking representation. The main issue here is how to sample the infinite number of weights for each of the corresponding atoms so that we can define G as in (3.4). This is made feasible if we choose a suitable truncation parameter $N_1 < \infty$ so that the distribution

$$G^* = \sum_{j=1}^{N_1} \psi_j \delta_{\lambda_j}(\cdot)$$

is a reasonable approximation to G so that then we need only sample N_1 weights (and atoms). Clearly $G^* \xrightarrow{d} G$ as $N_1 \rightarrow \infty$ and thus the level of approximation decreases

as N_1 increases. This truncation parameter needs to be chosen so that $\sum_{j>N_1} \psi_j \simeq 0$, an equivalent constraint to $\sum_{j \leq N_1} \psi_j \simeq 1$. Ishwaran and Zarepour (2002) describe how one might choose a suitable truncation parameter. We note however that this method of sampling can become infeasible for large α . Recall that the weights are defined as

$$\psi_j = v_j \prod_{\ell < j} (1 - v_\ell), \quad \text{where} \quad v_j \stackrel{\text{indep}}{\sim} \text{Beta}(1, \alpha),$$

and so for increasing α we have that $\psi_j \simeq 0$ for increasingly large values of j ; this follows from the result that $E(v_j) \rightarrow 0$ as $\alpha \rightarrow \infty$. Therefore, in the situation where α is large it may become infeasible to choose a suitably large truncation parameter N_1 so the constraint of $\sum_{j \leq N_1} \psi_j \simeq 1$ is satisfied. However, in practice it is often possible to choose a suitable value of N_1 so that $G^* \stackrel{d}{\simeq} G$ even for modestly large values of α .

An approximate realisation of G is obtained by using an appropriate truncation of the stick-breaking representation as follows:

- Choose N_1 sufficiently large.
- Choose $\alpha > 0$ or simulate from a suitable distribution.
- Simulate $\lambda_j^\dagger \stackrel{\text{indep}}{\sim} G_0$ for $j = 1, \dots, N_1$.
- Simulate $v_j \stackrel{\text{indep}}{\sim} \text{Beta}(1, \alpha)$ for $j = 1, \dots, N_1$.
- Set $\psi_j = v_j \prod_{\ell < j} (1 - v_\ell)$ for $j = 1, \dots, N_1$.

The discrete distribution (realisation of G) is that defined over atoms λ_j^\dagger with weights ψ_j , that is, $\Pr(\lambda = \lambda_j^\dagger) \propto \psi_j$ for $j = 1, \dots, N_1$.

Note that G^* only defines a mixture distribution if the weights ψ sum to one; see Section 3.2. However, the weights above only satisfy this in the limiting case when $N_1 = \infty$. Of course we could simply rescale these weights so that $\sum_{j=1}^{N_1} \psi_j = 1$ however it is often advantageous, particularly within an inference context, to construct the weights so that they sum to one irrespective of the choice of (finite) truncation parameter N_1 . This can be achieved by simulating $v_j \stackrel{\text{indep}}{\sim} \text{Beta}(1, \alpha)$ for $j = 1, \dots, N_1 - 1$ and fixing $v_{N_1} = 1$. The weights are then constructed in the usual manner however in this setting we have $\psi_{N_1} = \prod_{\ell < j} (1 - v_\ell)$ and so ψ_{N_1} contains all the remaining mass and this ensures that $\sum_{j=1}^{N_1} \psi_j = 1$. It follows that, in this case, G^* is a valid (discrete) mixture distribution defined by the probabilities $\Pr(\lambda = \lambda_j^\dagger) = \psi_j$ for $j = 1, \dots, N_1$.

3.3.3 Generating model parameters consistent with a Dirichlet process prior

In the previous section we considered how to obtain realisations of G where $G|\alpha, G_0 \sim \text{DP}(\alpha, G_0)$. We now suppose that the data are n observations and denote the parameter for the i th observation by λ_i . The focus of this section is how to obtain a realisation of $\boldsymbol{\lambda} = (\lambda_1, \dots, \lambda_n)$ given $\boldsymbol{\lambda}|G \sim G$, that is, how to draw realisations (of the parameters) *from* G , where G follows a Dirichlet process. Such realisations are reasonably straightforward to obtain under the stick-breaking representation. However, any realisation obtained using this method will only be from an approximation to the true distribution defined by the Dirichlet process. This approximation is due to the need to truncate G so that it is finite dimensional (see above). On the other hand, the Chinese restaurant/Pólya urn representations allow for *exact* realisations to be drawn directly from G (we need not obtain a realisation of G explicitly). Further, these samples remain from the true distribution defined by the Dirichlet process irrespective of the value of α .

To generate a realisation of $\boldsymbol{\lambda} = (\lambda_1, \dots, \lambda_n)$ we appeal to the latent cluster indicators \mathbf{c} , where $c_i = c$ denotes that observation i is within cluster c . Note that, given the (unique) cluster parameters λ_j^\dagger , the cluster indicators enable us to completely identify $\boldsymbol{\lambda}$ and so obtaining a realisation for the cluster indicators is equivalent to drawing a realisation for $\boldsymbol{\lambda}$. We now describe how to obtain realisations of \mathbf{c} (and therefore $\boldsymbol{\lambda}$) which are consistent with the Dirichlet process prior under both the stick-breaking and the Chinese restaurant/Pólya urn representations.

Stick-breaking representation

To generate a realisation of the cluster allocations consistent with a Dirichlet process prior using the stick-breaking representation we must first obtain a (approximate) realisation of the discrete distribution G . This can be obtained using the method described in the previous section which gives a realisation of G defined by $\Pr(\lambda = \lambda_j^\dagger) = \psi_j$ for $j = 1, \dots, N_1$. Given a realisation of G , we can generate a realisation of the cluster allocations for n observations as follows.

- Sample $c_i \stackrel{\text{iid}}{\sim} \text{Cat}(N_1, \boldsymbol{\psi})$ for $i = 1, \dots, n$.

The parameter vector $\boldsymbol{\lambda}$ is then given by $\lambda_i = \lambda_{c_i}^\dagger$ for $i = 1, \dots, n$.

Chinese restaurant/Pólya urn representation

Under the stick-breaking representation we need to first obtain a realisation of (the distribution) G before then drawing samples of the cluster allocations (from G). However, under the Chinese restaurant/Pólya urn representation this step is no longer needed and we can instead draw realisations (of cluster allocations \mathbf{c} and therefore $\boldsymbol{\lambda}$) directly from G using the following process:

- Choose $\alpha > 0$ or simulate from a suitable distribution.
- Set $c_1 = 1$ and the (current) number of clusters as $N^c = 1$.
- For $i = 2, \dots, n$ simulate the allocation of observation i to a cluster according to the discrete distribution

$$\Pr(c_i = j | c_1, \dots, c_{i-1}) = \frac{n_{ij}^c}{\alpha + i - 1}, \quad \text{for } j = 1, \dots, N^c,$$

$$\Pr(c_i = N^c + 1 | c_1, \dots, c_{i-1}) = \frac{\alpha}{\alpha + i - 1},$$

where n_{ij}^c denotes the number of points currently within cluster j (at iteration i), and $N^c \rightarrow N^c + 1$ if $c_i = N^c + 1$.

- Simulate $\lambda_j^\dagger \stackrel{\text{indep}}{\sim} G_0$ for $j = 1, \dots, N^c$.

Again, as for the stick-breaking representation, the parameter associated with observation i is given by $\lambda_{c_i}^\dagger$. It follows that the parameter vector $\boldsymbol{\lambda}$ is given by $\lambda_i = \lambda_{c_i}^\dagger$ for $i = 1, \dots, n$.

We now highlight a subtle but important difference between the two methods. Suppose we are in the scenario where $n \ll N_1$, that is, the number of observations is significantly smaller than the truncation parameter. In this case the error resulting from the approximation used in the stick-breaking approach will be reasonably small (conditional on a suitable choice of concentration parameter). However, in the stick-breaking approach, the realisation of G is defined over a (fixed) finite number of atoms (N_1), which, as a result, constrains the maximum number of clusters to N_1 . In other words, irrespective of the number of observations n , the parameter vector $\boldsymbol{\lambda}$ can contain at most N_1 unique values. It follows that this method of generating parameter realisations may result in a poor approximation to the true distribution defined by the Dirichlet process in the limit as $n \rightarrow \infty$. However, the Chinese restaurant/Pólya urn method allows for the possibility that each of the i observations can join a new cluster and is therefore assigned to a (unique) parameter which is an independent draw from the base distribution. It follows that in this case the upper limit on the number of clusters is theoretically infinite when considering the limit as $n \rightarrow \infty$.

3.3.4 Generalised Dirichlet process

The Pitman–Yor process is a generalised version of the Dirichlet process. This process is accredited to Pitman and Yor (1997) for their work on the two-parameter Poisson–Dirichlet distribution. However the name was coined by Ishwaran and James (2001) in their review of stick-breaking priors. Here we let $\text{PY}(\alpha, d, G_0)$ denote the Pitman–Yor process with governing parameters $\alpha (> -d)$, known as the *strength parameter*, a *discount parameter* $0 \leq d < 1$, and a base distribution G_0 . As for the Dirichlet process, a realisation from the Pitman–Yor process is a discrete distribution over an infinite set of atoms; also these atoms are (independent) draws from the base distribution G_0 . However, in contrast to the Dirichlet process, the weight (probability) associated with each atom is drawn from a two-parameter Poisson–Dirichlet distribution. This results in the Pitman–Yor process being more flexible than the Dirichlet process with regards to tail behaviour and it is often the preferred model for analysing data with power-law tails (the Dirichlet process has exponential tails).

To visualise the relationship between the Pitman–Yor and the Dirichlet processes, consider the stick-breaking representation of the former

$$\begin{aligned} G(\cdot) &= \sum_{j=1}^{\infty} \psi_j \delta_{\lambda_j}(\cdot) \\ \psi_j &= v_j \prod_{\ell < j} (1 - v_\ell) \\ v_j &\stackrel{\text{iid}}{\sim} \text{Beta}(1 - d, \alpha + jd) \\ \lambda_j &\stackrel{\text{iid}}{\sim} G_0. \end{aligned} \tag{3.5}$$

Clearly the case $d = 0$, produces a distribution G from (3.5) that is equivalent to that from the Dirichlet process (3.4), that is, $\text{PY}(\alpha, 0, G_0) \stackrel{d}{=} \text{DP}(\alpha, G_0)$. For this reason the Dirichlet process is considered to be a special case of the Pitman–Yor process. The Normalised Inverse–Gamma process is another special case given by $d = 0.5$ and $\alpha = 0$.

We need $\sum_j \psi_j = 1$ for G to be well defined, or equivalently, the atom weights must be on the simplex. If we let $a = 1 - d$ and $b_j = \alpha + jd$, Lemma 1 of Ishwaran and James (2001) shows that

$$\sum_{j=1}^{\infty} \psi_j = 1 \text{ almost surely} \iff \sum_{j=1}^{\infty} \log \left(1 + \frac{a}{b_j} \right) = \infty. \tag{3.6}$$

It is trivial to verify that condition (3.6) holds for the Dirichlet process. Recall that the Dirichlet process is a special case of the Pitman–Yor process with $d = 0$, and hence $a = 1$

and $b_j = \alpha$. Given this we have

$$\begin{aligned} \alpha > 0 &\implies 1 + \frac{1}{\alpha} > 1 \\ &\implies \log\left(1 + \frac{1}{\alpha}\right) > 0 \\ &\implies \sum_{j=1}^{\infty} \log\left(1 + \frac{1}{\alpha}\right) = \infty \\ &\implies \sum_{j=1}^{\infty} \psi_j = 1 \text{ almost surely} \end{aligned}$$

and so the distribution in (3.4) is well defined.

In what follows we focus on the Dirichlet process and note that this is a common choice of stick-breaking prior; primarily due to the availability of efficient sampling schemes. Many of these (efficient) inference schemes make use of the Chinese restaurant process representation. Unfortunately, such representations are typically unavailable for the Pitman–Yor process and so the stick-breaking representation must be used and, under this representation, it is only possible to obtain an approximate posterior distribution (although the approximation can be made arbitrarily small given sufficient computing power) – this is discussed further in Section 3.4.1.

3.4 Dirichlet process mixture models

The development of the Dirichlet process mixture model (DPMM) is accredited to Ferguson (1973) and Antoniak (1974). Since their conception, DPMMs have become popular within the Bayesian literature as they allow for both complex and flexible models to be constructed with relative ease. A typical Dirichlet process mixture model is a mixture of a distribution F over its parameters. For example

$$\begin{aligned} x_i | \lambda_i &\sim F(\lambda_i), \\ \lambda_i | G &\sim G, \\ G | \alpha, G_0 &\sim \text{DP}(\alpha, G_0), \end{aligned}$$

where DP denotes a Dirichlet process (formally defined by the stick-breaking representation in (3.4)) with concentration parameter α and base distribution G_0 .

3.4.1 Bayesian inference for DPMM

There are numerous ways to perform Bayesian inference for Dirichlet process mixture models. The majority of methods can be classified as taking either a conditional or a marginal approach, as summarised in, for example, Papaspiliopoulos and Roberts (2008). The conditional approaches typically use truncation in order to approximate the infinite-dimensional aspect of the stick-breaking prior, as pioneered by Ishwaran and James (2001). However, avoiding approximations is beneficial and the slice and retrospective samplers of Walker (2007) and Papaspiliopoulos and Roberts (2008) provide methods for achieving this. Unfortunately, these methods can suffer from poor mixing and convergence as they attempt to sample multimodal posterior distributions. One solution is the addition of appropriate label switching moves (Hastie et al. (2015), Papaspiliopoulos and Roberts (2008)) though, in general, further empirical work is needed to determine the number and types of moves that give an effective solution.

For these reasons we avoid conditional methods here and instead implement a marginal sampler. These samplers typically involve a Pólya urn scheme and marginalise over the infinite dimensional distribution (Escobar and West (1995), MacEachern and Müller (1998)), and thereby avoid the need for approximation. Algorithm 8 of Neal (2000), hereafter referred to as Neal’s Algorithm 8, is one such sampler. This algorithm has been shown to be one of the most efficient sampling methods for Dirichlet Process mixtures; see, for example, Papaspiliopoulos and Roberts (2008). Also there is no need for additional label switching moves. Efficiency is achieved by the algorithm only performing updates for the unique components which are currently assigned to an observation. Each observation is then assigned to either: (a) a component which is currently in use (“active”) or (b) to one of m *auxiliary* components whose parameters are (independent) draws from the base distribution. The number of auxiliary components, m , is chosen by the analyst. We note that this places little burden on the analyst as the choice of m is made solely for efficiency purposes – the equilibrium distribution of the Markov chain remains exact for all choices of $m \geq 1$ (Neal, 2000). From our experience we have found that taking $m = 2$ or 3 auxiliary components typically produces a well mixing Markov chain. Generally the mixing improves as the number of auxiliary components increases due to the observations having more opportunity to join an alternative cluster. Increasing m however does come with additional computational cost as m (independent) draws are needed from the base distribution for *each* observation at every iteration of our algorithm. Also the discrete (full conditional) distribution over the cluster allocations will also increase in dimension.

Neal’s Algorithm 8 is closely related to other (marginal) sampling schemes within the literature. When $m = 1$, Neal’s Algorithm 8 closely resembles the “no gaps” algorithm of MacEachern and Müller (1998). However, the probability that an observation moves from

its current (active) cluster into an auxiliary cluster is larger under Neal’s Algorithm 8. At first view this may not appear to be helpful given that the aim is to cluster observations together. However, the Dirichlet process has the unfortunate property of “masking” small clusters, that is, the penalty for creating an additional cluster to house a few observations is larger than the penalty for placing them in a current active cluster – even if these observations are somewhat different from the existing group. Clearly observations should be encouraged to form new clusters (join auxiliary ones) whenever they differ sufficiently from the existing clusters.

3.4.2 Allowing for uncertainty on the concentration parameter

Central to the implementation of a Dirichlet process mixture model is the choice of concentration parameter α . The choice of α results in an implied prior on the number of clusters or unique values (N^c). Antoniak (1974) provides an implicit form of the conditional prior distribution, $\pi(N^c|\alpha, n)$, when we have n observations; see Section A.1 within the appendices for full details.

Choosing a value of α is somewhat difficult unless we have substantial prior knowledge of the number of subgroups within the data. It can therefore be helpful to express (uncertain) beliefs about α in terms of a prior distribution and thereby infer its posterior distribution. Escobar and West (1995) and West (1992) show that when using a marginal sampling scheme and a finite mixture of Gamma distributions as a prior for α , it is possible to derive a closed form full conditional distribution for α . It is fairly straightforward to simulate from this full conditional distribution and so this can be incorporated within a Gibbs sampling scheme.

In the simplest case where α has a (single component mixture of) Gamma distribution, that is, $\alpha \sim \text{Ga}(a_\alpha, b_\alpha)$, the full conditional distribution is the two component mixture

$$\alpha | \dots \sim \pi \text{Ga}(a_\alpha + N^c, b_\alpha - \log \eta) + (1 - \pi) \text{Ga}(a_\alpha + N^c - 1, b_\alpha - \log \eta),$$

where the mixture weights are given by

$$\frac{\pi}{(1 - \pi)} = \frac{a_\alpha + N^c - 1}{n(b_\alpha - \log \eta)},$$

and

$$\eta | \dots \sim \text{Beta}(\alpha + 1, n).$$

Here η is a latent random variable which facilitates the conjugate update. This result is derived within Section A.1 of the appendices.

3.5 Uncovering heterogeneity between rankers

Until now our main aim has been to perform Bayesian inference and obtain a single preference ordering of entities that summarises a collection of rankings and is a form of rank aggregation. This aim is only appropriate if the rankers are homogeneous in terms of their beliefs about the entities, that is, each individual ranking (within a collection) follows the same underlying ranking distribution. Our current Weighted Plackett–Luce model does allow for heterogeneity between ranker abilities which in some sense does allow for different groups of rankers. However, this heterogeneity only allows for limited variability between informative and uninformative groups of rankers. This model is therefore inadequate to handle a scenario where several groups of rankers express alternative preferences regarding the entities that they are ranking.

In this chapter we have thus far introduced both finite and infinite mixture models and discussed how such models can be implemented to model heterogeneous data. We now appeal to these methods to build a model capable of handling heterogeneous ranked data. Within this section we suppose that there may be groups of rankers, each of which has their own beliefs about the preference of entities. Under this scenario we wish to build a model which allows each ranker group to have their own unique set of skill parameters which summarise their beliefs about the entities. Previously our models have considered only a single parameter vector which summarises all rankings, that is, we have only considered a single ranker group. To implement a grouping structure we now need a parameter vector associated with each ranker ($\boldsymbol{\lambda}_i$ for $i = 1, \dots, n$). These parameter vectors need not be unique and indeed we desire that rankers with the same beliefs about the entities also share the same parameter vector. Our model should therefore be built so that it allows $\Pr(\boldsymbol{\lambda}_i = \boldsymbol{\lambda}_j) \geq 0$ for all $i \neq j$. This structure can be achieved by implementing a Dirichlet process prior distribution where the atoms of the DP are (unique) parameter vectors $\boldsymbol{\lambda}$. In this scenario (for $\alpha < \infty$) the Dirichlet process prior specifies a discrete distribution over a range of parameter vectors, and therefore if we draw samples from this distribution we will have $\Pr(\boldsymbol{\lambda}_i = \boldsymbol{\lambda}_j) > 0$ for all $i \neq j$.

In what follows we describe a new model consisting of an infinite mixture of Weighted Plackett–Luce models with a Dirichlet process as the conjugate prior distribution. We are able to derive a complete set of full conditional distributions for each parameter of interest within our model by using latent variables. This allows us to form a Gibbs sampling algorithm in which we appeal to Neal’s Algorithm 8 (Neal, 2000) to sample the latent cluster indicator variables efficiently. We note that much of the literature concerning modelling heterogeneous ranked data has been limited to finite mixture models of standard Plackett–Luce models. However, here we allow for additional heterogeneity between rankers’ abilities by assuming the Weighted Plackett–Luce model is the true

underlying ranking distribution. We are also able to relax the assumption of a fixed number of components *a priori* by constructing the model as a Dirichlet process mixture model. The section is concluded with a simulation study.

3.5.1 The model

Suppose we have n rankers where each ranker reports positions for $n_i \leq K$ entities. The main components of our complete model, the Dirichlet process mixture of Weighted Plackett–Luce models (WDP), can be written as

$$\begin{aligned} \mathbf{X}_i | \boldsymbol{\lambda}_i, w_i &\stackrel{\text{iid}}{\sim} \text{PLW}(\boldsymbol{\lambda}_i, w_i) & i = 1, \dots, n, \\ \boldsymbol{\lambda}_i | G &\stackrel{\text{iid}}{\sim} G & i = 1, \dots, n, \\ G | \alpha, G_0 &\sim \text{DP}(\alpha, G_0). \end{aligned}$$

To make the form of the non-parametric prior distribution unambiguous we define the stick-breaking representation which here is

$$\begin{aligned} G(\cdot) &= \sum_{s=1}^{\infty} \psi_s \delta_{\boldsymbol{\lambda}_s}(\cdot) \\ \psi_s &= v_s \prod_{\ell < s} (1 - v_\ell) \\ v_s &\stackrel{\text{iid}}{\sim} \text{Beta}(1, \alpha) \\ \lambda_{sk} &\stackrel{\text{iid}}{\sim} G_0 \end{aligned}$$

for $s \in \mathbb{N}$ and $k = 1, \dots, K$. Note that under this model λ_{sk} now denotes the skill parameter for entity k in group/cluster s whereas previously the skill parameter of each entity was denoted λ_k (as we only had a single group of rankers). Furthermore there is no need to choose a single base distribution G_0 . Instead a unique base distribution can be chosen for each of the K skill parameters, that is, we can let $\lambda_{sk} \stackrel{\text{iid}}{\sim} G_{0k}$ in the stick-breaking representation above. That said, the choice of base distribution(s) must be exchangeable across cluster labels given the inherent exchangeability of components within the Dirichlet process prior, that is, G_{0k} *must* not depend on s .

3.5.2 Simulating data from the Dirichlet process mixture of Weighted Plackett–Luce models

In this section we describe how to simulate data from our Dirichlet process mixture of Weighted Plackett–Luce models. It is useful to introduce latent cluster indicator variables

when dealing with mixture models. Here we introduce $\mathbf{c}^r = (c_1^r, \dots, c_n^r)$ where $c_i^r = j$ denotes that ranker i is associated with parameter vector $\boldsymbol{\lambda}_j$. For example, if $\mathbf{c}^r = (1, 2, 1, 2)$ then rankers 1 and 3 are in cluster 1 and rankers 2 and 4 are in cluster 2; each cluster has a unique parameter vector, here $\boldsymbol{\lambda}_1$ and $\boldsymbol{\lambda}_2$. Furthermore we let $N^r = |\{c_i^r\}_{i=1}^n|$ denote the number of unique ranker clusters, that is, the number of unique parameter vectors, and let $\Lambda = (\boldsymbol{\lambda}_1, \dots, \boldsymbol{\lambda}_{N^r})$ denote the collection of these unique parameter vectors.

Given these latent cluster indicators we can now describe how to generate data under the WDP model outlined in the previous section. First we need to specify a ranker clustering structure and this can be achieved by either (a) explicitly defining values for the latent cluster allocation variables c_i^r and labelling these $1, \dots, N^r$, or (b) draw a realisation (marginally) from the Dirichlet process prior distribution as follows.

- Choose $\alpha > 0$ or simulate from a suitable distribution.
- Set $c_1^r = 1$ and the (current) number of clusters as $N^r = 1$.
- For $i = 2, \dots, n$ simulate the allocation of ranker i to a cluster according to the discrete distribution given by

$$\Pr(c_i^r = j | c_1^r, \dots, c_{i-1}^r) = \frac{n_{ij}^r}{\alpha + i - 1}, \quad \text{for } j = 1, \dots, N^r,$$

$$\Pr(c_i^r = N^r + 1 | c_1^r, \dots, c_{i-1}^r) = \frac{\alpha}{\alpha + i - 1},$$

where n_{ij}^r denotes the number of points currently within cluster j (at iteration i), and $N^r \rightarrow N^r + 1$ if $c_i^r = N^r + 1$.

Given a clustering structure of the rankers we now need to choose values for the (cluster specific) skill parameters λ_{sk} . Again these can either be chosen explicitly or alternatively can be drawn from the prior distribution by sampling $\lambda_{sk} \stackrel{\text{indep}}{\sim} G_{0k}$ for $s = 1, \dots, N^r$, $k = 1, \dots, K$. Recall that here we have a mixture of Weighted Plackett–Luce models and we therefore need to choose whether each ranking is to be informative or not, that is, choose (or sample) a value of $w_i \in \{0, 1\}$ for $i = 1, \dots, n$.

After the parameters of the model are fully specified, we can use the exponential latent variable representation of the Weighted Plackett–Luce model to generate rankings. A collection of n complete rankings $\{\mathbf{x}_i\}_{i=1}^n$ can be generated via the following process.

For $i = 1, \dots, n$

- Sample $\nu_{ij} \stackrel{\text{indep}}{\sim} \text{Exp}(\lambda_{c_i^r j}^{w_i})$ for $j = 1, \dots, K$.
- Set $x_{ij} = \underset{q \in S_{ij}}{\operatorname{argmin}} \nu_{iq}$ where $S_{ij} = \mathcal{K} \setminus \{x_{i1}, \dots, x_{ij-1}\}$ for $j = 1, \dots, K$.

Alternative types of rankings (such as a top–5 ranking) can be obtained from the complete rankings simulated using the same process as discussed in Section 2.2.5.

3.5.3 Prior specification and latent variables

When first implementing the standard (or Weighted) Plackett–Luce model in Chapter 2 we discussed how it is advantageous to use Gamma prior distributions on the skill parameters as this gives conjugate updates; see Section 2.3. In this model the equivalent prior specification is achieved by letting $G_{0k} = \text{Ga}(a_k, 1)$ which gives $\lambda_{sk} \stackrel{\text{indep}}{\sim} \text{Ga}(a_k, 1)$ for $s \in \mathbb{N}$ and $k = 1, \dots, K$. Our prior beliefs about the strength of entity k (relative to the other entities) is then expressed through the parameter a_k . Recall that the rate parameter is not likelihood identifiable and so we take this to be 1. The prior distribution on the latent binary ability indicators remains as before, with $w_i \stackrel{\text{indep}}{\sim} \text{Bern}(p_i)$ where $p_i \in (0, 1]$ for $i = 1, \dots, n$. We also wish to infer the DP concentration parameter from the data and so we need to specify a prior distribution: we take $\alpha \sim \text{Ga}(a_\alpha, b_\alpha)$.

Before we describe the (data augmentation based) posterior computation algorithm we need to define the latent cluster indicator variables. We use the cluster indicators introduced in the previous section, that is, $\mathbf{c}^r = (c_1^r, \dots, c_n^r)$ where $c_i^r = j$ denotes that ranking i is associated with parameter vector $\boldsymbol{\lambda}_j$. Recall that $N^r = |\{c_i^r\}_{i=1, \dots, n}|$ denotes the number of unique ranker clusters and $\Lambda = (\boldsymbol{\lambda}_1, \dots, \boldsymbol{\lambda}_{N^r})$ is the collection of the unique skill parameter vectors.

We are now in a position to define the prior distribution over the skill parameters in this model. The model contains N^r ranker clusters, each of which has an associated parameter vector $\boldsymbol{\lambda}$. The model contains $N^r \times K$ unique skill parameters, whence, conditional on the latent cluster parameters, the prior distribution of Λ is

$$\pi(\Lambda | \mathbf{c}^r) = \prod_{c=1}^{N^r} \prod_{k=1}^K \frac{\lambda_{ck}^{a_k-1} e^{-\lambda_{ck}}}{\Gamma(a_k)}.$$

The model assumes that each ranking follows the Weighted Plackett–Luce probability, that is, $\mathbf{X}_i | \Lambda, \mathbf{w}, \mathbf{c}^r \stackrel{\text{indep}}{\sim} \text{PLW}(\boldsymbol{\lambda}_{c_i^r}, w_i)$. Therefore the probability of the i th ranking can be expressed using the latent cluster indicator variables as

$$\Pr(\mathbf{X}_i = \mathbf{x}_i | \Lambda, \mathbf{w}, \mathbf{c}^r) = \prod_{j=1}^{n_i} \frac{\lambda_{c_i^r, x_{ij}}^{w_i}}{\sum_{m=j}^{n_i} \lambda_{c_i^r, x_{im}}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_i^r, m}^{w_i}},$$

and so, as the rankings are (conditionally) independent, the likelihood of all n rankings is

$$\pi(\mathcal{D}|\Lambda, \mathbf{w}, \mathbf{c}^r) = \prod_{i=1}^n \prod_{j=1}^{n_i} \frac{\lambda_{c_i^r, x_{ij}}^{w_i}}{\sum_{m=j}^{n_i} \lambda_{c_i^r, x_{im}}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_i^r, m}^{w_i}}.$$

As discussed in Chapter 2, the Plackett–Luce likelihood does not admit conjugate Bayesian inference. However, we saw that conjugate updates for the skill parameters can be achieved by augmenting the parameter space with appropriate latent variables. Here the latent variables are still defined in terms of the latent exponential inter-arrival times but now are based on cluster-specific skill parameters, that is,

$$z_{ij} | \mathcal{D}, \Lambda, \mathbf{w}, \mathbf{c}^r \stackrel{\text{indep}}{\sim} \text{Exp} \left(\sum_{m=j}^{n_i} \lambda_{c_i^r, x_{ij}}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_i^r, m}^{w_i} \right), \quad (3.7)$$

for $i = 1, \dots, n$ and $j = 1, \dots, n_i$.

3.5.4 Full conditional distributions

The posterior distribution is formed by applying Bayes' Theorem. The posterior distribution $\pi(Z, \Lambda, \mathbf{w}, \mathbf{c}^r, \alpha | \mathcal{D})$ is now a joint distribution of the latent random variables Z , the collection of unique skill parameters Λ , the binary indicator variables \mathbf{w} , the latent cluster indicators \mathbf{c}^r and the DP concentration parameter α . We can obtain realisations from the posterior distribution using a Gibbs sampling strategy which samples from the FCDs of each unknown quantity in turn. The latent cluster indicators can be drawn from their respective full conditional distributions using Neal's Algorithm 8 (Neal, 2000). Further, the FCD of the DP concentration parameter α is also known; see Section 3.4.2. In the remainder of this section we derive the FCDs for the remaining unknown quantities (Z, Λ, \mathbf{w}) and a complete outline of the MCMC scheme used to generate posterior samples can be found in Section 3.5.5.

Before starting the derivation of the full conditional distributions it is useful to first construct the density of all stochastic quantities. Note however that, given we already have the FCDs for \mathbf{c}^r and α , it is sensible to only consider the conditional density of all the remaining stochastic quantities given the latent cluster indicator variables and the DP concentration parameter. This (conditional) joint density is

$$\begin{aligned} \pi(\Lambda, \mathcal{D}, Z, \mathbf{w} | \mathbf{c}^r, \alpha) \\ = \pi(\Lambda, \mathcal{D}, Z, \mathbf{w} | \mathbf{c}^r) \\ = \pi(Z | \mathcal{D}, \Lambda, \mathbf{w}, \mathbf{c}^r) \pi(\mathcal{D} | \Lambda, \mathbf{w}, \mathbf{c}^r) \pi(\Lambda | \mathbf{c}^r) \pi(\mathbf{w}) \end{aligned}$$

$$\begin{aligned}
 &= \prod_{i=1}^n \prod_{j=1}^{n_i} \left(\sum_{m=j}^{n_i} \lambda_{c_i^r, x_{ij}}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_i^r, m}^{w_i} \right) \exp \left\{ - \left(\sum_{m=j}^{n_i} \lambda_{c_i^r, x_{ij}}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_i^r, m}^{w_i} \right) z_{ij} \right\} \\
 &\quad \times \prod_{i=1}^n \prod_{j=1}^{n_i} \lambda_{c_i^r, x_{ij}}^{w_i} \left(\sum_{m=j}^{n_i} \lambda_{c_i^r, x_{ij}}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_i^r, m}^{w_i} \right)^{-1} \times \prod_{c=1}^{N^r} \prod_{k=1}^K \frac{\lambda_{ck}^{a_k-1} e^{-\lambda_{ck}}}{\Gamma(a_k)} \times \prod_{i=1}^n p_i^{w_i} (1-p_i)^{1-w_i} \\
 &= \prod_{i=1}^n \prod_{j=1}^{n_i} \lambda_{c_i^r, x_{ij}}^{w_i} \exp \left\{ - \left(\sum_{m=j}^{n_i} \lambda_{c_i^r, x_{ij}}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_i^r, m}^{w_i} \right) z_{ij} \right\} \times \prod_{c=1}^{N^r} \prod_{k=1}^K \frac{\lambda_{ck}^{a_k-1} e^{-\lambda_{ck}}}{\Gamma(a_k)} \\
 &\quad \times \prod_{i=1}^n p_i^{w_i} (1-p_i)^{1-w_i}.
 \end{aligned} \tag{3.8}$$

The full conditional distributions (FCDs) can be obtained from this density by constructing the conditional distribution of each random quantity given all other quantities. The latent parameters $Z = \{z_{ij}\}$ are defined through their full conditional distribution and so it should come as no surprise that we obtain

$$\pi(Z|\mathcal{D}, \Lambda, \mathbf{w}, \mathbf{c}^r, \alpha) \propto \prod_{i=1}^n \prod_{j=1}^{n_i} \exp \left\{ - \left(\sum_{m=j}^{n_i} \lambda_{c_i^r, x_{ij}}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_i^r, m}^{w_i} \right) z_{ij} \right\},$$

and therefore the full conditional distributions for z_{ij} are as in (3.7) for $i = 1, \dots, n$, $j = 1, \dots, n_i$.

The full conditional distribution for the (cluster-specific) skill parameters λ_{ck} are derived as follows:

$$\begin{aligned}
 &\pi(\Lambda|\mathcal{D}, Z, \mathbf{w}, \mathbf{c}^r, \alpha) \\
 &\propto \prod_{i=1}^n \prod_{j=1}^{n_i} \lambda_{c_i^r, x_{ij}}^{w_i} \exp \left\{ - \left(\sum_{m=j}^{n_i} \lambda_{c_i^r, x_{ij}}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_i^r, m}^{w_i} \right) z_{ij} \right\} \times \prod_{c=1}^{N^r} \prod_{k=1}^K \frac{\lambda_{ck}^{a_k-1} e^{-\lambda_{ck}}}{\Gamma(a_k)} \\
 &\propto \prod_{c=1}^{N^r} \prod_{k=1}^K \lambda_{ck}^{a_k + \gamma_{ck} - 1} e^{-\lambda_{ck}} \times \prod_{i=1}^n \prod_{j=1}^{n_i} \exp \left\{ - \left(\sum_{m=j}^{n_i} \lambda_{c_i^r, x_{ij}}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_i^r, m}^{w_i} \right) z_{ij} \right\} \\
 &= \prod_{c=1}^{N^r} \prod_{k=1}^K \lambda_{ck}^{a_k + \gamma_{ck} - 1} \exp \left\{ - \left(1 + \sum_{i=1}^n w_i \sum_{j=1}^{n_i} \zeta_{ij}(c, k) z_{ij} \right) \lambda_{ck} \right\},
 \end{aligned}$$

where

$$\gamma_{ck} = \sum_{\ell=1}^n w_{\ell} \mathbb{I}(c_{\ell}^r = c) \mathbb{I}(k \in \{\mathbf{x}_{\ell}\})$$

is the number of informative rankings associated with cluster c in which entity k appears

and

$$\zeta_{ij}(c, k) = \mathbb{I}(c_i^r = c) \times \mathbb{I}(k \in \{x_{ij}, \dots, x_{in_i}\} \cup \mathcal{U}_i),$$

is an indicator function that entity k receives a rank no better than j in ranking i where ranking i is also associated with cluster c . It follows that the FCD for λ_{ck} is available in closed form and is

$$\lambda_{ck} | \dots \stackrel{\text{indep}}{\sim} \text{Ga} \left(a_k + \gamma_{ck}, 1 + \sum_{i=1}^n w_i \sum_{j=1}^{n_i} \zeta_{ij}(c, k) z_{ij} \right), \quad (3.9)$$

for $c = 1, \dots, N^r$, $k = 1, \dots, K$.

The only remaining random variables in the model are the ranker weights \mathbf{w} . The full conditional distribution for w_i is the discrete distribution with

$$\begin{aligned} \Pr(w_i = 1 | \mathcal{D}, \Lambda, Z, \mathbf{w}_{-i}, \mathbf{c}^r, \alpha) \\ \propto \Pr(w_i = 1) \pi(\mathcal{D} | w_i = 1, \Lambda, \mathbf{w}_{-i}, \mathbf{c}^r) \pi(Z | w_i = 1, \Lambda, \mathcal{D}, \mathbf{w}_{-i}, \mathbf{c}^r) \\ \propto p_i \prod_{j=1}^{n_i} \lambda_{c_i^r, x_{ij}} \exp \left\{ -z_{ij} \left(\sum_{m=j}^{n_i} \lambda_{c_i^r, x_{im}} + \sum_{m \in \mathcal{U}_i} \lambda_{c_i^r, m} \right) \right\} \end{aligned}$$

and

$$\begin{aligned} \Pr(w_i = 0 | \mathcal{D}, \Lambda, Z, \mathbf{w}_{-i}, \mathbf{c}^r, \alpha) \\ \propto \Pr(w_i = 0) \pi(\mathcal{D} | w_i = 0, \Lambda, \mathbf{w}_{-i}, \mathbf{c}^r) \pi(Z | w_i = 0, \Lambda, \mathcal{D}, \mathbf{w}_{-i}, \mathbf{c}^r) \\ \propto (1 - p_i) \prod_{j=1}^{n_i} \exp \{-z_{ij}(K_i - j + 1)\}. \end{aligned}$$

Therefore, for $i = 1, \dots, n$, the full conditional distribution is given by

$$w_i | \dots \stackrel{\text{indep}}{\sim} \text{Bern}(\rho_i), \quad (3.10)$$

where

$$\rho_i = \frac{\Pr(w_i = 1 | \mathcal{D}, \Lambda, Z, \mathbf{w}_{-i}, \mathbf{c}^r, \alpha)}{\Pr(w_i = 1 | \mathcal{D}, \Lambda, Z, \mathbf{w}_{-i}, \mathbf{c}^r, \alpha) + \Pr(w_i = 0 | \mathcal{D}, \Lambda, Z, \mathbf{w}_{-i}, \mathbf{c}^r, \alpha)},$$

is the probability that ranking i is informative (given the other quantities).

Recall that the latent cluster indicators \mathbf{c}^r can be sampled using Neal's Algorithm 8 (Neal, 2000) which implements a Pólya urn scheme to marginalise out the infinite dimensional parameters. The resulting full conditional distribution for the cluster allocations is a discrete distribution over the clusters which are active and m auxiliary components. The DP concentration parameter α can also be sampled from its full conditional distribution

given in Section 3.4.2. The following section gives a complete outline of the Gibbs sampler used to obtain posterior realisations.

3.5.5 MCMC using Neal's Algorithm 8

We are now in a position to describe the algorithm used to sample from the posterior distribution $\pi(\Lambda, Z, \mathbf{c}^r, \mathbf{w}, \alpha | \mathcal{D})$. First we define the contribution to the complete data likelihood from ranker i to be

$$f(\mathbf{x}_i, \mathbf{z}_i | \Lambda, \mathbf{w}, \mathbf{c}^r, \alpha) = \prod_{j=1}^{n_i} \lambda_{c_i^r, x_{ij}}^{w_i} \exp \left\{ -z_{ij} \left(\sum_{m=j}^{n_i} \lambda_{c_i^r, x_{im}}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_i^r, m}^{w_i} \right) \right\}.$$

We can now describe the algorithm concisely. Suppose we have n rankings of $n_i < K$ entities. The state of the Markov chain has elements $\Lambda = (\boldsymbol{\lambda}_c : c \in \{c_1^r, \dots, c_n^r\})$, $Z = (z_{ij})$, $\mathbf{c}^r = (c_i^r)$, $\mathbf{w} = (w_i)$ and α for $i = 1, \dots, n$, $j = 1, \dots, n_i$. The algorithm repeatedly samples as follows:

- For $i = 1, \dots, n$: Let q^- be the number of distinct c_j^r for $j \neq i$ and $h = q^- + m$. Label these c_j^r values in $\{1, \dots, q^-\}$. If $c_i^r = c_j^r$ for some $j \neq i$, draw $\lambda_{ck} \stackrel{\text{indep}}{\sim} G_{0k}$ for $q^- < c \leq h$, $k = 1, \dots, K$. If $c_i^r \neq c_j^r \forall j \neq i$, let c_i^r have the label $q^- + 1$, and draw $\lambda_{ck} \stackrel{\text{indep}}{\sim} G_{0k}$ for $q^- + 1 < c \leq h$, $k = 1, \dots, K$.

Draw a new value for c_i^r from $\{1, \dots, h\}$ using the following probabilities:

$$\Pr(c_i^r = c | \mathcal{D}, Z, \Lambda, \mathbf{c}_{-i}^r, \mathbf{w}, \alpha) = \begin{cases} b n_{-i,c} f(\mathbf{x}_i, \mathbf{z}_i | \Lambda, \mathbf{w}, c_i^r = c, \mathbf{c}_{-i}^r, \alpha), & 1 \leq c \leq q^-, \\ b \frac{\alpha}{m} f(\mathbf{x}_i, \mathbf{z}_i | \Lambda, \mathbf{w}, c_i^r = c, \mathbf{c}_{-i}^r, \alpha), & q^- < c \leq h, \end{cases}$$

where $\Lambda = \{\boldsymbol{\lambda}_1, \dots, \boldsymbol{\lambda}_h\}$, $n_{-i,c} = \#\{c_j^r = c, j \neq i\}$, and b is the appropriate normalising constant. Change the state to contain only those $\boldsymbol{\lambda}_c$ that are now associated with one or more observations, that is, let $\Lambda = (\boldsymbol{\lambda}_c : c \in \{c_1^r, \dots, c_n^r\})$.

- Sample λ_{ck} from (3.9) for $c = 1, \dots, N^r$, $k = 1, \dots, K$.
- Sample z_{ij} from (3.7) for $i = 1, \dots, n$, $j = 1, \dots, n_i$.
- Sample w_i from (3.10) for $i = 1, \dots, n$.
- Rescale

- Sample $\Lambda^\dagger \sim \text{Ga} \left(N^r \sum_{k=1}^K a_k, 1 \right)$.

- Calculate $\Sigma = \sum_{c=1}^{N^r} \sum_{k=1}^K \lambda_{ck}$.
- For $c = 1, \dots, N^r$, $k = 1, \dots, K$, let $\lambda_{ck} \rightarrow \lambda_{ck} \Lambda^\dagger / \Sigma$.
- Sample α as in Section 3.4.2 with $n = n$ and $N^c = N^r$.

Note that this rescaling step is a straightforward generalisation of that discussed in Section 2.2.3. The generalisation is needed as we now have K unique skill parameters within each of the N^r ranker clusters.

3.5.6 Simulation study – revisiting Dataset 2

For our first simulation study we choose to revisit Dataset 2 introduced in Chapter 2. Recall that this dataset contains $n = 50$ rankings, the first 40 of which are informative rankings and the remaining 10 (labelled 41–50) are uninformative/random permutations. Each ranking within this dataset is a complete ranking of $K = 20$ entities.

Before we can perform Bayesian inference we must first of course choose a suitable prior distribution. As in our previous analyses of these data we choose to let each ordering of the entities be equally likely *a priori*, that is, let $a_k = 1$ for all k with the resulting prior distribution over the skill parameters being $\lambda_{sk} \stackrel{\text{indep}}{\sim} \text{Ga}(1, 1)$. Specifying a (prior) choice for the concentration parameter of the Dirichlet process is somewhat difficult and so we place a prior distribution over α . We choose $a_\alpha = b_\alpha = 1$ so that $\alpha \sim \text{Ga}(1, 1)$, which gives a fairly weak prior distribution over the number of ranker clusters. Note that here the modal prior number of ranker clusters is 1 (with probability 0.19) and thus seems reasonable given the nature of this dataset – see Table 3.1 for the full prior distribution over the number of clusters. We also need to choose prior probabilities that each ranker is informative, that is, specify $p_i = \Pr(w_i = 1)$ for each ranker i . Here we consider 3 analyses, each defined by particular choices of the p_i . Analyses 1 and 2 take the equivalent specification to those studies considered within Section 2.7, that is, for Analysis 1 we let $p_i = 0.5$ (each ranker is equally likely to be informative as it is uninformative) and in Analysis 2 we take $p_i = 0.8$ (the true proportion within these data). For the final analysis (Analysis 3) we assume the standard Plackett–Luce model which is achieved by taking $p_i = 1$. This choice is used to assess how robust our analysis is to assuming all rankers are informative when, in fact, there are uninformative rankers in the dataset. Intuitively we might think that the Dirichlet process mixture model would be flexible enough to cluster together the informative rankings and form a separate cluster to house the uninformative rankings. However, as we shall see, this turns out not to be the case. Analyses 1 and 2 allow us to compare how our DP mixture model performs in comparison to the (single component) homogeneous model considered in Chapter 2.

Posterior analysis

To generate realisations from the posterior distribution (for each analysis) we implemented the sampling algorithm outlined in Section 3.5.5 with $m = 2$ auxiliary variables. Each Markov chain was initialised at a random draw from the prior distribution. To obtain 10K (almost) un-autocorrelated realisations from the posterior distribution we needed to thin the output by factors of 60, 20 and 5 for Analyses 1–3 respectively. We therefore ran the scheme for 600K, 200K and 50K iterations for each respective analysis and also allowed each chain a burn-in period of 10K iterations after initialisation – these samples were discarded. The computational time required to perform inference was (approximately) 126, 33 and 11 seconds for each analysis. The mixing of the MCMC chains was assessed by inspecting trace plots of the log complete data likelihood; see Figure 3.3. This is convenient not only because our state space is vast but also because the dimension of the posterior distribution can change at each iteration (depending upon the number of unique ranker groups). It is therefore not realistic to inspect trace plots of individual parameters within the Markov chain, particularly as cluster labels can swap arbitrarily. Convergence was assessed by initialising numerous chains at differing starting values and verifying that the resulting posterior distributions were equivalent (up to stochastic noise).

We begin by determining the posterior distribution formed under Analysis 3 ($p_i = 1$) – assuming a Dirichlet process mixture of standard Plackett–Luce models. Our intuition *a priori* led us to believe that our mixture model (outlined in Section 3.5.1) might allow the formation of a cluster housing the informative rankers and a separate cluster to house the uninformative rankers. The marginal posterior distribution for the number of ranker

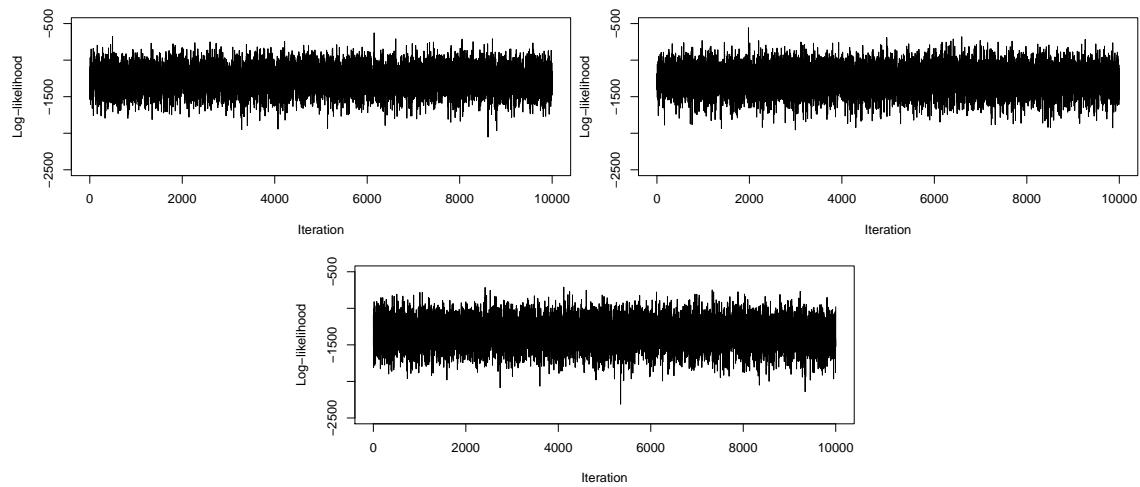


Figure 3.3: Trace plots of the log complete data likelihood for Analyses 1, 2: $p_i = 0.5, 0.8$ (top left and right respectively) and Analysis 3: $p_i = 1$ (bottom)

Analysis	i										E	SD
	1	2	3	4	5	6	7	8	9	≥ 10		
1	0.86	0.12	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.16	0.42
2	0.75	0.19	0.05	0.01	0.00	0.00	0.00	0.00	0.00	0.00	1.32	0.62
3	0.00	0.01	0.04	0.11	0.20	0.24	0.20	0.12	0.06	0.02	6.12	1.68
Prior	0.19	0.17	0.15	0.12	0.10	0.07	0.06	0.04	0.03	0.07	4.18	3.00

Table 3.1: Posterior probabilities of the number of ranker clusters, $\Pr(N^r = i|\mathcal{D})$, for each of the three analyses. The expectation and standard deviation of the marginal posterior distribution are also shown along with the prior distribution. The modal values are highlighted in bold.

groups (Table 3.1) gives $\Pr(N^r = 2|\mathcal{D}) = 0.01$ and so there is little posterior support for this suggestion. Perhaps surprisingly the posterior modal number of ranker groups here is six. However, we note that there is a large amount of uncertainty on this. In contrast, for Analyses 1 and 2, we see significant posterior support for a single homogeneous group with $\Pr(N^r = 1|\mathcal{D}) = 0.86$ and 0.75 respectively. Clearly allowing for uncertainty on ranker reliability results in a significant change in posterior beliefs about the ranker groups contained within these data – this is a feature of the model which will be discussed in more detail later.

The marginal posterior distribution for the number of ranker clusters provides a useful insight into the posterior distribution; however, it does not tell the full story. A more in-depth summary of the posterior distribution can be obtained if we consider the underlying grouping structure of the rankers. The posterior distribution of the allocation of rankers to ranker groups is, of course, quite complex. A common way to summarise ranker heterogeneity is through a single summary allocation to each ranker group, such as the *maximum a posteriori* (MAP) allocation or the improvements to the MAP allocation proposed by Dahl (2006) and Lau and Green (2007). However, these summaries can be misleading unless the posterior probability of the modal number of groups is fairly large. Note that for the Analysis 3 posterior distribution this is certainly not the case. Instead we prefer to summarise ranker heterogeneity using dissimilarity probabilities $\Delta_{ij} = \Pr(c_i^r \neq c_j^r|\mathcal{D})$, that is, the posterior probability that two rankers (i and j) are not allocated to the same cluster. The allocation of rankers to groups could then be determined by thresholding these probabilities. However this too can suffer from inconsistent allocations of say ranker triples, particularly when their dissimilarity probabilities are near the threshold. Therefore, following Medvedovic and Sivaganesan (2002), we use a standard summary method from cluster analysis, namely a dendrogram calculated from the dissimilarity probabilities Δ_{ij} . Note that we consider dendograms formed using the complete linkage method, also known as furthest neighbour clustering. This method tends to produce more densely packed clusters and does not suffer from “chaining”; see Everitt et al. (2011) for further

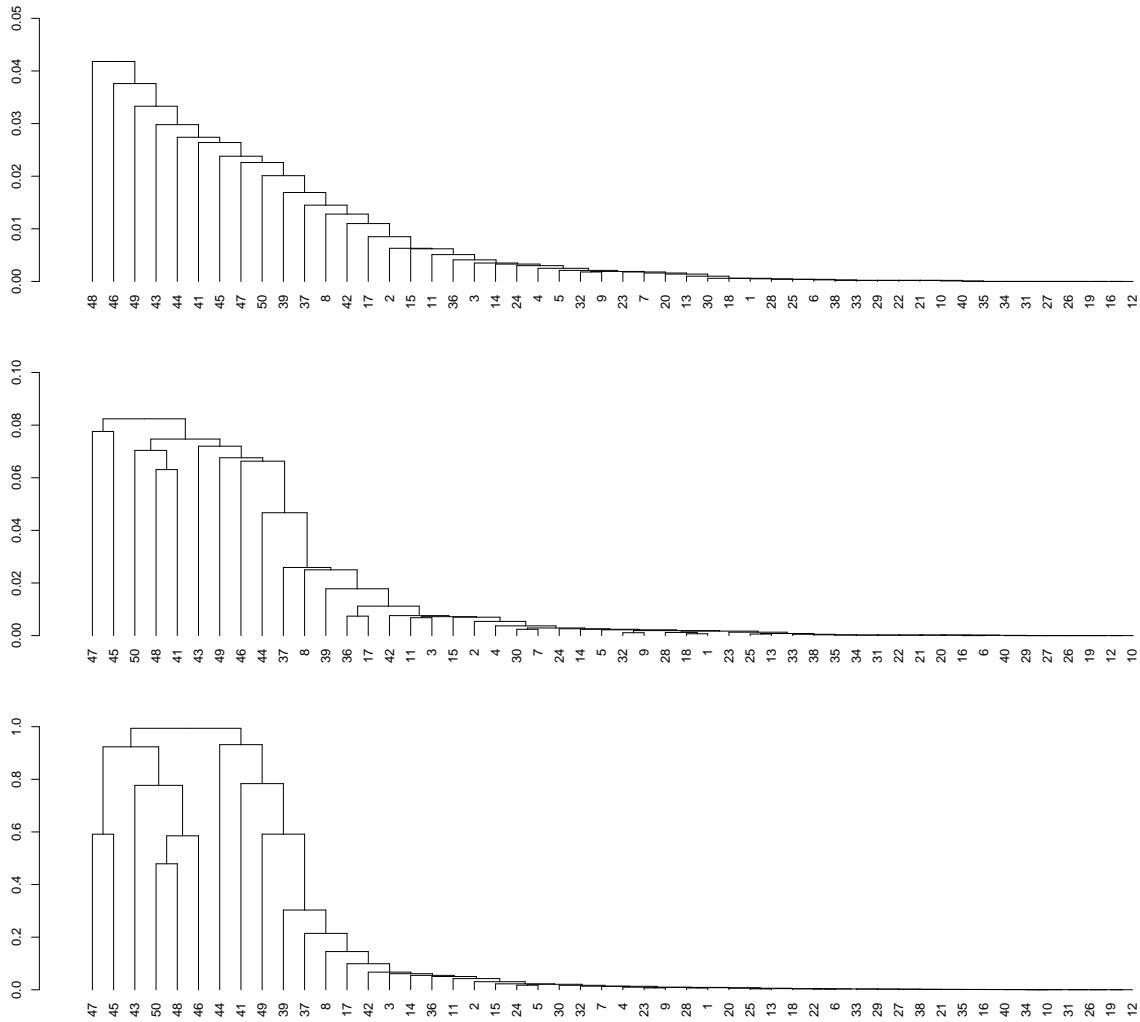


Figure 3.4: Complete linkage dendograms based on the dissimilarities between each pair of rankers for Analyses 1–3 from top to bottom respectively.

details on linkage methods. Of course many other methods could also be used to summarise the heterogeneity between rankers, see, for example, Rastelli and Friel (2017) and the references therein.

Figure 3.4 depicts the dendograms computed from the dissimilarity matrices for each of the analyses considered. The allocation of rankers to ranker groups for Analyses 1 and 2 is somewhat trivial given $\Pr(N^r = 1 | \mathcal{D}) = 0.86$ and 0.75 respectively. The corresponding dendograms confirm that all rankers are often grouped together; evident through the values of dissimilarity at which rankers join the main cluster. However it is encouraging to see that the uninformative rankers (with the exception of ranker 42) are last to join the main cluster: these rankers have the largest dissimilarity values. In Analysis 3 the allocation of

rankers to groups is not quite as straightforward. The corresponding dendrogram shows that there is a large cluster containing those rankers numbered $\{1, 2, \dots, 40, 42\}$. This conclusion can be drawn since $\Delta_{ij} \leq 0.30 \implies (1 - \Delta_{ij}) > 0.70$ for $i \neq j \in \{1, 2, \dots, 40, 42\}$, that is, any pair of rankers within this set are clustered together at least 70% of the time. Given the large proportion of the time that these rankers are co-clustered, it is reasonable to conclude that they have similar beliefs about the entities. The remaining rankers, those numbered 41, 43, \dots , 50, typically have a dissimilarity greater than 0.5. It is clear from looking at the left hand side of the dendrogram that there is no clear grouping structure between any of these rankers. This is perhaps not surprising given their associated rankings are random permutations and are therefore likely to express contradicting preferences.

We now return to the point we noted earlier, namely that allowing for uncertainty on ranker ability changes posterior beliefs about the number of ranker groups. After investigating the posterior distribution for each analysis, perhaps this result is not as surprising as it first seems. In Analysis 3 the standard Plackett–Luce model does not have the flexibility to down weight the contribution uninformative rankers make to the overall likelihood, and this leads instead to the formation of additional clusters to house those rankers which are not consistent with others (the uninformative rankings); see Figure 3.4. Moreover, these rankers do not even form a single homogeneous cluster due to the high variation in random permutations (as mentioned previously). On the other hand, in Analyses 1 and 2 (mixture of Weighted Plackett–Luce models) the model is able to down weight the uninformative rankers; see Figure 3.5. Recall that when $w_i = 0$ the likelihood of ranking i is constant ($\Pr(\mathbf{X}_i = \mathbf{x}_i | \boldsymbol{\lambda}, w_i = 0) = 1/P(K_i, n_i)$) and does not depend on $\boldsymbol{\lambda}$. Thus a ranker who is deemed to be uninformative is free to join a cluster regardless of their beliefs about the entities as the likelihood is unaffected. Indeed, such a ranker will typically join the largest “active” cluster – this follows from *the rich get richer* notion underpinning the Dirichlet process (as mentioned in Section 3.3.1). Consequently it is not surprising that the uninformative rankings (41–50) join the main cluster, that is, the cluster housing the informative rankers under Analyses 1 and 2.

We conclude this section with a brief comparison of Analyses 1–3 and those where we assumed all rankers were homogeneous in their beliefs about the entities in Chapter 2. There are significant similarities between the posterior distributions of the ranker weights under Analyses 1 and 2 in both this section and Section 2.7; see Figures 3.5 and 2.4. This is perhaps not surprising given the ranker weights are not cluster-specific and we have significant posterior support for a single ranker cluster in these analyses. Note that when there is only a single ranker cluster, the analyses presented here are analogous to those in Section 2.7. The aggregate rankings formed under Analyses 1 and 2 here are very similar to those under the corresponding homogeneous analyses considered in Section 2.7; see Tables 2.3 and 3.2. Note that here the aggregate ranking is determined by ordering the

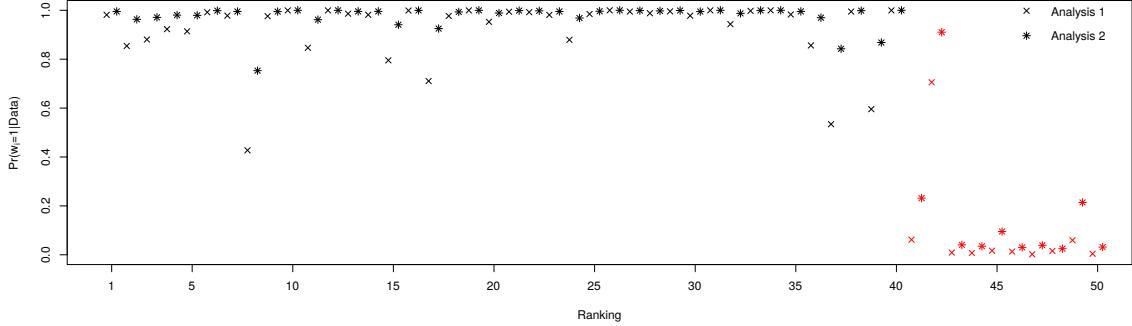


Figure 3.5: $\Pr(w_i = 1 | \mathcal{D})$ – Posterior probability that ranking i is informative under each analysis (Analysis 1: $p_i = 0.5$, Analysis 2: $p_i = 0.8$). Rankings which are random permutations (41–50) are shown in red.

mean of the (fully) marginal posterior distribution for each entity (marginalised over ranker clusters). Recall that, for the homogeneous analyses in Chapter 2 we observed that the aggregate rankings for the analysis of Dataset 2 under the Weighted Plackett–Luce model were equivalent to those formed by analysing Dataset 1 under the standard Plackett–Luce model. This is the case as the WPL model is able to correctly identify the uninformative rankers and down weight them. The same conclusion can be drawn here; see Figure 3.5 and Table 3.2. Unsurprisingly, for Analysis 3 ($p_i = 1$; the standard Plackett–Luce model), the aggregate ranking is affected by the misleading prior information that states that the uninformative rankers are informative. This was also observed when considering the homogenous analysis under the standard Plackett–Luce model; see Section 2.4.

The posterior distribution from Analysis 3 clearly suggests that there is significant heterogeneity between rankers beliefs; see Table 3.1. Summarising such heterogeneous data through an overall aggregate ranking is perhaps not sensible. The differences in preferences between the ranker groups is easily seen though the within-cluster aggregate rankings. Such an aggregate is formed by first conditioning on an appropriate number of ranker groups and then ordering the marginal posterior means of the skill parameters within each group. For Analysis 3, conditioning on 6 ranker groups (the posterior mode), the within-cluster aggregate ranking for ranker cluster 1 (that which typically houses informative rankers) is very similar to the overall aggregate under the other analyses. The remaining within-cluster aggregates (those for ranker clusters 2–6) show little coherence with the true entity preference order and instead appear to be random permutations of the entities. This is perhaps not surprising as these clusters typically house the uninformative rankers.

\hat{x}	λ	Heterogeneous PLW						Homogeneous PL			
		$p_i = 0.5$		$p_i = 0.8$		$p_i = 1$		Dataset 1		Dataset 2	
		Dataset 2		Dataset 2		Dataset 2		x_1^{agg}	$\bar{\lambda}_1$	x_2^{agg}	$\bar{\lambda}_2$
		x^{agg}	$\bar{\lambda}$	x^{agg}	$\bar{\lambda}$	x^{agg}	$\bar{\lambda}$				
1	20.00	3	30.91	3	25.94	3	7.20	3	27.47	3	11.67
2	19.00	1	26.27	1	22.70	2	6.41	1	25.83	1	11.44
3	18.00	5	25.73	5	22.28	1	6.36	5	23.90	2	11.29
4	17.00	2	24.94	2	21.59	5	6.10	2	22.66	4	9.84
5	16.00	4	20.73	4	17.58	4	5.21	4	18.11	9	9.54
6	15.00	6	18.97	6	16.41	9	5.06	6	17.98	5	9.41
7	14.00	8	18.72	8	16.35	8	5.02	8	17.31	8	9.11
8	13.00	7	16.94	7	14.85	6	4.82	7	16.12	6	8.55
9	12.00	9	16.10	9	14.59	7	4.35	9	15.91	7	8.10
10	11.00	10	14.78	10	13.47	10	4.14	10	14.50	10	7.48
11	10.00	11	12.14	11	10.58	11	3.38	11	11.84	11	6.36
12	9.00	12	10.20	12	9.20	12	2.86	12	9.95	12	5.42
13	8.00	13	8.68	13	7.74	13	2.72	13	9.00	13	5.02
14	7.00	14	7.28	14	6.60	16	2.25	14	7.17	14	4.17
15	6.00	16	7.28	16	6.48	14	2.19	16	7.08	16	3.98
16	5.00	15	5.17	15	4.77	15	1.90	15	4.99	15	3.47
17	4.00	17	4.57	17	4.21	17	1.61	17	4.58	17	3.10
18	3.00	18	2.85	18	2.67	19	1.57	18	2.81	19	2.16
19	2.00	19	2.62	19	2.59	18	1.34	19	2.68	18	2.08
20	1.00	20	1.00	20	1.00	20	1.00	20	1.00	20	1.00

Table 3.2: Aggregate rankings under the infinite mixture of Weighted Plackett–Luce model for the analysis of Dataset 2 (for analyses 1–3; $p_i = 0.5, 0.8, 1$) along with the corresponding posterior means. The results from Table 2.2 (homogeneous standard Plackett–Luce analyses) are also given to facilitate comparison.

3.6 Uncovering entity subgroups within a ranker group

We begin by noting that so far within this thesis we have assumed that the preference (strength) of each entity is summarised by a unique skill parameter λ_k . However in this section we now consider the notion that a (homogeneous) group of rankers may not be able to distinguish between some entities, that is, they believe some entities are tied in strength. To allow for this we consider an alternative non-parametric prior distribution which allows entities to cluster together. To achieve clustering on the entities we consider a Dirichlet process prior on the skill parameters, that is, we take a (infinite dimensional) discrete distribution over λ_k such that $\Pr(\lambda_i = \lambda_j) \neq 0$ for $i \neq j$. Note that in what follows we only consider the scenario where there is a single group of rankers ($\mathbf{c}^r = \mathbf{1}$); we relax this assumption in Chapter 4. Therefore, unlike the previous section, we assume that all rankers share similar beliefs about the entities.

3.6.1 The model

Suppose that we have rankings from n rankers where ranker i reports positions (ranks) for $n_i < K$ entities. Here we only consider a single group of rankers and therefore the skill parameter of entity k is denoted by λ_k . The rankers are assumed to be homogeneous and so their rankings follow the same underlying ranking distribution, defined by the Weighted Plackett–Luce model. The model described here is therefore akin to that considered within Section 2.5. However, unlike this previous scenario (where each skill parameter followed a *unique* continuous distribution) all K skill parameters now follow an infinite dimensional discrete distribution G , where G follows a Dirichlet process. This model can be summarised as

$$\begin{aligned} \mathbf{X}_i | \boldsymbol{\lambda} &\stackrel{\text{iid}}{\sim} \text{PLW}(\boldsymbol{\lambda}, w_i) & i = 1, \dots, n, \\ \lambda_k | G &\stackrel{\text{iid}}{\sim} G & k = 1, \dots, K, \\ G | \alpha, G_0 &\sim \text{DP}(\alpha, G_0). \end{aligned}$$

To make the form of the non-parametric prior distribution unambiguous we define its stick-breaking representation, and this is

$$\begin{aligned} G(\cdot) &= \sum_{s=1}^{\infty} \psi_s \delta_{\lambda_s}(\cdot) \\ \psi_s &= v_s \prod_{\ell < s} (1 - v_\ell) \\ v_s &\stackrel{\text{iid}}{\sim} \text{Beta}(1, \alpha) \\ \lambda_s &\stackrel{\text{iid}}{\sim} G_0 \end{aligned}$$

for $s \in \mathbb{N}$. Given the form of the stick-breaking construction it is clear that the atoms of the Dirichlet process are in fact scalar quantities and not parameter vectors as in Section 3.5. Indeed this model only considers a single parameter vector $\boldsymbol{\lambda}$. However, unlike the normal implementation of the Weighted Plackett–Luce model its elements need not be unique. Another important feature of this model (in comparison to the model which accounts for ranker heterogeneity) is that we no longer have the freedom to specify a unique base distribution for each of the K entities. This follows from the exchangeability of the atoms within the Dirichlet process and thus G_0 *must not* depend on s . The implication of this constraint on our prior specification will be discussed further in Section 3.6.3.

3.6.2 Simulating data from the Weighted Plackett–Luce model with entity clustering

In this section we describe how to simulate data from the Weighted Plackett–Luce model with a Dirichlet process prior on the entity skill parameters. As mentioned previously, introducing latent cluster indicator variables is helpful when dealing with mixture models. Here we adopt similar latent cluster indicators and notation as in Section 3.5 where we considered clustering on rankers. We suppose there are N^e entity clusters, that is, the parameter vector for the K skill parameters contains N^e *unique* values. The collection of these unique parameters is denoted $\Lambda = (\lambda_1, \dots, \lambda_{N^e})$. Further we also use latent cluster membership indicators $\mathbf{c}^e = (c_1^e, c_2^e, \dots, c_K^e)$ with $c_k^e \in \{1, 2, \dots, N^e\}$ for $k = 1, 2, \dots, K$. For example, if $\mathbf{c}^e = (1, 2, 1, 2)$ then entities 1 and 3 are in cluster 1 and entities 2 and 4 are in cluster 2; the two clusters have the unique parameters $\lambda_1 \neq \lambda_2$. This is equivalent to saying that entities 1 and 3 are equivalent, and are deemed to be tied in strength; the same is also true for entities 2 and 4. Using these latent cluster indicators, the full skill parameter vector, here denoted $\boldsymbol{\lambda}^\dagger$, that contains the parameter for each entity is given by $\lambda_k^\dagger = \lambda_{c_k^e}$ for $k = 1, \dots, K$.

We can now describe how to generate data under this model (outlined in Section 3.6.1). Again first we must specify a cluster structure but this time for the entities and not the rankers. This can be achieved by either (a) explicitly defining values for the latent cluster allocation variables c_k^e and making sure these are labelled $1, \dots, N^e$, or (b) drawing a realisation (marginally) from the Dirichlet process prior distribution as follows.

- Choose $\alpha > 0$ or simulate from a suitable distribution.
- Set $c_1^e = 1$ and the (current) number of clusters as $N^e = 1$.
- For $k = 2, \dots, K$ simulate the allocation of entity k to a cluster according to the discrete distribution

$$\Pr(c_k^e = j | c_1^e, \dots, c_{k-1}^e) = \frac{n_{kj}^e}{\alpha + k - 1}, \quad \text{for } j = 1, \dots, N^e,$$

$$\Pr(c_k^e = N^e + 1 | c_1^e, \dots, c_{k-1}^e) = \frac{\alpha}{\alpha + k - 1},$$

where n_{kj}^e denotes the number of points currently within cluster j (at iteration k), and $N^e \rightarrow N^e + 1$ if $c_k^e = N^e + 1$.

Once we have a clustering structure of the entities we now choose values for the (cluster-specific) skill parameters λ_s . These can either be chosen explicitly or alternately drawn from the prior (base) distribution by sampling $\lambda_s \stackrel{\text{indep}}{\sim} G_0$ for $s = 1, \dots, N^e$. We also must

choose whether each ranker is informative or not, that is, choose (or sample) a value of $w_i \in \{0, 1\}$ for $i = 1, \dots, n$.

The parameters of the model are now fully specified and so we can now use the exponential latent variable representation of the Weighted Plackett–Luce model to generate rankings. A collection of n complete rankings $\{\mathbf{x}_i\}_{i=1}^n$ is generated via the following process.

For $i = 1, \dots, n$

- Sample $\nu_{ij} \stackrel{\text{iid}}{\sim} \text{Exp}(\lambda_{c_j^e}^{w_i})$ for $j = 1, \dots, K$.
- Set $x_{ij} = \underset{q \in S_{ij}}{\text{argmin}} \nu_{iq}$ where $S_{ij} = \mathcal{K} \setminus \{x_{i1}, \dots, x_{ij-1}\}$ for $j = 1, \dots, K$.

Note that the process outlined above is equivalent to the data generating process for the (no clustering) Weighted Plackett–Luce model as in Section 2.5.1 but here the parameter vector $\boldsymbol{\lambda}$ is replaced with $\boldsymbol{\lambda}^\dagger = (\lambda_{c_1^e}, \dots, \lambda_{c_K^e})$. Alternative types of rankings, for example, a top–5 ranking, can be obtained from the complete rankings simulated using the same process as discussed in Section 2.2.5.

3.6.3 Prior and latent variable specification

We now outline our prior distribution and the likelihood under this model and appeal to the notation and latent cluster indicators introduced in the previous section. We suppose there to be N^e entity clusters and let $\Lambda = (\lambda_1, \dots, \lambda_{N^e})$ denote the collection of the unique skill parameters. The latent cluster membership indicators are given by $\mathbf{c}^e = (c_1^e, c_2^e, \dots, c_K^e)$ with $c_k^e \in \{1, 2, \dots, N^e\}$ for $k = 1, 2, \dots, K$ where $c_k^e = j$ denotes that entity k belongs to cluster j .

We noted earlier that choosing a value for the Dirichlet process concentration parameter α can be difficult. Therefore, as for when we considered ranker clustering, we instead take $\alpha \sim \text{Ga}(a_\alpha, b_\alpha)$ so that we can infer this parameter from the data. We also choose a Gamma prior distribution for the skill parameters so that a conjugate update can be performed (after data augmentation). Recall that for this model we must choose a single base distribution for all skill parameters, that is, G_0 can no longer depend on k . Here we take $G_0 = \text{Ga}(a, 1)$ which gives $\lambda_s \stackrel{\text{iid}}{\sim} \text{Ga}(a, 1)$ *a priori*. Note that not being able to specify a unique prior for each of the k entities has a significant effect on how much information can be fed into the analysis through the prior distribution. Indeed given that $G_0 = \text{Ga}(a, 1)$, the only choice we have is to place an uninformative prior on the skill parameters; namely that each ordering of the entities is equally likely. This is a consequence of the skill parameters λ_k being a random sample for any choice of $a \in \mathbb{R}^+$ and therefore $\Pr(\lambda_i > \lambda_j) = \Pr(\lambda_j > \lambda_i)$ for all $i \neq j$. Given this we might think it is

sufficient to let $a = 1$. However, the value of a still provides information about the variance of the skill parameters, for example. The limited flexibility of the prior distribution for the skill parameters is somewhat of a drawback of this model. Unfortunately there is no way to resolve this given the exchangeability assumption of the Dirichlet process. However, we note that in a real world scenario we might well wish for the data to be the main driving force behind inference and therefore the inability to place strong prior information into the model is perhaps not too important. Indeed we have seen in our previous analyses that taking $a_k = a = 1$ still enables informative inferences on the skill parameters.

Using $G_0 = \text{Ga}(a, 1)$ and recalling that the model contains N^e unique skill parameters, the prior distribution for Λ (conditional on the cluster indicators \mathbf{c}^e) has density

$$\pi(\Lambda | \mathbf{c}^e) = \prod_{c=1}^{N^e} \frac{\lambda_c^{a-1} e^{-\lambda_c}}{\Gamma(a)}.$$

Also the model assumes that each ranking follows the Weighted Plackett–Luce probability, that is, $\mathbf{X}_i | \Lambda, \mathbf{w}, \mathbf{c}^e \stackrel{\text{indep}}{\sim} \text{PLW}(\boldsymbol{\lambda}^\dagger, w_i)$. Therefore we also need a prior distribution for the latent ranker weights, \mathbf{w} . Here we choose the prior specification used in previous analyses, that is, $w_i \stackrel{\text{indep}}{\sim} \text{Bern}(p_i)$ where $p_i \in (0, 1]$ for $i = 1, \dots, n$. We now construct the likelihood under this model. The probability of the i th ranking, expressed in terms of the latent cluster indicator variables, is

$$\Pr(\mathbf{X}_i = \mathbf{x}_i | \Lambda, \mathbf{c}^e, \mathbf{w}) = \prod_{j=1}^{n_i} \frac{\lambda_{c_{x_{ij}}^e}^{w_i}}{\sum_{m=j}^{n_i} \lambda_{c_{x_{im}}^e}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_m^e}^{w_i}},$$

and therefore, as the rankings are (conditionally) independent, the likelihood is

$$\pi(\mathcal{D} | \Lambda, \mathbf{c}^e, \mathbf{w}) = \prod_{i=1}^n \prod_{j=1}^{n_i} \frac{\lambda_{c_{x_{ij}}^e}^{w_i}}{\sum_{m=j}^{n_i} \lambda_{c_{x_{im}}^e}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_m^e}^{w_i}}.$$

Unsurprisingly, and as we have seen previously, the form of the likelihood does not admit conjugate Bayesian inference. We can however use data augmentation techniques by introducing appropriate latent variables so that the full conditional distributions for the skill parameters are simple. Here the latent variables required are again those which correspond to the latent exponential inter-arrival times and (expressed in terms of our latent cluster indicators) are

$$z_{ij} | \mathcal{D}, \Lambda, \mathbf{w}, \mathbf{c}^e \stackrel{\text{indep}}{\sim} \text{Exp} \left(\sum_{m=j}^{n_i} \lambda_{c_{x_{im}}^e}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_m^e}^{w_i} \right), \quad (3.11)$$

for $i = 1, \dots, n$ and $j = 1, \dots, n_i$.

3.6.4 Full conditional distributions

We can now use Bayes' Theorem to obtain the posterior distribution. The posterior distribution $\pi(\Lambda, Z, \mathbf{c}^e, \mathbf{w}, \alpha | \mathcal{D})$ is now a joint distribution of the latent random variables Z , the collection of unique skill parameters Λ , the binary indicator variables \mathbf{w} , the latent cluster allocations \mathbf{c}^e and the DP concentration parameter α . Again we can sample the latent cluster indicators using Neal's Algorithm 8 (Neal, 2000) and the FCD of the DP concentration parameter α is as in Section 3.4.2. The remainder of this section is concerned with deriving the FCDs for the remaining unknown quantities (Z, Λ, \mathbf{w}) and a complete outline of the Gibbs sampling scheme used to generate posterior samples can be found in Section 3.6.5.

Before starting the derivation of the full conditional distributions it is useful to first construct the density of all stochastic quantities. Note however that, given we already have the FCDs for \mathbf{c}^e and α , it is sensible to only consider the conditional density of all the remaining stochastic quantities given the latent cluster indicator variables and the DP concentration parameter. This (conditional) joint density is

$$\begin{aligned}
 & \pi(\Lambda, \mathcal{D}, Z, \mathbf{w} | \mathbf{c}^e, \alpha) \\
 &= \pi(\Lambda, \mathcal{D}, Z, \mathbf{w} | \mathbf{c}^e) \\
 &= \pi(Z | \mathcal{D}, \Lambda, \mathbf{w}, \mathbf{c}^e) \pi(\mathcal{D} | \Lambda, \mathbf{w}, \mathbf{c}^e) \pi(\Lambda | \mathbf{c}^e) \pi(\mathbf{w}) \\
 &= \prod_{i=1}^n \prod_{j=1}^{n_i} \left(\sum_{m=j}^{n_i} \lambda_{c_{x_{im}}^e}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_m^e}^{w_i} \right) \exp \left\{ - \left(\sum_{m=j}^{n_i} \lambda_{c_{x_{im}}^e}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_m^e}^{w_i} \right) z_{ij} \right\} \\
 &\quad \times \prod_{i=1}^n \prod_{j=1}^{n_i} \lambda_{c_{x_{ij}}^e}^{w_i} \left(\sum_{m=j}^{n_i} \lambda_{c_{x_{im}}^e}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_m^e}^{w_i} \right)^{-1} \times \prod_{c=1}^{N^e} \frac{\lambda_c^{a-1} e^{-\lambda_c}}{\Gamma(a)} \times \prod_{i=1}^n p_i^{w_i} (1-p_i)^{1-w_i} \\
 &= \prod_{i=1}^n \prod_{j=1}^{n_i} \lambda_{c_{x_{ij}}^e}^{w_i} \exp \left\{ - \left(\sum_{m=j}^{n_i} \lambda_{c_{x_{im}}^e}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_m^e}^{w_i} \right) z_{ij} \right\} \times \prod_{c=1}^{N^e} \frac{\lambda_c^{a-1} e^{-\lambda_c}}{\Gamma(a)} \\
 &\quad \times \prod_{i=1}^n p_i^{w_i} (1-p_i)^{1-w_i}. \tag{3.12}
 \end{aligned}$$

The derivation of the full conditional distributions (FCDs) for our parameters follows in a similar way to that used for ranker clustering in Section 3.5.4. We now construct the conditional distribution of each random quantity given all other stochastic quantities. By construction, the full conditional distribution for the latent variables Z are as in (3.11) for $i = 1, \dots, n$, $j = 1, \dots, n_i$. This result can also be derived directly from (3.12) as this

gives

$$\pi(Z|\mathcal{D}, \Lambda, \mathbf{w}, \mathbf{c}^e, \alpha) \propto \prod_{i=1}^n \prod_{j=1}^{n_i} \exp \left\{ - \left(\sum_{m=j}^{n_i} \lambda_{c_{x_{im}}^e}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_m^e}^{w_i} \right) z_{ij} \right\}.$$

The full conditional distribution for the *unique* skill parameters λ_c is derived as follows. We have

$$\begin{aligned} \pi(\Lambda|\mathcal{D}, Z, \mathbf{w}, \mathbf{c}^e, \alpha) &\propto \prod_{i=1}^n \prod_{j=1}^{n_i} \lambda_{c_{x_{ij}}^e}^{w_i} \exp \left\{ - \left(\sum_{m=j}^{n_i} \lambda_{c_{x_{im}}^e}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_m^e}^{w_i} \right) z_{ij} \right\} \times \prod_{c=1}^{N^e} \frac{\lambda_c^{a-1} e^{-\lambda_c}}{\Gamma(a)} \\ &\propto \prod_{c=1}^{N^e} \lambda_c^{a+\beta_c-1} e^{-\lambda_c} \times \prod_{i=1}^n \prod_{j=1}^{n_i} \exp \left\{ - \left(\sum_{m=j}^{n_i} \lambda_{c_{x_{im}}^e}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_m^e}^{w_i} \right) z_{ij} \right\} \\ &= \prod_{c=1}^{N^e} \lambda_c^{a+\beta_c-1} \exp \left\{ - \left(1 + \sum_{i=1}^n w_i \sum_{j=1}^{n_i} \gamma_{ij}(c) z_{ij} \right) \lambda_c \right\}, \end{aligned}$$

where

$$\beta_c = \sum_{i=1}^n w_i \sum_{j=1}^{n_i} \mathbb{I}(c_{x_{ij}}^e = c),$$

is the number of times that an entity in cluster c appears within an informative ranking and

$$\gamma_{ij}(c) = \sum_{m=j}^{n_i} \mathbb{I}(c_{x_{im}}^e = c) + \sum_{m \in \mathcal{U}_i} \mathbb{I}(c_m^e = c),$$

is the number of times that entities in cluster c are ranked no better than j th in the i th ranking. It follows that the full conditional distribution for λ_c is available in closed form as

$$\lambda_c | \dots \stackrel{\text{indep}}{\sim} \text{Ga} \left(a + \beta_c, 1 + \sum_{i=1}^n w_i \sum_{j=1}^{n_i} \gamma_{ij}(c) z_{ij} \right), \quad (3.13)$$

for $c = 1, \dots, N^e$.

The only remaining random quantities in the model that we do not currently have a full conditional distribution for are the latent ranker weights \mathbf{w} . Recall that we denote the collection of latent ranker weights excluding that associated with ranker i by $\mathbf{w}_{-i} = (w_1, \dots, w_{i-1}, w_{i+1}, \dots, w_n)$. The full conditional distribution for w_i is the discrete distribution

$$\begin{aligned}
 & \Pr(w_i = 1 | \mathcal{D}, \Lambda, Z, \mathbf{w}_{-i}, \mathbf{c}^e, \alpha) \\
 & \propto \Pr(w_i = 1) \pi(\mathcal{D} | w_i = 1, \Lambda, \mathbf{w}_{-i}, \mathbf{c}^e) \pi(Z | w_i = 1, \Lambda, \mathcal{D}, \mathbf{w}_{-i}, \mathbf{c}^e) \\
 & \propto p_i \prod_{j=1}^{n_i} \lambda_{c_{x_{ij}}^e} \exp \left\{ -z_{ij} \left(\sum_{m=j}^{n_i} \lambda_{c_{x_{im}}^e} + \sum_{m \in \mathcal{U}_i} \lambda_{c_m^e} \right) \right\}, \\
 & \Pr(w_i = 0 | \mathcal{D}, \Lambda, Z, \mathbf{w}_{-i}, \mathbf{c}^e, \alpha) \\
 & \propto \Pr(w_i = 0) \pi(\mathcal{D} | w_i = 0, \Lambda, \mathbf{w}_{-i}, \mathbf{c}^e) \pi(Z | w_i = 0, \Lambda, \mathcal{D}, \mathbf{w}_{-i}, \mathbf{c}^e) \\
 & \propto (1 - p_i) \prod_{j=1}^{n_i} \exp \{-z_{ij}(K_i - j + 1)\}.
 \end{aligned}$$

Therefore, for $i = 1, \dots, n$, the full conditional distribution is

$$w_i | \dots \stackrel{\text{indep}}{\sim} \text{Bern}(\rho_i), \quad (3.14)$$

where

$$\rho_i = \frac{\Pr(w_i = 1 | \mathcal{D}, \Lambda, Z, \mathbf{w}_{-i}, \mathbf{c}^e, \alpha)}{\Pr(w_i = 1 | \mathcal{D}, \Lambda, Z, \mathbf{w}_{-i}, \mathbf{c}^e, \alpha) + \Pr(w_i = 0 | \mathcal{D}, \Lambda, Z, \mathbf{w}_{-i}, \mathbf{c}^e, \alpha)},$$

is the probability that ranking i is informative (given the other quantities).

Recall that the latent cluster indicators \mathbf{c}^e can be sampled using Neal's Algorithm 8 (Neal, 2000) which implements a Pólya urn scheme to marginalise out the infinite dimensional parameters. The resulting full conditional distribution for the cluster allocations is a discrete distribution over the clusters which are active and m auxiliary components. The DP concentration parameter α can also be sampled from its full conditional distribution given in Section 3.4.2. The following section gives a complete outline of the Gibbs sampler used to obtain posterior realisations.

3.6.5 MCMC using Neal's Algorithm 8

Before outlining the Gibbs sampling scheme to generate realisations from the posterior distribution, $\pi(\Lambda, Z, \mathbf{c}^e, \mathbf{w}, \alpha | \mathcal{D})$, it is useful to first define the contribution to the complete data likelihood from ranker i to be

$$f(\mathbf{x}_i, \mathbf{z}_i | \Lambda, \mathbf{w}, \mathbf{c}^e, \alpha) = \prod_{j=1}^{n_i} \lambda_{c_{x_{ij}}^e}^{w_i} \exp \left\{ -z_{ij} \left(\sum_{m=j}^{n_i} \lambda_{c_{x_{im}}^e}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_m^e}^{w_i} \right) \right\}.$$

We can now describe the algorithm concisely. Suppose we have n rankings of $n_i < K$ entities. The state of the Markov chain has elements $\Lambda = (\lambda_c : c \in \{c_1^e, \dots, c_K^e\})$, $Z = (z_{ij})$, $\mathbf{c}^e = (c_k^e)$, $\mathbf{w} = (w_i)$ and α for $i = 1, \dots, n$, $j = 1, \dots, n_i$, $k = 1, \dots, K$. The algorithm repeatedly samples as follows:

- For $i = 1, \dots, K$: Let q^- be the number of distinct c_j^e for $j \neq i$ and $h = q^- + m$. Label these c_j^e values in $\{1, \dots, q^-\}$. If $c_i^e = c_j^e$ for some $j \neq i$, draw values independently from G_0 for those λ_c for which $q^- < c \leq h$. If $c_i^e \neq c_j^e \forall j \neq i$, let c_i^e have the label $q^- + 1$, and draw values independently from G_0 for those λ_c for which $q^- + 1 < c \leq h$.

Draw a new value for c_i^e from $\{1, \dots, h\}$ using the following probabilities:

$$\Pr(c_i^e = c | \mathcal{D}, Z, \Lambda, \mathbf{w}, \mathbf{c}_{-i}^e, \alpha) = \begin{cases} b n_{-i,c} f(\mathbf{x}_i, \mathbf{z}_i | \Lambda, \mathbf{w}, c_i^e = c, \mathbf{c}_{-i}^e, \alpha), & 1 \leq c \leq q^-, \\ b \frac{\alpha}{m} f(\mathbf{x}_i, \mathbf{z}_i | \Lambda, \mathbf{w}, c_i^e = c, \mathbf{c}_{-i}^e, \alpha), & q^- < c \leq h, \end{cases}$$

where $\Lambda = \{\lambda_1, \dots, \lambda_h\}$, $n_{-i,c} = \#\{c_j^e = c : j \neq i\}$ and b is the appropriate normalising constant. Change the state to contain only those λ_c that are now associated with one or more observations, that is, let $\Lambda = (\lambda_c : c \in \{c_1^e, \dots, c_K^e\})$.

- Sample λ_c from (3.13) for $c = 1, \dots, N^e$.
- Sample z_{ij} from (3.11) for $i = 1, \dots, n$, $j = 1, \dots, n_i$.
- Sample w_i from (3.14) for $i = 1, \dots, n$.
- Rescale
 - Sample $\Lambda^\dagger \sim \text{Ga}(N^e a, 1)$.
 - Calculate $\Sigma = \sum_{c=1}^{N^e} \lambda_c$.
 - For $c = 1, \dots, N^e$, let $\lambda_c \rightarrow \lambda_c \Lambda^\dagger / \Sigma$.
- Sample α as in Section 3.4.2 with $n = K$ and $N^c = N^e$.

The rescaling step given here is akin to that mentioned in Section 2.2.3.

3.6.6 Simulation study – revisiting Dataset 1

For our first simulation study we revisit Dataset 1, introduced in Chapter 2. Recall that this dataset contains $n = 40$ complete rankings of $K = 20$ entities, whence $n_i = K$ for $i = 1, \dots, n$. Also note that these data were simulated from the standard Plackett–Luce model. Under our current setting we therefore consider these data to contain $N^e = K = 20$

entity clusters with each entity within its own cluster. Alternatively, and equivalently, we could consider these data as being from the Weighted Plackett–Luce model with a Dirichlet process prior on the skill parameters using the process outlined in Section 3.5.2 with $c_k^e = k$ for $k = 1, \dots, K$, $\lambda = (20, 19, \dots, 1)$ and $w_i = 1$ for $i = 1, \dots, n$.

The purpose of re-analysing these data is to see how this model performs in a scenario where the collection of rankings to be analysed contains a large number of entity clusters; specifically the scenario where each entity is in its own cluster ($N^e = K$). It will be interesting to see if we obtain significant posterior support for 20 entity clusters and if not, how inferences are affected by the introduction of our entity clustering structure. We also take this opportunity to perform a prior sensitivity analysis, and consider how sensitive the posterior distribution is to the choice of prior distribution on the concentration parameter α .

Before we can perform Bayesian inference we must first describe a suitable prior distribution. As in previous analyses of these data we choose to let each ordering (of the entities) be equally likely *a priori*; note that the DP prior also requires this uniform distribution on orderings. As before we let $a = 1$ and so the prior distribution over the skill parameters is $\lambda_k \stackrel{\text{indep}}{\sim} \text{Ga}(1, 1)$ for $k = 1, \dots, K$. As discussed when considering clustering rankers, specifying a (prior) value for the concentration parameter of the Dirichlet process is somewhat difficult. Therefore instead we allow α to be uncertain and assign a suitable prior distribution. We assess how the posterior distribution is affected by the choice of prior for α by considering 4 separate analyses. For Analyses 1 and 2 we use the priors commonly used within the literature (e.g. Rodriguez et al., 2008), namely $a_\alpha = b_\alpha = 1$ and $a_\alpha = b_\alpha = 3$ respectively. In Analyses 3 and 4 we take $b_\alpha = 1$ and consider $a_\alpha = 3$ and $a_\alpha = 5$ for each analysis respectively; the posterior distribution under these analyses will allow us to investigate the effect of increasing the prior mean for α . Table 3.3 (top) shows the (induced) prior distribution for the number of entity clusters for each of the analyses considered. Note that in each case the prior probabilities were obtained by simulation as a closed form of $\pi(N^e | \alpha, K)$ only exists when α is a fixed constant. The prior means and standard deviations of both the number of entity clusters along with the concentration parameter itself are also given in Table 3.3 (bottom) for each analysis. Finally we also need to specify the prior probability that each ranker is informative. Although, in general, it might be pragmatic to use a conservative choice of the p_i we suppose that, for this analysis at least, we are fairly confident that the rankers are informative – these data were simulated from under the standard Plackett–Luce model (equivalent to the Weighted Plackett–Luce model with $w = 1$) – and so take $p_i = 0.9$ for each ranker.

a_α	b_α	1	2	3	4	5	6	7	8	9	10	11	12	13	14	≥ 15
1	1	0.24	0.21	0.17	0.13	0.10	0.06	0.04	0.03	0.01	0.01	0.00	0.00	0.00	0.00	0.00
3	3	0.11	0.21	0.23	0.19	0.12	0.07	0.05	0.01	0.01	0.00	0.00	0.00	0.00	0.00	0.00
3	1	0.02	0.05	0.09	0.12	0.14	0.15	0.13	0.11	0.08	0.05	0.03	0.02	0.01	0.00	0.00
5	1	0.00	0.01	0.02	0.05	0.08	0.11	0.14	0.15	0.14	0.12	0.08	0.05	0.03	0.01	0.01

a_α	b_α	$E(C^e)$	$SD(C^e)$	$E(\alpha)$	$SD(\alpha)$
1	1	3.18	2.07	1	1
3	3	3.45	1.76	1	$1/\sqrt{3}$
3	1	6.23	2.53	3	$\sqrt{3}$
5	1	8.04	2.70	5	$\sqrt{5}$

Table 3.3: Prior probabilities, $\Pr(N^e = j)$, of the number of entity clusters for each analysis (top) and the prior expectations and standard deviations of both the number of entity clusters and the concentration parameter (bottom). Modal values are highlighted in bold.

Posterior analysis

We generate realisations from the posterior distribution (for each analysis) using the Gibbs sampling algorithm outlined in Section 3.6.5. After performing a few pilot runs it appeared that choosing $m = 3$ gave good mixing over the cluster labels. Note that, for this model, increasing m does not effect the computational burden as significantly as when we considered the ranker clusterings as here we are only required to draw m auxiliary entity clusters (scalars) and not ranker clusters (parameter vectors). That said, as when considering ranker clustering, the discrete (full conditional) distribution over the cluster labels also increases in dimension. Each Markov chain was initialised at a random draw from the prior distribution. We ran the MCMC scheme for 110K iterations, discarding the first 10K samples as burn-in and thinning the remaining iterates by a factor of 10. This left a posterior sample of 10K (almost) un-autocorrelated realisations from the posterior distribution. The computational time required was (approximately) 215, 196, 223 and 249 seconds for Analyses 1–4 respectively. Figure 3.6 shows the trace plots of the log complete data likelihood for all analyses. The chains appear to be mixing reasonably well in each case. Again assessing convergence and mixing in this way is convenient not only because the state space is vast but also because the dimension of the posterior distribution changes at each iteration (depending upon the number of unique entity clusters). Convergence has been assessed further by initialising numerous chains at different starting values and verifying that the posterior distribution obtained from each chain is the same up to stochastic noise.

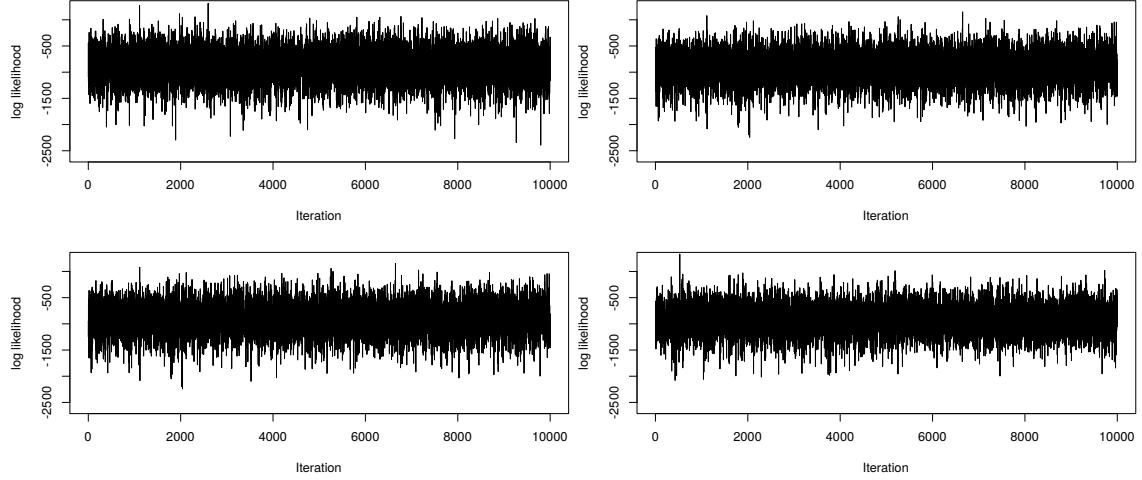


Figure 3.6: Trace plots of the log complete data likelihood for Analyses 1 and 2 (top left, right) and Analyses 3 and 4 (bottom left, right).

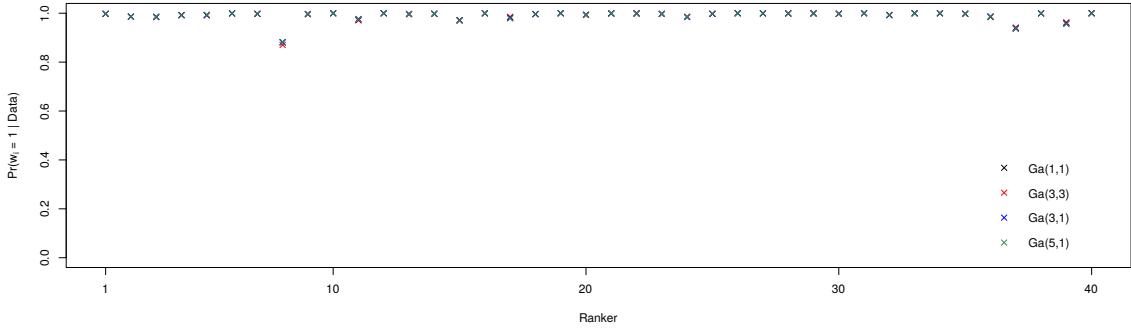


Figure 3.7: $\Pr(w_i = 1 | \mathcal{D})$ – Posterior probability that ranking i is informative under each analysis (Analysis 1: $a_\alpha = b_\alpha = 1$, Analysis 2: $a_\alpha = b_\alpha = 3$). Colours distinguish between the different priors on α .

Under each analysis we observe in Figure 3.7 that the posterior probability of each ranker being informative is large with $\Pr(w_i = 1 | \mathcal{D}) > 0.8$ for $i = 1, \dots, n$. This comes as no surprise as these data were simulated under the standard Plackett–Luce model (which has $w_i = 1$) and we expressed high confidence in each ranker being informative *a priori*. Note that ranker 8 has a lower posterior probability than specified through the prior ($p_i = 0.9$). Closer inspection of this ranking reveals that it is somewhat atypical of this dataset: entities 2 and 4 both appear within the bottom 5 positions and entities 13, 11 and 10 all feature within the top 5 positions. These features are somewhat at odds with the true parameter values from which these data were simulated, and were noted when we analysed these data under the Weighted Plackett–Luce model with no clustering in Section 2.7.

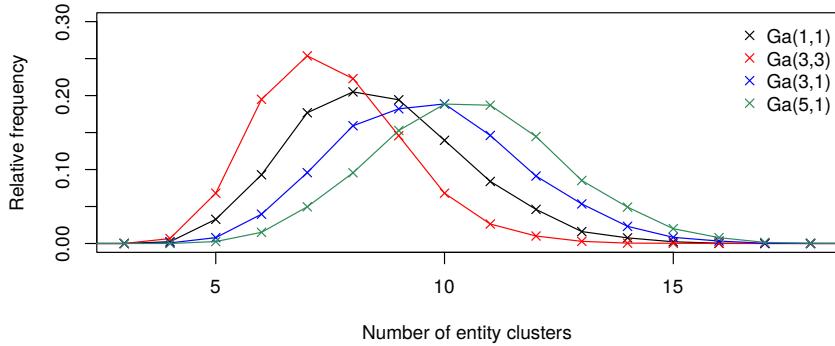


Figure 3.8: $\Pr(N^e = i | \mathcal{D})$ – Marginal posterior distribution of the number of entity clusters for each analysis.

We now turn our attention to the entity clustering. Figure 3.8 shows the marginal posterior distribution of the number of entity clusters for each analysis (as frequency polygons). The first striking observation we note is the fairly large difference in the marginal posteriors between analyses. Of course, this aspect of the posterior distribution was always quite likely to be affected given that the concentration parameter α controls the level of entity clustering. It appears, for these data at least, that the prior beliefs for α play an important role in the analysis. This might be due to there being little information in these data about the number of entity clusters: recall that these data only contain 40 rankings of 20 entities.

If we compare the marginal posterior distributions of the number of entity clusters for Analyses 1 and 2, that is, those which specify a unit mean on the concentration parameter *a priori*, we note that there is less posterior variation for the latter. This suggests that, for these data, information about the variation on the number of entity clusters contained within the prior also plays an important role in the analysis – recall the prior distributions for these analyses specify standard deviations of 1 and $1/3$ for α respectively. Further, and perhaps not surprisingly, we observe that as the prior mean for α increases, so does the posterior mean on the number of entity clusters; see Analyses 1, 3 and 4 in Figure 3.8.

As with ranker clustering, the marginal posterior for the number of clusters does not tell the full story. Again we use complete linkage dendograms formed from the dissimilarity matrix with entries Δ_{ij} , where $\Delta_{ij} = \Pr(c_i^e \neq c_j^e | \mathcal{D})$ is the posterior probability that entities i and j are not clustered together. Figure 3.9 shows the dendograms of entity clustering for each analysis. It is clear that the clustering structure shown within the dendograms is similar for each analysis and so this aspect of the posterior distribution is fairly robust to the choice of prior on α (unlike the marginal posterior over the number of entity clusters). As the prior mean for α increases we observe that the dissimilarity values at which clusters form increases, that is, rankers are co-clustered together less often. This is consistent with observing an increased number of entity clusters. The only notable change

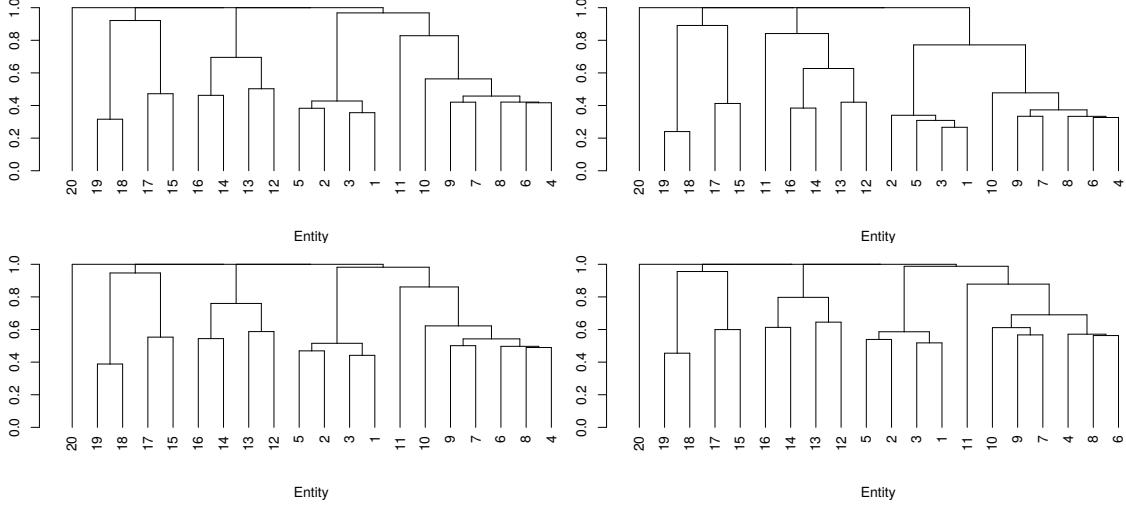


Figure 3.9: Dendrograms of entity clustering for Analyses 1, 2 (top left and right) and Analyses 3, 4 (bottom left and right).

in clustering structure from the dendrograms is that entity 11 has changed allegiance from entities $\{4, 6, \dots, 10\}$ to entities $\{12, 13, 14, 16\}$ under Analysis 2 (top right). It is pleasing to see that although our model is unable to detect each entity is within its own cluster (they have unique values with $\lambda = (20, 19, \dots, 1)$) those entities which form groups are typically those with labels that are most similar.

It is interesting to see how inferences on the skill parameter values are affected by incorporating entity clustering into the model. Table 3.4 shows the marginal posterior means for each of the K entities under all analyses considered. Note that to ease comparison the skill parameters have been rescaled (offline) so that λ_{20} takes its true value, that is, we let $\lambda_k \rightarrow \lambda_k/\lambda_{20}$ for each realisation from our posterior distribution. The aggregate rankings (formed by ordering entities by their posterior mean) are the same for each analysis and therefore this is only given once in the table. It follows that this aspect of the posterior appears to be fairly robust to the choice of prior for α . The (posterior) aggregate ranking(s) are fairly similar to the *optimal* ranking, $\hat{\mathbf{x}} = (1, 2, \dots, 20)$, formed conditionally on the true parameter values. Recall the optimal ranking is that which maximises the Plackett–Luce probability and is formally defined in (2.7). Further if we compare posterior inferences here to those when we assumed the (no clustering) standard Plackett–Luce model in Section 2.4.1, we notice striking similarities. The aggregate rankings are the same and there is little discrepancy between the posterior means of the skill parameters for each model. Therefore, for these data, the posterior inferences are robust to incorporating entity clustering structure within the model. We note however that the posterior distribution formed under the model that allows for entity clustering is much richer in

Rank	\mathbf{x}^{agg}	Analysis			
		1	2	3	4
1	3	24.76	23.83	25.39	25.88
2	1	24.03	23.34	24.57	24.85
3	5	23.10	22.56	23.44	23.71
4	2	22.56	22.04	22.75	22.95
5	4	19.66	19.68	19.61	19.61
6	6	19.50	19.45	19.39	19.34
7	8	19.04	19.09	18.91	18.84
8	7	18.24	18.36	18.04	17.95
9	9	18.08	18.19	17.85	17.69
10	10	16.75	16.84	16.55	16.36
11	11	13.02	12.97	12.98	13.04
12	12	10.18	10.01	10.24	10.32
13	13	9.21	9.10	9.28	9.33
14	14	7.66	7.66	7.60	7.59
15	16	7.61	7.58	7.56	7.55
16	15	5.40	5.41	5.38	5.36
17	17	4.95	4.92	4.92	4.91
18	18	3.02	3.02	2.99	2.97
19	19	2.95	2.96	2.92	2.89
20	20	1.00	1.00	1.00	1.00

Table 3.4: Marginal posterior means of the skill parameters for each analysis. The aggregate ranking is the same under all analyses.

information. For example, we can quantify, in a principled manner, the (posterior) level of similarity between entities – something that would require an *ad hoc* approach under a standard (no clustering) analysis.

3.7 Summary

In this chapter we have shown that it is possible to reveal latent group structure contained within ranked data by appealing to Dirichlet process mixture models. In Section 3.5 we explored the area given much attention in the literature, namely revealing differences between rankers’ preferences. We used an infinite mixture of Weighted Plackett–Luce models which, through simulation studies, was shown to be an appropriate model for analysing such data. In Section 3.6 we considered the notion that rankers may not be able to distinguish between certain groups of entities, that is, they consider some entities to be indistinguishable (tied in strength). The analysis used a novel (Dirichlet process) prior distribution over the skill parameters themselves – something which, to the best of

our knowledge, has not previously been considered within the literature. This allowed for the exploration of (potential latent) clustering structure within the entities. Further the richness of information within the posterior distribution allows us to quantify the level of similarity between entities – this would require an *ad hoc* approach if using standard (no clustering) techniques. Efficient (marginal) posterior sampling schemes were discussed and a Gibbs sampling strategy (made possible by appealing to data augmentation techniques) was outlined for each model.

We acknowledge that the simulation studies within this chapter consider data which were simulated from homogeneous models. However, we believe our models performed sufficiently well and gave reasonable inferences even in the scenario where no ranker or entity clustering was present. It was particularly interesting to see that incorporating our DP prior on the skill parameters had little effect on posterior inferences.

In the next chapter we explore two-way clustering techniques with the aim of building a *single* model which can explore not only heterogeneity between rankers but also the clustering structure of entities within ranker groups. As part of this we consider simulation studies on data where (true) clustering structure is present and therefore the effectiveness of our models (in recovering both ranker and entity group structure) will be examined within the latter part of Chapter 4.

Chapter 4

The Bayesian WAND

4.1 Introduction

In Chapter 3 (Sections 3.5 and 3.6) we presented two different non-parametric prior distributions which allowed for either clustering of rankers or clustering of entities. In this chapter we develop a non-parametric prior distribution which allows for the clustering of both rankers *and* entities. We do this by appealing to other Bayesian non-parametric priors used for two-way clustering. We begin by reviewing briefly some of the existing methods within the literature before describing the non-parametric prior distribution we shall use for clustering both rankers and entities.

4.2 Two-way clustering

There are a few Bayesian non-parametric priors which allow for multiple layers of clustering via Dirichlet processes. Two of these are the Hierarchical Dirichlet Process (Teh et al., 2006) and the Nested Dirichlet Process (Rodriguez et al., 2008) which, for conciseness, we refer to as the HDP and NDP respectively.

The HDP has the typical model specification

$$\begin{aligned}\boldsymbol{\lambda}_i | G_i &\stackrel{\text{indep}}{\sim} G_i, & i = 1, \dots, n, \\ G_i | \alpha, G_0 &\stackrel{\text{indep}}{\sim} \text{DP}(\alpha, G_0), & i = 1, \dots, n, \\ G_0 | \gamma, H &\sim \text{DP}(\gamma, H).\end{aligned}$$

Under this prior each sample is drawn from a distribution over a common set of atoms (which is a realisation of a Dirichlet process) whose base distribution is in turn another

Dirichlet process realisation (with associated base distribution H). In our setting a typical realisation from this prior is an $n \times K$ matrix Λ containing the skill parameter vectors for each of the n rankers. The skill parameter vectors for each ranker are drawn from the same set of atoms. However, these atoms have different weights for each of the n rankers. One way to think about this is that each ranker is first assigned their own *unique* DP before drawing a sample from this DP for their K skill parameters.

The NDP is slightly different and under this NDP prior there is a single Dirichlet process whose atoms are themselves unique Dirichlet processes. As with the HDP, a typical realisation (in our context) from this prior is the parameter matrix Λ containing the skill parameter vectors. However, unlike the HDP, the NDP stipulates that two realisations of the skill parameter vector, say $\boldsymbol{\lambda}_1$ and $\boldsymbol{\lambda}_2$, are either drawn from a distribution (realisation of a DP) over the same atoms with the same weights, or alternatively, a distribution over different atoms with different weights. Formally the NDP is defined through its stick breaking representation and we let $Q|\alpha, \gamma, H \sim \text{NDP}(\alpha, \gamma, H)$ denote that Q follows the Nested Dirichlet Process prior distribution with stick-breaking representation

$$\begin{aligned}
 Q(\Lambda) &= \sum_{s=1}^{\infty} \psi_s \delta_{P_s}(\Lambda), \\
 \psi_s &\stackrel{\text{iid}}{\sim} v_s \prod_{\ell < s} (1 - v_\ell), \quad s = 1, 2, \dots, \\
 v_s &\stackrel{\text{iid}}{\sim} \text{Beta}(1, \alpha), \quad s = 1, 2, \dots, \\
 P_s(\Lambda) &= \sum_{t=1}^{\infty} \omega_{st} \delta_{\lambda_{st}^\dagger}(\lambda^\dagger), \quad s = 1, 2, \dots, \\
 \omega_{st} &= u_{st} \prod_{\ell < t} (1 - u_{s\ell}), \quad s = 1, 2, \dots, \quad t = 1, 2, \dots, \\
 u_{st} &\stackrel{\text{iid}}{\sim} \text{Beta}(1, \gamma), \quad s = 1, 2, \dots, \quad t = 1, 2, \dots, \\
 \lambda_{st}^\dagger &\stackrel{\text{iid}}{\sim} H, \quad s = 1, 2, \dots, \quad t = 1, 2, \dots.
 \end{aligned}$$

A typical model specification for the NDP is therefore

$$\begin{aligned}
 \boldsymbol{\lambda}_i | G_i &\sim G_i, & i = 1, \dots, n, \\
 G_i | Q &\sim Q, & i = 1, \dots, n, \\
 Q | \alpha, \gamma, H &\sim \text{NDP}(\alpha, \gamma, H),
 \end{aligned}$$

where Q follows a Nested Dirichlet Process prior with concentration parameters α, γ and base distribution H .

Unfortunately neither of these priors are appropriate for our problem. They are designed for situations where x_{ij} is itself an observation. However, for ranked data, this is not the case as an observation is the entity ranking (vector) \mathbf{x}_i . In our setting both of these priors assign a distribution (realisation of a DP) to each ranker and then draw samples for the K skill parameters (one for each entity) based solely on the information contained within that single ranking. However, in order to cluster entities (within each ranker group) we require information from numerous rankers. The properties of the NDP are somewhat desirable and we therefore adapt this prior distribution so that it can be applied in a ranked data context. The adaptation required is fairly straightforward and is discussed in detail within the next section.

4.3 The Adapted Nested Dirichlet Process (ANDP) prior

As mentioned we need to adapt the Nested Dirichlet Process prior so that it can be used within a ranked data context and enable clustering of both the rankers and entities. Under the standard NDP, rankers are first assigned to a distribution (realisation of a DP) before a sample of the K skill parameters is then drawn (independently) for each ranker. However we adapt the prior so that we first assign all rankers to a distribution (realisation of a DP) before then proceeding to draw a *single* sample (of the K skill parameters) from each of the unique DP realisations that rankers are assigned to. These samples are drawn based on the information from all rankers which are assigned to each respective DP realisation. Further the single sample (drawn from each respective DP realisation) is shared between all rankers within that “cluster”. This results in a slightly different prior to the NDP, which we call the Adapted Nested Dirichlet Process (ANDP) prior, and this dictates that those rankers who are assigned to the same DP realisation (cluster) have the *exact* same skill parameter vector λ . Recall that the NDP only requires that the parameter vectors for each of the rankers (assigned to the same cluster) are drawn from the same distribution (realisation of a DP).

The Adapted Nested Dirichlet Process prior distribution has a “top level” Dirichlet process whose atoms are parameter vectors λ . Each of these parameter vectors is a sample from a *unique* realisation of a “low level” Dirichlet process. We let $G|\alpha, \gamma, G_0 \sim \text{ANDP}(\alpha, \gamma, G_0)$

denote that G follows the ANDP prior distribution with stick-breaking representation

$$G(\Lambda) = \sum_{s=1}^{\infty} \psi_s \delta_{\lambda_s^*}(\lambda^*), \quad (4.1)$$

$$\psi_s \stackrel{\text{indep}}{\sim} v_s \prod_{\ell < s} (1 - v_\ell), \quad s = 1, 2, \dots,$$

$$v_s \stackrel{\text{indep}}{\sim} \text{Beta}(1, \alpha), \quad s = 1, 2, \dots,$$

$$P(\lambda_s^*) = \sum_{t=1}^{\infty} \omega_{st} \delta_{\lambda_{st}^\dagger}(\lambda^\dagger), \quad s = 1, 2, \dots, \quad (4.2)$$

$$\omega_{st} = u_{st} \prod_{\ell < t} (1 - u_{s\ell}), \quad s = 1, 2, \dots, \quad t = 1, 2, \dots,$$

$$u_{st} \stackrel{\text{indep}}{\sim} \text{Beta}(1, \gamma_s), \quad s = 1, 2, \dots, \quad t = 1, 2, \dots,$$

$$\lambda_{st}^\dagger \stackrel{\text{indep}}{\sim} G_0, \quad s = 1, 2, \dots, \quad t = 1, 2, \dots.$$

It is worth noting that the NDP (and therefore the ANDP) priors are usually specified using two concentration parameters, one controls the top level clustering (in our case rankings) and the second corresponds to the lower level clustering (the entity clustering). However, for the ANDP we choose to instead introduce an infinite dimensional space for our low level concentration parameters, that is, we introduce γ_s for $s \in \mathbb{N}$ and let $\gamma = (\gamma_1, \gamma_2, \dots)$ be the collection of these concentration parameters. Although this change might seem somewhat incidental it means that the ANDP prior now has more flexibility to handle differing levels of entity clustering within each ranker group. Note that as the low level Dirichlet processes are themselves atoms of the high level Dirichlet process, the associated concentration parameters γ_s *must* be exchangeable (with respect to the cluster label s). This has the consequence that if these parameters are chosen to be fixed constants, they must all be equal, that is, $\gamma_s = \gamma$ with $\gamma > 0$ and $s \in \mathbb{N}$. In this scenario the additional modelling flexibility is lost and the concentration parameter (for the entity clustering) is the same across ranker groups. Alternatively γ_s can be given a prior distribution which does not depend on the cluster label s . In other words, we can choose $\gamma_s \stackrel{\text{indep}}{\sim} f(\cdot)$ *a priori* but *not* $\gamma_s \stackrel{\text{indep}}{\sim} f_s(\cdot)$ due to the exchangeability requirement of the top level Dirichlet process. We note that if the density $f(\cdot)$ is a mixture of Gamma distributions then, as was the case when we considered one-way clustering (of either rankers or entities), the full conditional distribution for each γ_s is straightforward. Of course, other prior specifications could be chosen but come at the cost of a loss of conjugacy. We also note that the concentration parameter of the top level Dirichlet process (α) remains a scalar and controls clustering of the rankers.

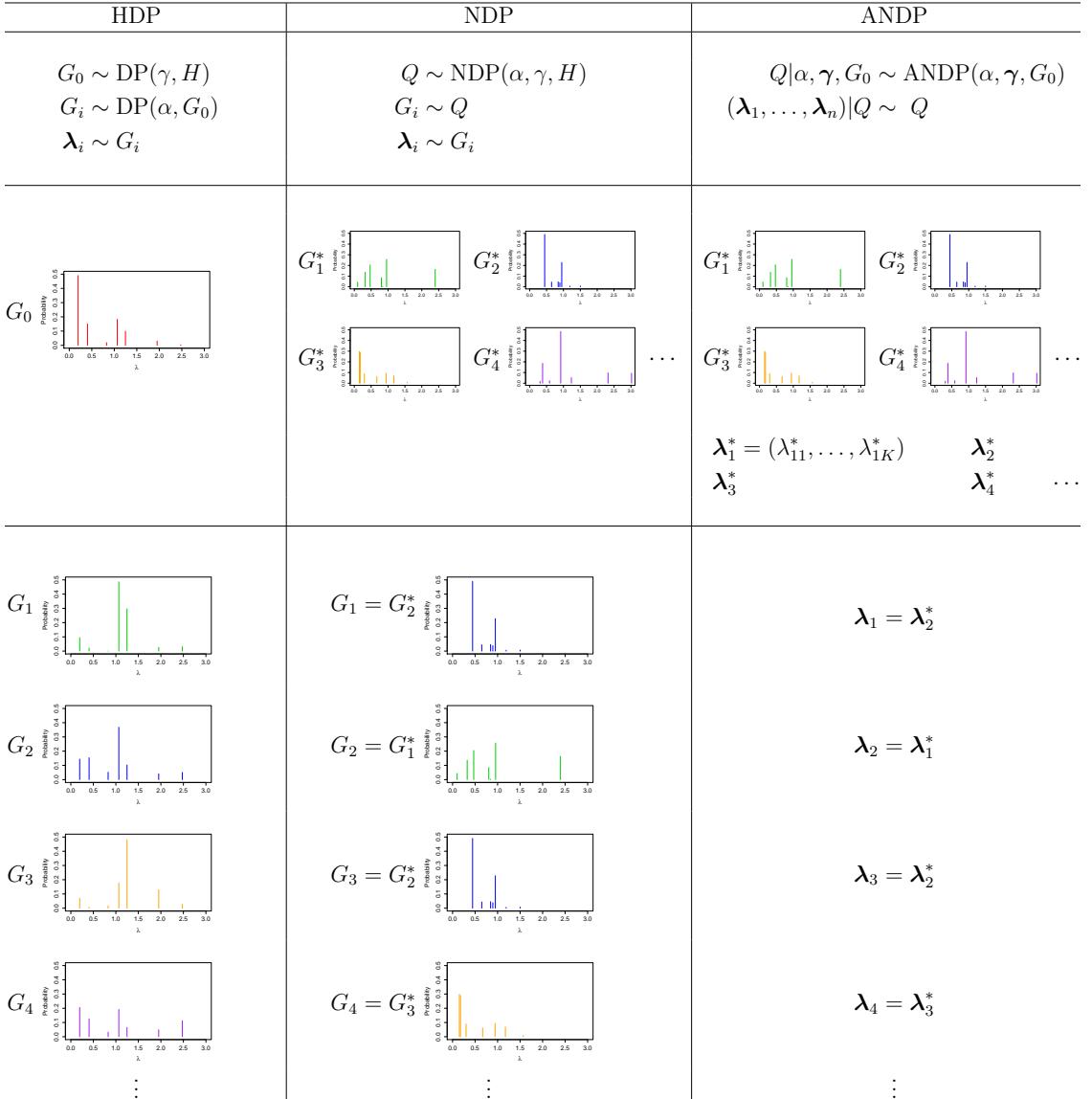


Figure 4.1: Comparison of non-parametric prior distributions for two-way clustering

Figure 4.1 gives a graphical representation of the HDP, NDP and ANDP (based on Figure 1 in Rodriguez et al. (2008)) and thus clarifies the differences between these non-parametric prior distributions.

4.3.1 Generating prior samples (stick-breaking representation)

We now describe how to obtain realisations from the ANDP prior distribution when the data contain n rankers and K entities. Recall that a typical observation from this prior distribution is the matrix Λ which holds the parameter vectors for each of the n rankers.

Such a realisation can be obtained fairly trivially using the stick-breaking representation outlined by (4.1) and (4.2). A marginal method using an appropriate Pólya urn scheme will be considered later within this chapter. When implementing the stick-breaking approach for a single Dirichlet process we noted that we must first choose a suitable truncation parameter which results in a reasonable approximation to the infinite dimensional distribution defined by the Dirichlet process. It is perhaps not surprising that the ANDP requires two truncation parameters, N_1 and N_2 , so that the distributions G and P defined in (4.1) and (4.2) are reasonably approximated. These truncation parameters must be chosen so that

$$\sum_{s=N_1+1}^{\infty} \psi_s \simeq 0 \quad \text{and} \quad \sum_{s=1}^{N_1} \sum_{t=N_2+1}^{\infty} \omega_{st} \simeq 0, \quad (4.3)$$

or, equivalently, so that

$$\sum_{s=1}^{N_1} \psi_s \simeq 1 \quad \text{and} \quad \sum_{t=1}^{N_2} \omega_{st} \simeq 1 \quad \text{for } s = 1, \dots, N_1.$$

We can then generate a prior realisation using the following process.

- Choose N_1 and N_2 sufficiently large.
- Choose $\alpha > 0$ or sample from an appropriate prior distribution.
- Choose $\gamma_s = \gamma > 0$ or sample γ_s (independently) from an appropriate prior distribution which is exchangeable in s for $s = 1, \dots, N_1$.
- Sample $\lambda_{st}^{\dagger} \stackrel{\text{indep}}{\sim} G_0$ for $s = 1, \dots, N_1$, $t = 1, \dots, N_2$.
- Sample $u_{st} \stackrel{\text{indep}}{\sim} \text{Beta}(1, \gamma_s)$ for $s = 1, \dots, N_1$, $t = 1, \dots, N_2$.
- Set $\omega_{st} = u_{st} \prod_{\ell < t} (1 - u_{s\ell})$ for $s = 1, \dots, N_1$, $t = 1, \dots, N_2$.
- Sample λ_{sk}^* from the discrete distribution(s) with atoms $\boldsymbol{\lambda}_s^{\dagger}$ and weights $\boldsymbol{\omega}_s$ for $k = 1, \dots, K$ and $s = 1, \dots, N_1$.
- Sample $v_s \stackrel{\text{indep}}{\sim} \text{Beta}(1, \alpha)$ for $s = 1, \dots, N_1$.
- Set $\psi_s = v_s \prod_{\ell < s} (1 - v_{\ell})$ for $s = 1, \dots, N_1$.
- Sample $\boldsymbol{\lambda}_i$ from the discrete distribution with atoms $\{\boldsymbol{\lambda}_1^*, \dots, \boldsymbol{\lambda}_{N_1}^*\}$ and weights $\boldsymbol{\psi}$ for $i = 1, \dots, n$.

We now explore this prior distribution further in the next section by investigating the effect of the concentration parameters on prior realisations.

4.3.2 Exploring the ANDP prior

We now investigate how the concentration parameters affect realisations from the ANDP prior distribution. Suppose we have $n = 50$ rankers and $K = 20$ entities. It follows that the maximum number of *unique* parameter values that could be used to summarise these data is $n \times K = 1000$; this scenario would arise when each ranker is assigned to their own cluster and each entity (within each ranker group) is also within its own cluster. Also the maximum number of unique parameter vectors (ranker clusters) is $n = 50$. Figure 4.2 shows the number of unique values of λ (total number of unique skill parameters across *all* ranker clusters) along with the number of ranker clusters (unique parameter vectors) there are for varying values of α and γ . Here we fix the concentration parameters in ranker groups to be constant and so set $\gamma_s = \gamma$ for $s = 1, \dots, N_1$. Note that Figure 4.2 shows the empirical prior probabilities calculated from one million (independent) prior realisations drawn using the method described in Section 4.3.1.

For fixed α , the prior distribution of the number of ranker clusters (the number of unique parameter vectors $\boldsymbol{\lambda}$) is the same for all γ values; see Figure 4.2 (top left). This follows as the DP (4.1) whose atoms are skill parameter vectors is conditionally independent of γ given $\boldsymbol{\lambda}_s^*$ for $s \in \mathbb{N}$. The consequence of this is that γ plays no role in the level of ranker clustering and this aspect of the prior distribution is controlled by α alone. As when we considered clustering rankers (Section 3.5), the number of unique ranker clusters increases as α increases; see Figure 4.2 (top right). It follows from our first observation that, for any particular choice of α , the prior distribution of the number of ranker clusters is the same irrespective of γ . Therefore, for this aspect of the prior distribution, allowing both α and γ to vary is equivalent to only considering variation on α . Before we consider the level of entity clustering, recall that here we are considering the *total* number of entity clusters across all ranker clusters, that is, the total number of unique skill parameters λ , and not the number of entity clusters within individual ranker clusters. As expected, for fixed α , the number of entity clusters increases as γ increases; see Figure 4.2 (bottom left). Note that here we have larger prior uncertainty for the number of entity clusters compared to when we had clustering on entities alone. This follows as we have additional uncertainty on the number of ranker clusters with the ANDP prior. Also, with fixed γ , the number of unique entity clusters increases as α increases; see Figure 4.2 (bottom right). This is perhaps counter-intuitive as, for fixed γ , the prior distribution of the number of entity clusters (within a ranker group) remains unchanged. However, as α increases, the number of ranker clusters increases and each unique ranker cluster contains at least one entity cluster. Therefore, as the number of ranker clusters increases (and therefore the number of parameter vectors $\boldsymbol{\lambda}$) so too does the number of unique λ values. This follows as the skill parameter vector for each different ranker cluster is drawn from a *unique* distribution

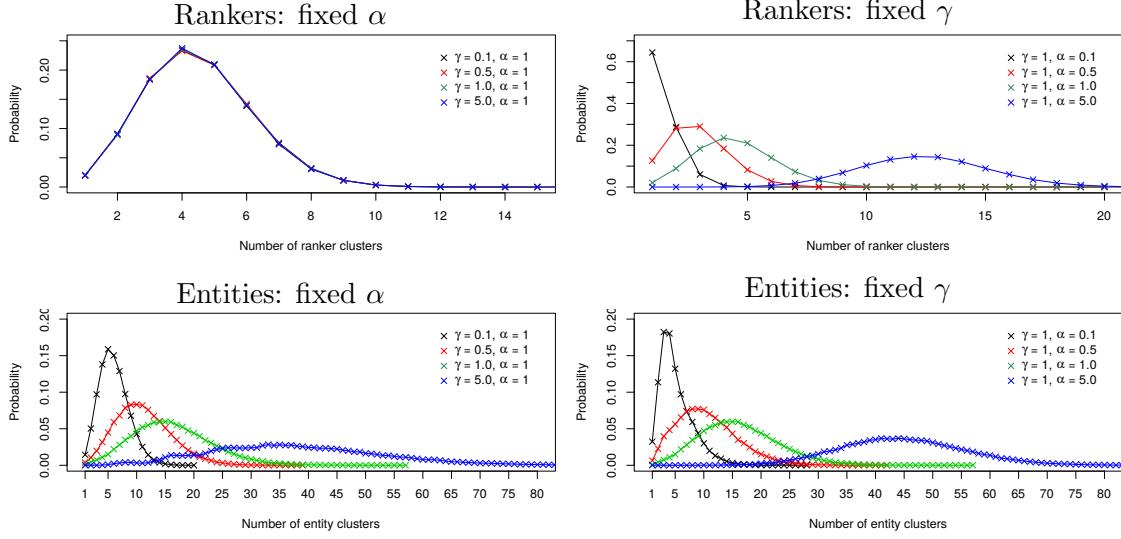


Figure 4.2: Number of ranker and entity clusters under the Adapted Nested Dirichlet Process prior for various values of concentration parameters α and γ .

(realisation of a DP) and therefore $\Pr(\lambda_{c_1 i} = \lambda_{c_2 j}) = 0$ for two unique cluster labels $c_1 \neq c_2$ and all $i, j \in \{1, \dots, K\}$.

4.4 The model

We now describe the full model which incorporates both the ranker reliability parameter and the two-way clustering of both rankers and entities (within ranker groups). The model is an infinite mixture of Weighted Plackett–Luce models with the ANDP chosen as the prior distribution. We refer to this model as the Weighted Adapted Nested Dirichlet (WAND) process mixture of Plackett–Luce models. The main components of this model can be written as

$$\begin{aligned} \mathbf{X}_i | \boldsymbol{\lambda}_i, w_i &\sim \text{PLW}(\boldsymbol{\lambda}_i, w_i), \quad i = 1, \dots, n, \\ (\boldsymbol{\lambda}_1, \dots, \boldsymbol{\lambda}_n) | Q &\sim Q \\ Q | \alpha, \gamma, G_0 &\sim \text{ANDP}(\alpha, \gamma, G_0), \end{aligned}$$

where the stick-breaking representation of the ANDP prior is as in Section 4.3.

4.5 A conditional sampling approach

In Section 3.4.1 we discussed different methods of implementing MCMC sampling algorithms for Dirichlet Process mixture models. When considering one-way clustering (Chapter 3) it was fairly straightforward to implement a marginal sampling scheme based on Neal’s Algorithm 8 (Neal, 2000). However this non-parametric prior distribution is more complex due to the two-way clustering it induces. Although there are inherent problems with implementing conditional sampling schemes (such as the choice of truncation parameters and the requirement of label switching moves) these methods are often the most intuitive and so we consider them a good place to start in this case.

4.5.1 Simulating data from the WAND model

In this section we describe how to simulate data from the Weighted Adapted Nested Dirichlet (WAND) process mixture of Plackett–Luce models. Again it is helpful to first introduce latent cluster indicators. Let $\mathbf{c} = (c_1, \dots, c_n)$ where $c_i = j$ denotes that ranking i is associated with parameter vector $\boldsymbol{\lambda}_j$. We also need indicators to denote the entity clustering within each ranker group (parameter vector). Let $d_{ij} = \ell$ denote that entity j within parameter vector i is allocated to entity cluster ℓ and let $D = (d_{ij})$ denote the matrix of latent entity cluster indicators. In this notation, the value of the skill parameter assigned to entity j from ranking i is $\lambda_{c_i, d_{c_i, j}}$.

Note that under the (vanilla) NDP, the skill parameter corresponding to $\lambda_{c_i, d_{c_i, j}}$ would be $\lambda_{c_i, d_{i,j}}$. Although subtle, the difference is that under the NDP each ranker would have their own entity clustering structure, that is, the (entity) cluster indicators within \mathbf{d}_i need not be the same as those within \mathbf{d}_j even if rankers i and j are within the same ranker cluster ($c_i = c_j$). The ANDP on the other hand requires that if two rankers are in the same cluster then they also have the equivalent entity clustering, whence the clustering structure for a ranker in cluster c_i is given by \mathbf{d}_{c_i} . Recall that this constraint is made so that entity clustering can be inferred as there is no information (about the entity clustering) contained within a single ranking and therefore we must use all the information from those rankers within cluster c to get information about \mathbf{d}_c .

We now describe how to generate ranked data from this model. We first need to specify both the ranker clustering structure and the entity clustering structure within each (active) ranker cluster. We could, of course, just choose these structures by giving values to the c_i and d_{ck} . Note that, as we are using a conditional sampling approach there is no longer a requirement for the active (entity or ranker) clusters to be labelled incrementally from 1 as our state space (over cluster indicators and the collection of unique skill parameters Λ) remains of fixed dimension, namely $N_1 \times N_2$. The cluster labels are therefore only required

to be chosen so that $c_i \in \{1, \dots, N_1\}$ and $d_{ck} \in \{1, \dots, N_2\}$. Alternatively we could draw cluster structures for both the rankers and entities from the ANDP prior distribution as follows.

- Choose N_1 and N_2 sufficiently large.
- Choose $\alpha > 0$ or sample from an appropriate prior distribution.
- Sample $v_s \stackrel{\text{indep}}{\sim} \text{Beta}(1, \alpha)$ for $s = 1, \dots, N_1 - 1$ and let $v_{N_1} = 1$.
- Set $\psi_s = v_s \prod_{\ell < s} (1 - v_\ell)$ for $s = 1, \dots, N_1$.
- Sample $c_i \stackrel{\text{indep}}{\sim} \text{Cat}(N_1, \psi)$ for $i = 1, \dots, n$.
- Choose $\gamma_s = \gamma > 0$ or sample γ_s (independently) from an appropriate prior distribution which is exchangeable with respect to s for $s \in \{\mathbf{c}\}$.
- Sample $u_{st} \stackrel{\text{indep}}{\sim} \text{Beta}(1, \gamma_s)$ for $s \in \{\mathbf{c}\}$, $t = 1, \dots, N_2 - 1$ and let $u_{sN_2} = 1$.
- Set $\omega_{st} = u_{st} \prod_{\ell < t} (1 - u_{s\ell})$ for $s \in \{\mathbf{c}\}$, $t = 1, \dots, N_2$.
- Sample $d_{sk} \stackrel{\text{indep}}{\sim} \text{Cat}(N_2, \omega_s)$ for $s \in \{\mathbf{c}\}$, $k = 1, \dots, K$.

Once we have a clustering structure of both the rankers and entities we need to choose values for the (cluster-specific) skill parameters λ_{sk} . Again these can either be chosen explicitly or drawn from the prior distribution by sampling $\lambda_{sk} \stackrel{\text{indep}}{\sim} G_0$ for $s \in \{\mathbf{c}\}$, $k \in \{\mathbf{d}_s\}$. Finally we need to specify whether rankers are informative or not and so must choose a value of the binary w_i or sample them independently from $\text{Bern}(p_i)$ distributions.

Now all the parameters of the model are given, we can use the exponential latent variable representation of the Weighted Plackett–Luce model to generate rankings. A collection of n complete rankings $\{\mathbf{x}_i\}_{i=1}^n$ is generated via the following process.

For $i = 1, \dots, n$

- Sample $\nu_{ij} \stackrel{\text{indep}}{\sim} \text{Exp}(\lambda_{c_i, d_{c_i, j}}^{w_i})$ for $j = 1, \dots, K$.
- Set $x_{ij} = \underset{q \in S_{ij}}{\text{argmin}} \nu_{iq}$ where $S_{ij} = \mathcal{K} \setminus \{x_{i1}, \dots, x_{ij-1}\}$ for $j = 1, \dots, K$.

Alternative types of rankings, such as a top–5 ranking, can be obtained from the complete rankings simulated using the same process as discussed in Section 2.2.5.

4.5.2 Prior specification and latent variables

Again we choose a Gamma prior distribution for the skill parameters so that a conjugate update can be performed (after data augmentation). As in Section 3.6 our model allows for entity clustering and so we need to choose a single base distribution G_0 for all skill parameters. This follows from the exchangeability requirement of the DP prior over the entity parameters as discussed previously. Here we take $G_0 = \text{Ga}(a, 1)$, giving $\lambda_{st} \stackrel{\text{indep}}{\sim} \text{Ga}(a, 1)$ *a priori*; recall that the rate parameter is not likelihood identifiable and so is fixed at one. Note that as we are using a conditional sampling approach, we must also choose truncation parameters to approximate the infinite dimensional aspect of the Dirichlet processes. The state space of the skill parameters therefore remains of fixed dimension $(N_1 \times N_2)$. Let $\Lambda = (\lambda_{st})$ denote the collection of all unique skill parameters ($s = 1, \dots, N_1$, $t = 1, \dots, N_2$). The prior distribution over the skill parameters is therefore

$$\pi(\Lambda) = \prod_{s=1}^{N_1} \prod_{t=1}^{N_2} \frac{\lambda_{st}^{a-1} e^{-\lambda_{st}}}{\Gamma(a)}.$$

We choose the prior distribution for w_i to be as before, with $w_i \stackrel{\text{indep}}{\sim} \text{Bern}(p_i)$, $p_i \in (0, 1]$ for $i = 1, \dots, n$. Recall that the prior choice $p_i = 0$ is not allowed. We could specify the concentration parameters α and γ_s as fixed constants but making specific choices can be difficult. Also the ANDP prior requires that these entity concentration parameters must be the same for all ranker clusters. Instead we choose to place a prior distribution over the concentration parameters for both ranker and entity clustering and let $\alpha \sim \text{Ga}(a_\alpha, b_\alpha)$ and $\gamma_s \stackrel{\text{indep}}{\sim} \text{Ga}(a_\gamma, b_\gamma)$ for $s = 1, \dots, N_1$ *a priori*.

We also introduce the latent cluster indicators from the previous section: recall that $c_i = j$ denotes that ranker i is associated with parameter vector $\boldsymbol{\lambda}_j$ and $d_{ij} = \ell$ denotes that entity j within parameter vector i is allocated to entity cluster ℓ . The value of the skill parameter assigned to entity j from ranking i is therefore given by $\lambda_{c_i, d_{c_i, j}}$. Using these latent variables, the likelihood is

$$\pi(\mathcal{D}|\Lambda, \mathbf{c}, D, \mathbf{w}) = \prod_{i=1}^n \prod_{j=1}^{n_i} \frac{\left(\lambda_{c_i, d_{c_i, x_{ij}}}\right)^{w_i}}{\sum_{m=j}^{n_i} \left(\lambda_{c_i, d_{c_i, x_{im}}}\right)^{w_i} + \sum_{m \in \mathcal{U}_i} \left(\lambda_{c_i, d_{c_i, m}}\right)^{w_i}}. \quad (4.4)$$

Unsurprisingly, given what we have seen previously, the form of the likelihood does not admit conjugate Bayesian inference. The implementation of a Gibbs sampler to maintain computational efficiency without the need for multiple tuning parameters is however highly desirable. To facilitate this we appeal to the same technique as for previous models, that

is, data augmentation. A Gibbs sampling solution can be obtained using a straightforward generalisation of the latent variables in Caron and Doucet (2012), namely

$$z_{ij} | \mathcal{D}, \Lambda, \mathbf{c}, D, \mathbf{w} \stackrel{\text{indep}}{\sim} \text{Exp} \left\{ \sum_{m=j}^{n_i} \left(\lambda_{c_i, d_{c_i, x_{im}}} \right)^{w_i} + \sum_{m \in \mathcal{U}_i} \left(\lambda_{c_i, d_{c_i, m}} \right)^{w_i} \right\} \quad (4.5)$$

for $i = 1, \dots, n$, $j = 1, \dots, n_i$.

Using these latent variables, the complete data likelihood is

$$\begin{aligned} \pi(\mathcal{D}, Z | \Lambda, \mathbf{c}, D, \mathbf{w}) &= \pi(\mathcal{D} | \Lambda, \mathbf{c}, D, \mathbf{w}) \pi(Z | \mathcal{D}, \Lambda, \mathbf{c}, D, \mathbf{w}) \\ &= \prod_{i=1}^n \prod_{j=1}^{n_i} \left(\lambda_{c_i, d_{c_i, x_{ij}}} \right)^{w_i} \exp \left\{ - \left(\sum_{m=j}^{n_i} \left(\lambda_{c_i, d_{c_i, x_{im}}} \right)^{w_i} + \sum_{m \in \mathcal{U}_i} \left(\lambda_{c_i, d_{c_i, m}} \right)^{w_i} \right) z_{ij} \right\} \\ &= \prod_{s=1}^{N_1} \prod_{t=1}^{N_2} \lambda_{st}^{\beta_{st}} \prod_{i=1}^n \prod_{j=1}^{n_i} \exp \left\{ - \left(\sum_{m=j}^{n_i} \left(\lambda_{c_i, d_{c_i, x_{im}}} \right)^{w_i} + \sum_{m \in \mathcal{U}_i} \left(\lambda_{c_i, d_{c_i, m}} \right)^{w_i} \right) z_{ij} \right\} \\ &= \prod_{s=1}^{N_1} \prod_{t=1}^{N_2} \lambda_{st}^{\beta_{st}} \exp \left(-\lambda_{st} \sum_{i=1}^n w_i \sum_{j=1}^{n_i} \zeta_{ij}(s, t) z_{ij} \right), \end{aligned} \quad (4.6)$$

where

$$\beta_{st} = \sum_{i=1}^n w_i \mathbb{I}(c_i = s) \sum_{j=1}^{n_i} \mathbb{I}(d_{c_i, x_{ij}} = t) \quad (4.7)$$

is the number of times that the random variable λ_{st} is assigned to an entity within an informative ranking, and

$$\zeta_{ij}(s, t) = \mathbb{I}(c_i = s) \left(\sum_{m=j}^{n_i} \mathbb{I}(d_{c_i, x_{im}} = t) + \sum_{m \in \mathcal{U}_i} \mathbb{I}(d_{c_i, m} = t) \right) \quad (4.8)$$

is the number of times that the random variable λ_{st} represents an entity within an informative ranking and is ranked no higher than j th in the i th ranking.

4.5.3 Full conditional distributions

Previously, when applying a marginal sampling approach, we used Neal's Algorithm 8 (Neal, 2000) to sample the latent indicators from their full conditional distributions. The full conditional distribution for the DP concentration parameter was also known; see Section 3.4.2. It follows that we could obtain the FCDs for the remaining parameters by considering the density of all (remaining) stochastic quantities conditional on the latent indicators and the concentration parameter. However, when implementing a conditional

sampling approach, we must derive the full conditional distributions for all of the unknown quantities (including the latent indicators and the concentration parameter). Here the posterior distribution of interest is $\pi(\Lambda, Z, \mathbf{v}, \mathbf{c}, u, D, \alpha, \boldsymbol{\gamma}, \mathbf{w} | \mathcal{D})$ and in the remainder of this section we derive the full conditional distributions for each of the unknown quantities. A complete outline of the MCMC scheme used to generate posterior realisations is described in Section 4.5.7.

We begin by constructing the density of *all* stochastic quantities as

$$\begin{aligned}
 & \pi(\Lambda, \mathcal{D}, Z, \mathbf{v}, \mathbf{c}, u, D, \alpha, \boldsymbol{\gamma}, \mathbf{w}) \\
 &= \pi(Z | \mathcal{D}, \Lambda, \mathbf{c}, D, \mathbf{w}) \pi(\mathcal{D} | \Lambda, \mathbf{c}, D, \mathbf{w}) \pi(D | u) \pi(u | \boldsymbol{\gamma}) \pi(\mathbf{c} | \mathbf{v}) \pi(\mathbf{v} | \alpha) \pi(\Lambda) \pi(\alpha) \pi(\boldsymbol{\gamma}) \pi(\mathbf{w}) \\
 &= \prod_{s=1}^{N_1} \prod_{t=1}^{N_2} \lambda_{st}^{\beta_{st}} \exp \left\{ -\lambda_{st} \sum_{i=1}^n w_i \sum_{j=1}^{n_i} \zeta_{ij}(s, t) z_{ij} \right\} \\
 &\quad \times \prod_{s=1}^{N_1} \prod_{t=1}^{N_2-1} \frac{\Gamma(1 + \gamma_s)}{\Gamma(1)\Gamma(\gamma_s)} (1 - u_{st})^{\gamma_s - 1} \times \prod_{s=1}^{N_1} \prod_{j=1}^K \omega_{s,d_{sj}} \\
 &\quad \times \prod_{s=1}^{N_1} \prod_{t=1}^{N_2} \frac{\lambda_{st}^{a-1} e^{-\lambda_{st}}}{\Gamma(a)} \times \prod_{s=1}^{N_1-1} \frac{\Gamma(1 + \alpha)}{\Gamma(1)\Gamma(\alpha)} (1 - v_s)^{\alpha - 1} \times \prod_{i=1}^n \psi_{c_i} \\
 &\quad \times \frac{b_\alpha^{a_\alpha}}{\Gamma(a_\alpha)} \alpha^{a_\alpha - 1} e^{-\alpha b_\alpha} \times \prod_{s=1}^{N_1} \frac{b_\gamma^{a_\gamma}}{\Gamma(a_\gamma)} \gamma_s^{a_\gamma - 1} e^{-\gamma_s b_\gamma} \times \prod_{i=1}^n p_i^{w_i} (1 - p_i)^{1 - w_i} \\
 &= \prod_{s=1}^{N_1} \prod_{t=1}^{N_2} \frac{\lambda_{st}^{a+\beta_{st}-1}}{\Gamma(a)} \exp \left\{ - \left(1 + \sum_{i=1}^n w_i \sum_{j=1}^{n_i} \zeta_{ij}(s, t) z_{ij} \right) \lambda_{st} \right\} \\
 &\quad \times \prod_{s=1}^{N_1} \prod_{t=1}^{N_2-1} \gamma_s (1 - u_{st})^{\gamma_s - 1} \times \prod_{s=1}^{N_1} \prod_{t=1}^{N_2} \left\{ u_{st} \prod_{\ell < t} (1 - u_{s\ell}) \right\}^{m_{st}} \\
 &\quad \times \prod_{s=1}^{N_1-1} \alpha (1 - v_s)^{\alpha - 1} \times \prod_{s=1}^{N_1} \left\{ v_s \prod_{\ell < s} (1 - v_\ell) \right\}^{m_s} \times \frac{b_\alpha^{a_\alpha}}{\Gamma(a_\alpha)} \alpha^{a_\alpha - 1} e^{-\alpha b_\alpha} \\
 &\quad \times \prod_{s=1}^{N_1} \frac{b_\gamma^{a_\gamma}}{\Gamma(a_\gamma)} \gamma_s^{a_\gamma - 1} e^{-\gamma_s b_\gamma} \times \prod_{i=1}^n p_i^{w_i} (1 - p_i)^{1 - w_i} \\
 &= \prod_{s=1}^{N_1} \prod_{t=1}^{N_2} \frac{\lambda_{st}^{a+\beta_{st}-1}}{\Gamma(a)} \exp \left\{ - \left(1 + \sum_{i=1}^n w_i \sum_{j=1}^{n_i} \zeta_{ij}(s, t) z_{ij} \right) \lambda_{st} \right\} \\
 &\quad \times \prod_{s=1}^{N_1-1} \alpha v_s^{m_s} (1 - v_s)^{\alpha - 1 + \sum_{\ell=s+1}^{N_1} m_\ell} \prod_{s=1}^{N_1} \prod_{t=1}^{N_2-1} \gamma_s u_{st}^{m_{st}} (1 - u_{st})^{\gamma_s - 1 + \sum_{\ell=t+1}^{N_2} m_{s\ell}} \\
 &\quad \times \frac{b_\alpha^{a_\alpha}}{\Gamma(a_\alpha)} \alpha^{a_\alpha - 1} e^{-\alpha b_\alpha} \times \prod_{s=1}^{N_1} \frac{b_\gamma^{a_\gamma}}{\Gamma(a_\gamma)} \gamma_s^{a_\gamma - 1} e^{-\gamma_s b_\gamma} \times \prod_{i=1}^n p_i^{w_i} (1 - p_i)^{1 - w_i}, \quad (4.9)
 \end{aligned}$$

where

$$m_s = \sum_{i=1}^n \mathbb{I}(c_i = s) \quad \text{and} \quad m_{st} = \sum_{j=1}^K \mathbb{I}(d_{sj} = t)$$

are the number of rankers assigned to ranker cluster s and the number of entities assigned to entity cluster t within ranker cluster s respectively.

The derivation of the full conditional distribution (FCD) for each random quantity within the model follows a similar procedure to that seen previously. Each full conditional distribution is obtained by taking the appropriate parts of (4.9). The FCDs are as follows.

- Λ : For $s = 1, \dots, N_1$, $t = 1, \dots, N_2$,

$$\lambda_{st} | \mathcal{D}, Z, \mathbf{v}, \mathbf{c}, u, D, \alpha, \boldsymbol{\gamma}, \mathbf{w} \stackrel{\text{indep}}{\sim} \text{Ga} \left(a + \beta_{st}, 1 + \sum_{i=1}^n w_i \sum_{j=1}^{n_i} \zeta_{ij}(s, t) z_{ij} \right). \quad (4.10)$$

- Z : As defined in (4.5) for $i = 1, \dots, n$, $j = 1, \dots, n_i$,

$$z_{ij} | \dots \stackrel{\text{indep}}{\sim} \text{Exp} \left\{ \sum_{m=j}^{n_i} \left(\lambda_{c_i, d_{c_i, x_{im}}} \right)^{w_i} + \sum_{m \in \tau_i} \left(\lambda_{c_i, d_{c_i, m}} \right)^{w_i} \right\}. \quad (4.11)$$

- \mathbf{v} : For $s = 1, \dots, N_1 - 1$,

$$v_s | \dots \stackrel{\text{indep}}{\sim} \text{Beta} \left(1 + m_s, \alpha + \sum_{\ell=s+1}^{N_1} m_\ell \right). \quad (4.12)$$

- U : For $s = 1, \dots, N_1$, $t = 1, \dots, N_2 - 1$,

$$u_{st} | \dots \stackrel{\text{indep}}{\sim} \text{Beta} \left(1 + m_{st}, \gamma_s + \sum_{\ell=t+1}^{N_2} m_{s\ell} \right). \quad (4.13)$$

- \mathbf{c} : For $i = 1, \dots, n$, c_i has a discrete distribution with probabilities given by

$$\begin{aligned} \Pr(c_i = s | \dots) &= \Pr(\mathbf{c} = \tilde{\mathbf{c}} | \dots) \\ &\propto \prod_{s=1}^{N_1} \left\{ v_s \prod_{\ell < s} (1 - v_1) \right\}^{m_s} \times \prod_{s=1}^{N_1} \prod_{t=1}^{N_2} \lambda_{st}^{\beta_{st}} \exp \left\{ -\lambda_{st} \sum_{i=1}^n w_i \sum_{j=1}^{n_i} \zeta_{ij}(s, t) z_{ij} \right\} \\ &= \prod_{i'=1}^n \psi_{\tilde{c}_{i'}} \times \prod_{i'=1}^n \prod_{j=1}^{n_i} \left(\lambda_{\tilde{c}_{i'}, d_{\tilde{c}_{i'}, x_{i'j}}} \right)^{w_{i'}} \\ &\quad \times \exp \left[-z_{i'j} \left\{ \sum_{m=j}^{n_{i'}} \left(\lambda_{\tilde{c}_{i'}, d_{\tilde{c}_{i'}, x_{i'm}}} \right)^{w_{i'}} + \sum_{m \in \mathcal{U}_{i'}} \left(\lambda_{c_{i'}, d_{c_{i'}, m}} \right)^{w_{i'}} \right\} \right] \end{aligned}$$

$$\propto \psi_{\tilde{c}_i} \prod_{j=1}^{n_i} \left(\lambda_{\tilde{c}_i, d_{\tilde{c}_i, x_{ij}}} \right)^{w_i} \exp \left[-z_{ij} \left\{ \sum_{m=j}^{n_i} \left(\lambda_{\tilde{c}_i, d_{\tilde{c}_i, x_{im}}} \right)^{w_i} + \sum_{m \in \mathcal{U}_i} \left(\lambda_{\tilde{c}_i, d_{\tilde{c}_i, m}} \right)^{w_i} \right\} \right] \quad (4.14)$$

where $1 \leq s \leq N_1$, and $\tilde{\mathbf{c}}_{i,s}$ has $\tilde{c}_j = c_j$ for $j \neq i$ and $\tilde{c}_i = s$. Note that this is simply the complete data likelihood for ranker i given that they are in (ranker) cluster s . Also note that if $w_i = 0$ then

$$\begin{aligned} \Pr(c_i = s | w_i = 0, \dots) &= \Pr(\mathbf{c} = \tilde{\mathbf{c}} | w_i = 0, \dots) \\ &\propto \psi_s \prod_{j=1}^{n_i} \left(\lambda_{s, d_{s, x_{ij}}} \right)^0 \exp \left[-z_{ij} \left\{ \sum_{m=j}^{n_i} \left(\lambda_{s, d_{s, x_{im}}} \right)^0 + \sum_{m \in \mathcal{U}_i} \left(\lambda_{s, d_{sm}} \right)^0 \right\} \right] \\ &= \psi_s \prod_{j=1}^{n_i} 1 \times \exp \left\{ - \sum_{j=1}^{n_i} z_{ij} \left(\sum_{m=j}^{n_i} 1 + \sum_{m \in \mathcal{U}_i} 1 \right) \right\} \\ &= \psi_s \prod_{j=1}^{n_i} \exp \{ -z_{ij}(K_i - j + 1) \} \\ &\propto \psi_s, \end{aligned}$$

and so when a ranker is deemed uninformative, the computational cost is reduced as there are no likelihood evaluations needed.

- D : For $s = 1, \dots, N_1$, $j = 1, \dots, K$, d_{sj} has the discrete distribution given by

$$\begin{aligned} \Pr(d_{sj} = t | \dots) &= \Pr(\mathbf{d}_{s \cdot} = \tilde{\mathbf{d}}_{st} | \dots) \\ &\propto \prod_{s'=1}^{N_1} \prod_{t=1}^{N_2} \left\{ u_{s't} \prod_{\ell < t} (1 - u_{s'\ell}) \right\}^{m_{s't}} \lambda_{s't}^{\beta_{s't}} \exp \left\{ -\lambda_{s't} \sum_{i=1}^n w_i \sum_{j=1}^{n_i} \zeta_{ij}(s', t) z_{ij} \right\} \\ &= \prod_{s'=1}^{N_1} \prod_{j'=1}^K \omega_{s', \tilde{d}_{s'j'}} \times \prod_{i=1}^n \prod_{j'=1}^{n_i} \left(\lambda_{c_i, \tilde{d}_{c_i, x_{ij'}}} \right)^{w_i} \\ &\quad \times \exp \left[-z_{ij'} \left\{ \sum_{m=j'}^{n_i} \left(\lambda_{c_i, \tilde{d}_{c_i, x_{im}}} \right)^{w_i} + \sum_{m \in \mathcal{U}_i} \left(\lambda_{c_i, \tilde{d}_{c_i, m}} \right)^{w_i} \right\} \right] \\ &\propto \omega_{st} \times \prod_{i \in F} \prod_{j'=1}^{n_i} \left(\lambda_{c_i, \tilde{d}_{c_i, x_{ij'}}} \right)^{w_i} \\ &\quad \times \exp \left[-z_{ij'} \left\{ \sum_{m=j'}^{n_i} \left(\lambda_{c_i, \tilde{d}_{c_i, x_{im}}} \right)^{w_i} + \sum_{m \in \mathcal{U}_i} \left(\lambda_{c_i, \tilde{d}_{c_i, m}} \right)^{w_i} \right\} \right], \quad (4.15) \end{aligned}$$

where $1 \leq t \leq N_2$, $F = \{i : c_i = s\}$ and \tilde{d}_{st} is given by $\tilde{d}_{s\ell} = d_{s\ell}$ for $\ell \neq j$ and $\tilde{d}_{sj} = t$. Note that for any ranker $q \in F$, if $w_q = 0$ then

$$\begin{aligned} & \prod_{j'=1}^{n_i} \left(\lambda_{c_i, \tilde{d}_{c_i, x_{ij'}}} \right)^{w_i} \exp \left[-z_{ij'} \left\{ \sum_{m=j'}^{n_i} \left(\lambda_{c_i, \tilde{d}_{c_i, x_{im}}} \right)^{w_i} + \sum_{m \in \mathcal{U}_i} \left(\lambda_{c_i, \tilde{d}_{c_i, m}} \right)^{w_i} \right\} \right] \\ &= \prod_{j'=1}^{n_i} \exp \left\{ -z_{ij'} (K_i - j' + 1) \right\} \end{aligned}$$

and is therefore constant with respect to changes in the value of d_{sj} . It follows that we can redefine the set as $F = \{i : (c_i = s) \cap (w_i = 1)\}$ which will help to reduce the computational burden.

- \mathbf{w} : For $i = 1, \dots, n$, w_i has the discrete distribution given by

$$\begin{aligned} & \Pr(w_i = 1 | \mathbf{w}_{-i}, \dots) \\ & \propto \Pr(w_i = 1) \pi(\mathcal{D} | \mathbf{w}_{-i}, w_i = 1, \dots) \pi(Z | \mathbf{w}_{-i}, w_i = 1, \dots) \\ & \propto p_i \prod_{i=1}^n \prod_{j=1}^{n_i} \left(\lambda_{c_i, d_{c_i, x_{ij}}} \right)^{w_i} \exp \left[-z_{ij} \left\{ \sum_{m=j}^{n_i} \left(\lambda_{c_i, d_{c_i, x_{im}}} \right)^{w_i} + \sum_{m \in \mathcal{U}_i} \left(\lambda_{c_i, d_{c_i, m}} \right)^{w_i} \right\} \right] \\ & \propto p_i \prod_{j=1}^{n_i} \lambda_{c_i, d_{c_i, x_{ij}}} \exp \left\{ -z_{ij} \left(\sum_{m=j}^{n_i} \lambda_{c_i, d_{c_i, x_{im}}} + \sum_{m \in \mathcal{U}_i} \lambda_{c_i, d_{c_i, m}} \right) \right\}. \\ & \Pr(w_i = 0 | \mathbf{w}_{-i}, \dots) \\ & \propto \Pr(w_i = 0) \pi(\mathcal{D} | \mathbf{w}_{-i}, w_i = 0, \dots) \pi(Z | \mathbf{w}_{-i}, w_i = 0, \dots) \\ & \propto (1 - p_i) \prod_{i=1}^n \prod_{j=1}^{n_i} \left(\lambda_{c_i, d_{c_i, x_{ij}}} \right)^{w_i} \exp \left[-z_{ij} \left\{ \sum_{m=j}^{n_i} \left(\lambda_{c_i, d_{c_i, x_{im}}} \right)^{w_i} + \sum_{m \in \mathcal{U}_i} \left(\lambda_{c_i, d_{c_i, m}} \right)^{w_i} \right\} \right] \\ & \propto (1 - p_i) \prod_{j=1}^{n_i} \exp \left\{ -z_{ij} \left(\sum_{m=j}^{n_i} 1 + \sum_{m \in \mathcal{U}_i} 1 \right) \right\} \\ & \propto (1 - p_i) \exp \left\{ - \sum_{j=1}^{n_i} z_{ij} (K_i - j + 1) \right\}. \end{aligned}$$

Therefore, for $i = 1, \dots, n$, the full conditional is

$$w_i | \dots \stackrel{\text{indep}}{\sim} \text{Bern}(\rho_i), \quad (4.16)$$

where

$$\rho_i = \frac{\Pr(w_i = 1 | \mathbf{w}_{-i}, \dots)}{\Pr(w_i = 1 | \mathbf{w}_{-i}, \dots) + \Pr(w_i = 0 | \mathbf{w}_{-i}, \dots)},$$

is the probability that ranking i is informative (given the other quantities).

- α : From (4.9) we have

$$\begin{aligned}\pi(\alpha | \dots) &= \frac{b_\alpha^{a_\alpha}}{\Gamma(a_\alpha)} \alpha^{a_\alpha-1} e^{-\alpha b_\alpha} \times \prod_{s=1}^{N_1-1} \alpha(1-v_s)^{\alpha-1} \\ &\propto \alpha^{a_\alpha+N_1-2} e^{-\alpha b_\alpha} \prod_{s=1}^{N_1-1} \exp\{(\alpha-1) \log(1-v_s)\} \\ &\propto \alpha^{a_\alpha+N_1-2} \exp\left\{-\alpha \left(b_\alpha - \sum_{s=1}^{N_1-1} \log(1-v_s)\right)\right\},\end{aligned}$$

whence

$$\alpha | \dots \sim \text{Ga}\left(a_\alpha + N_1 - 1, b_\alpha - \sum_{s=1}^{N_1-1} \log(1-v_s)\right) \quad (4.17)$$

as $b_\alpha - \sum_{s=1}^{N_1-1} \log(1-v_s) > 0$ since the $v_s \in (0, 1)$.

- γ : From (4.9) we have

$$\begin{aligned}\pi(\gamma | \dots) &= \prod_{s=1}^{N_1} \frac{b_\gamma^{a_\gamma}}{\Gamma(a_\gamma)} \gamma_s^{a_\gamma-1} e^{-\gamma_s b_\gamma} \times \prod_{s=1}^{N_1} \prod_{t=1}^{N_2-1} \gamma_s(1-u_{st})^{\gamma_s-1} \\ &\propto \prod_{s=1}^{N_1} \gamma_s^{a_\gamma+N_2-2} e^{-\gamma_s b_\gamma} \prod_{s=1}^{N_1} \prod_{t=1}^{N_2-1} \exp\{(\gamma_s-1) \log(1-u_{st})\} \\ &\propto \prod_{s=1}^{N_1} \gamma_s^{a_\gamma+N_2-2} \exp\left\{-\gamma_s \left(b_\gamma - \sum_{t=1}^{N_2-1} \log(1-u_{st})\right)\right\},\end{aligned}$$

whence for $s = 1, \dots, N_1$,

$$\gamma_s | \dots \stackrel{\text{indep}}{\sim} \text{Ga}\left(a_\gamma + N_2 - 1, b_\gamma - \sum_{t=1}^{N_2-1} \log(1-u_{st})\right) \quad (4.18)$$

as $b_\gamma - \sum_{t=1}^{N_2-1} \log(1-u_{st}) > 0$ since the $u_{st} \in (0, 1)$ for all $s = 1, \dots, N_1$.

We note in passing that, although perhaps counter-intuitive, the most computationally expensive FCDs to evaluate are the discrete distributions over the ranker and entity cluster labels \mathbf{c} and D . This follows as numerous likelihood calculations are needed (each involving many terms) in order to sample each cluster allocation, particularly for entity labels.

To generate realisations from the posterior distribution $\pi(\Lambda, Z, \mathbf{v}, \mathbf{c}, u, D, \alpha, \gamma, \mathbf{w} | \mathcal{D})$ we could employ a Gibbs sampling strategy and repeatedly sample from the FCDs given in (4.10–4.18) in turn. However, this strategy could lead to the Markov chain becoming stuck in a local mode (of the stationary distribution). This follows as the full conditional distributions in (4.14) and (4.15) allow for updating the (ranker and entity) cluster alloca-

tions, \mathbf{c} and D , using one-at-a-time updates. Therefore a large cluster containing numerous rankers or entities will have difficulty in changing its allocation variable. Initially this may not appear to be a significant issue as the likelihood (4.4) is invariant to permutations of the latent cluster indicator variables. However, if we consider the full posterior distribution $\pi(\Lambda, Z, \mathbf{v}, \mathbf{c}, u, D, \alpha, \gamma, \mathbf{w} | \mathcal{D})$, it becomes clear that the value of the density will be affected by changes to the cluster labels through the construction of the weights of the atoms (for both rankers and entities within each ranker cluster). The weights are defined to be stochastically decreasing and therefore the weights of the clusters with larger labels have smaller expectation. It follows that, for example, if we had two clusters occupied, we would rather them be labelled 1 and 2 as opposed to $N_1 - 2$ and $N_1 - 1$ as the first labelling has an increased value of posterior density.

Thankfully it is possible to overcome this issue by introducing appropriate label switching steps which improve the mixing of the chain over the latent cluster indicators. We now describe these moves before outlining a full MCMC scheme that can be used to generate posterior realisations in Section 4.5.7.

4.5.4 Label switching moves

Papaspiliopoulos and Roberts (2008) and more recently Hastie et al. (2015) noted that when using a conditional sampling scheme for Dirichlet process mixture models (derived from the stick-breaking construction), the Markov chain can suffer from poor mixing of the cluster allocations. Initially this does not appear to be much of a concern as the likelihood (4.4) is invariant to permutations of the cluster labels. However, the full posterior distribution $\pi(\Lambda, Z, \mathbf{v}, \mathbf{c}, u, D, \alpha, \gamma, \mathbf{w} | \mathcal{D})$ is affected by changes to the labels. In particular, inferences on the concentration parameters α and γ can be significantly affected if the Markov chain fails to mix well over cluster labels; see Hastie et al. (2015) for details. The poor mixing of cluster labels is an issue for both ranker cluster allocations \mathbf{c} and the entity cluster allocations in each row of D .

As the weights of the atoms within each Dirichlet process are stochastically decreasing, the cluster allocation with highest posterior support is the one where the largest cluster has label 1, the second largest is labelled 2, and so on. However, our proposed Gibbs sampling scheme performs one-at-a-time updates to the cluster allocations. This results in cluster allocations swapping very rarely and could therefore be overly influenced by the (possibly random) initialisation of the sampler. To improve mixing, Papaspiliopoulos and Roberts (2008) proposed two label switching moves, here called Swap 1 and 2. In addition it has recently been noted by Hastie et al. (2015) that although these two swaps encourage movement to the cluster allocation of highest posterior support, once this state is reached the mixing becomes poor. They propose a further label switching move, which

we call Swap 3. All three of these swaps are accepted or rejected though Metropolis-Hastings proposals. In the following section we describe how these label switching proposal moves are implemented within our model setting. As our model contains nested Dirichlet processes we pay particular attention to the required state space swaps within the Markov chain when a label switching move is accepted.

4.5.5 Ranker allocations

We begin by considering proposed swaps to the latent ranker cluster allocations. Due to the nested nature of the Dirichlet processes, if we change a ranker cluster label we also need to change the labels of all parameters corresponding to the low level DP associated with the ranker clusters. We now describe the required state space swaps along with their proposal mechanisms.

Swap 1

Recalling that $m_s = \sum_{i=1}^n \mathbb{I}(c_i = s)$ denotes the number of rankers assigned to ranker cluster s we let $C^{(\text{al})} = \{s : m_s > 0, 1 \leq s \leq N_1\}$ be the set of ranker cluster allocation labels for those clusters which are populated, that is, the labels of clusters to which one or more rankers are assigned. This label switching step proposes to swap the labels of two random “alive” clusters $j, l \in C^{(\text{al})}$ and is accepted with probability

$$\min \left\{ 1, \left(\frac{\psi_j}{\psi_l} \right)^{m_l - m_j} \right\}.$$

If the swap is accepted then we need to make a number of changes to the state space of the Markov chain. Of course, we must swap the ranker cluster labels within the allocation vector \mathbf{c} and also the skill parameters associated with each ranker cluster, that is, swap the appropriate rows of Λ . However, in order to preserve the entity clustering within each ranker cluster we must also swap the corresponding rows of D , u and ω , that is, swap the low level Dirichlet processes associated with each ranker cluster. Finally we must swap γ_j with γ_l as the unique concentration parameter for each of the low level DPs must also be preserved. The details of the changes required are given by (4.19)–(4.23) below.

Firstly swap the labels in the ranker allocation vector \mathbf{c} :

$$c'_i = \begin{cases} l & i : c_i = j, \\ j & i : c_i = l, \\ c_i & \text{otherwise.} \end{cases} \quad (4.19)$$

Secondly swap the rows in the parameter matrix Λ :

$$\boldsymbol{\lambda}'_{i \cdot} = \begin{cases} \boldsymbol{\lambda}_l & i = j, \\ \boldsymbol{\lambda}_j & i = l, \\ \boldsymbol{\lambda}_i & \text{otherwise.} \end{cases} \quad (4.20)$$

Also, to maintain the entity clustering structure (within the ranking clusters) the rows in the entity clustering allocation matrix D are swapped:

$$\mathbf{d}'_{i \cdot} = \begin{cases} \mathbf{d}_l & i = j, \\ \mathbf{d}_j & i = l, \\ \mathbf{d}_i & \text{otherwise.} \end{cases} \quad (4.21)$$

Recall that we also need to swap the Dirichlet process associated with each of the ranker clusters. This requires the swapping of the rows within the matrix u and recalculating the weights ω for each (ranker) cluster. However, as the values of u (on which the values of ω are based) do not change it is more convenient to simply relabel these too:

$$\mathbf{u}'_{i \cdot} = \begin{cases} \mathbf{u}_l & i = j, \\ \mathbf{u}_j & i = l, \\ \mathbf{u}_i & \text{otherwise.} \end{cases} \quad \boldsymbol{\omega}'_{i \cdot} = \begin{cases} \boldsymbol{\omega}_l & i = j, \\ \boldsymbol{\omega}_j & i = l, \\ \boldsymbol{\omega}_i & \text{otherwise.} \end{cases} \quad (4.22)$$

Finally we also change the value of the concentration parameters for the DPs corresponding to clusters j and l :

$$\gamma'_i = \begin{cases} \gamma_l & i = j, \\ \gamma_j & i = l, \\ \gamma_i & \text{otherwise.} \end{cases} \quad (4.23)$$

The state of the Markov chain is then updated by letting $\mathbf{c} \rightarrow \mathbf{c}'$, $\Lambda \rightarrow \Lambda'$, $D \rightarrow D'$, $u \rightarrow u'$, $\omega \rightarrow \omega'$ and $\gamma \rightarrow \gamma'$.

Swap 2

The second label switching move we consider was also derived by Papaspiliopoulos and Roberts (2008) and proposes to swap the labels of two neighbouring ranker clusters, regardless of whether they are occupied or not. First we randomly sample a ranker cluster label $j \in \{1, \dots, N_1 - 1\}$ and then let $l = j + 1$. The cluster allocation variables j and l

are swapped with probability

$$\min \left\{ 1, \frac{(1 - v_l)^{m_j}}{(1 - v_j)^{m_l}} \right\}.$$

If accepted, the state space of the Markov chain needs to be changed in the same way as for Swap 1, that is, by the swaps in (4.19)–(4.23). In addition, for this proposal, if the swap is accepted we also swap the corresponding values of the beta random variables v associated with the ranker clusters:

$$v'_i = \begin{cases} v_l & i = j, \\ v_j & i = l, \\ v_i & \text{otherwise.} \end{cases}$$

We must also now recalculate the “weights” for each ranker cluster by recomputing ψ . It is not possible to simply relabel the corresponding weights due to their construction.

Swap 3

The final label switching proposal we implement is that of Hastie et al. (2015); this also proposes swapping the ranker cluster labels of two neighbouring clusters. Let $C^* = \max_{1 \leq i \leq n} c_i$ be the largest label of a ranker cluster which is occupied. We then sample a random cluster label $j \in \{1, \dots, C^* - 1\}$ and let $l = j + 1$. Before giving the acceptance probability it is useful to define the following terms:

$$\begin{aligned} E_1 &= \frac{\mathbb{E}(\psi_j | \mathbf{c}', \alpha)}{\mathbb{E}(\psi_l | \mathbf{c}, \alpha)} = \frac{1 + \alpha + m_l + \sum_{q>l} m_q}{\alpha + m_l + \sum_{q>l} m_q}, \\ E_2 &= \frac{\mathbb{E}(\psi_l | \mathbf{c}', \alpha)}{\mathbb{E}(\psi_j | \mathbf{c}, \alpha)} = \frac{\alpha + m_j + \sum_{q>l} m_q}{1 + \alpha + m_j + \sum_{q>l} m_q}, \\ \psi^+ &= \psi_j + \psi_l, \\ \hat{\psi} &= \psi_l E_1 + \psi_j E_2. \end{aligned}$$

The ranker cluster labels j and l are then swapped with probability

$$\min \left\{ 1, \left(\frac{\psi^+}{\hat{\psi}} \right)^{m_j + m_l} E_1^{m_l} E_2^{m_j} \right\}.$$

Again, if accepted, the state space swaps given in (4.19)–(4.23) must be applied. Further we also need to update the weights ψ (for the ranker clusters) and values for the beta

random variables v :

$$\psi'_i = \begin{cases} \frac{\psi_l \psi^+ E_1}{\hat{\psi}} & i = j, \\ \frac{\psi_j \psi^+ E_2}{\hat{\psi}} & i = l, \\ \psi_i & \text{otherwise.} \end{cases} \quad v'_i = \begin{cases} \frac{\psi'_j}{\prod_{q < j} (1 - v_q)} & i = j, \\ \frac{\psi'_l}{(1 - v'_j) \prod_{q < j} (1 - v_q)} & i = l, \\ v_i & \text{otherwise.} \end{cases}$$

4.5.6 Entity allocations

We have found that the proposed cluster label swaps outlined in the previous section are sufficient to ensure adequate mixing over the ranker cluster labels. For this model however we also need to ensure adequate mixing over the latent entity cluster variables within each of the ranker groups. We do this by using the same three swapping mechanisms as for the ranker allocations. However each of these swaps needs to be performed within each of the N_1 low level Dirichlet processes. The changes required to the state space of our Markov chain are somewhat more straightforward for the entity labels as we are dealing with a typical Dirichlet process (whose atoms are scalars) and therefore the details (given below) are more similar to those described in Papaspiliopoulos and Roberts (2008) and Hastie et al. (2015).

Swap 1

Recall first that $m_{st} = \sum_{j=1}^K \mathbb{I}(d_{sj} = t)$ denotes the number of entities assigned to entity cluster t within ranker cluster s . For $s = 1, \dots, N_1$, let $D_s^{(\text{al})} = \{t : m_{st} > 0, 1 \leq t \leq N_2\}$ be the set of entity cluster labels which are populated within Dirichlet process s . Note that there is no requirement for a ranker to be assigned to ranker cluster s and we only consider the entity clusters here. This label switching step proposes to swap the labels of two random alive clusters $j, l \in D_s^{(\text{al})}$. This swap is accepted with probability

$$\min \left\{ 1, \left(\frac{\omega_{sj}}{\omega_{sl}} \right)^{m_{sl} - m_{sj}} \right\}.$$

If the proposal is accepted then we need only make two changes to the state space of our Markov chain. We must, of course, swap the entity cluster labels within row s of the entity cluster label matrix D but also must swap the skill parameter values for entities j and l .

Formally we swap the labels in row s of our entity allocation matrix, D , by letting

$$d'_{si} = \begin{cases} l & i : d_{si} = j, \\ j & i : d_{si} = l, \\ d_{si} & \text{otherwise.} \end{cases} \quad (4.24)$$

We then swap the skill parameters for the entity clusters within row s of Λ by letting

$$\lambda'^\dagger_{si} = \begin{cases} \lambda_{sl}^\dagger & i = j, \\ \lambda_{sj}^\dagger & i = l, \\ \lambda_{si}^\dagger & \text{otherwise.} \end{cases} \quad (4.25)$$

The state of the Markov chain is then updated by letting $\Lambda \rightarrow \Lambda'$ and $D \rightarrow D'$. All other quantities remain unchanged.

Swap 2

Recall that for this swap changes are made to the labels of neighbouring clusters whether they are occupied or not. For $s = 1, \dots, N_1$, we randomly sample an entity cluster label $j \in \{1, \dots, N_2 - 1\}$ and let $l = j + 1$. The entity cluster labels j and l are swapped with probability

$$\min \left\{ 1, \frac{(1 - u_{sl})^{m_{sj}}}{(1 - u_{sj})^{m_{sl}}} \right\}.$$

If accepted we update the state space of our Markov chain as in (4.24) and (4.25). Recall that when applying this move for the ranker labels we also swapped the corresponding values of the beta random variables for these two clusters. The equivalent swap here is done within row s of the matrix u and we let

$$u'_{si} = \begin{cases} u_{sl} & i = j, \\ u_{sj} & i = l, \\ u_{si} & \text{otherwise.} \end{cases}$$

Note that the entity cluster weights within DP s (ω_s) must also be recalculated given their (inherent) dependence on the values of u . These can not simply be swapped due to their construction.

Swap 3

Finally we implement the label switching move described by Hastie et al. (2015). For $s = 1, \dots, N_1$ let $D_s^* = \max_{1 \leq j \leq K} d_{sj}$ be the largest label of an active entity cluster within DP s . We sample a random cluster label $j \in \{1, \dots, D_s^* - 1\}$, set $l = j + 1$ and calculate

$$\begin{aligned} E_1 &= \frac{\mathbb{E}(\omega_{sj}|d', \gamma_s)}{\mathbb{E}(\omega_{sl}|d, \gamma_s)} = \frac{1 + \gamma_s + m_{sl} + \sum_{q>l} m_{sq}}{\gamma_s + m_{sl} + \sum_{q>l} m_{sq}}, \\ E_2 &= \frac{\mathbb{E}(\omega_{sj}|d', \gamma_s)}{\mathbb{E}(\omega_{sl}|d, \gamma_s)} = \frac{\gamma_s + m_{sj} + \sum_{q>l} m_{sq}}{1 + \gamma_s + m_{sj} + \sum_{q>l} m_{sq}}, \\ \omega^+ &= \omega_{sj} + \omega_{sl}, \\ \hat{\omega} &= \omega_{sl}E_1 + \omega_{sj}E_2. \end{aligned}$$

The acceptance probability for this proposal is

$$\min \left\{ 1, \left(\frac{\omega^+}{\hat{\omega}} \right)^{m_{sj} + m_{sl}} E_1^{m_{sl}} E_2^{m_{sj}} \right\}.$$

Again, if accepted, the state space swaps given in (4.24) and (4.25) must be applied. Further we must update the weights ω and the corresponding values for the beta random variables u :

$$\omega'_{si} = \begin{cases} \frac{\omega_{sl}\omega^+ E_1}{\hat{\omega}} & i = j, \\ \frac{\omega_{sj}\omega^+ E_2}{\hat{\omega}} & i = l, \\ \omega_{si} & \text{otherwise.} \end{cases} \quad u'_{si} = \begin{cases} \frac{\omega'_{sj}}{\prod_{q<j} (1 - u_{sq})} & i = j, \\ \frac{\omega'_{sl}}{(1 - u'_{sj}) \prod_{q<j} (1 - u_{sq})} & i = l, \\ u_{si} & \text{otherwise.} \end{cases}$$

4.5.7 MCMC – a conditional sampler

We now describe an algorithm for generating samples from the posterior distribution $\pi(\Lambda, Z, \mathbf{v}, \mathbf{c}, u, D, \alpha, \boldsymbol{\gamma}, \mathbf{w} | \mathcal{D})$. In Section 4.5.3 we derived a complete set of full conditional distributions for each random quantity of interest so that we could use a Gibbs sampling scheme by repeatedly sampling from these distributions. However, as discussed, there are problems with mixing over the latent cluster indicator variables \mathbf{c} and D . To remedy this issue we employ additional label switching moves with proposed changes assessed through Metropolis-Hastings steps as described in the previous section. The algorithm we use is therefore a Metropolis-within-Gibbs sampler.

Given a suitable choice of truncation parameters, N_1 and N_2 , the posterior samples can be generated by repeatedly performing the following steps.

- Sample λ_{st} from (4.10) for $s = 1, \dots, N_1$, $t = 1, \dots, N_2$.
- Sample z_{ij} from (4.11) for $i = 1, \dots, n$, $j = 1, \dots, n_i$.
- Sample w_i from (4.16) for $i = 1, \dots, n$.
- Sample v_s from (4.12) for $s = 1, \dots, N_1 - 1$, and let $v_{N_1} = 1$
- Compute ψ , that is, let $\psi_s = v_s \prod_{\ell < s} (1 - v_\ell)$ for $s = 1, \dots, N_1$.
- Sample c_i from (4.14) for $i = 1, \dots, n$.
- Sample u_{st} from (4.13) for $s = 1, \dots, N_1$, $t = 1, \dots, N_2 - 1$, and let $u_{sN_2} = 1$.
- Compute ω , that is, let $\omega_{st} = u_{st} \prod_{\ell < t} (1 - u_{s\ell})$ for $s = 1, \dots, N_1$, $t = 1, \dots, N_2$.
- Sample d_{sj} from (4.15) for $s = 1, \dots, N_1$, $j = 1, \dots, K$.
- Sample α from (4.17).
- Sample γ_s from (4.18) for $s = 1, \dots, N_1$.
- Propose ranker label swaps as in Section 4.5.5.
- Propose entity label swaps (for $s = 1, \dots, N_1$) as in Section 4.5.6.
- Rescale
 - Sample $\Lambda^\dagger \sim \text{Ga}(N_1 N_2 a, 1)$.
 - Calculate $\Sigma = \sum_{s=1}^{N_2} \sum_{t=1}^{N_2} \lambda_{st}$.
 - For $s = 1, \dots, N_1$, $t = 1, \dots, N_2$, let $\lambda_{st} \rightarrow \lambda_{st} \Lambda^\dagger / \Sigma$.

A note on the choice of truncation parameters

As the algorithm employs truncation (using N_1 and N_2) any posterior samples generated are from an approximation of the true posterior. The level of approximation depends on the choice of truncation parameters N_1 and N_2 , with the samples increasingly being from the true posterior as $N_1, N_2 \rightarrow \infty$. The choice of N_1 and N_2 *a priori* is somewhat difficult as the level of truncation required heavily depends on the value of the concentration parameters, α and γ . The strategy we advocate is to perform a few pilot runs of the

MCMC scheme to gauge plausible values of the concentration parameters. The truncation parameters will then need to be changed so that the conditions in (4.3) hold. Of course, it would be helpful to make the truncation parameters as large as possible but increasing N_1 and N_2 has a major effect on the computational burden and can lead to considerable redundant prior sampling. The choice of truncation is therefore rather situation specific and perhaps limited by the computational resources available to the analyst.

4.5.8 A brief summary

So far this chapter has outlined the Adapted Nested Dirichlet process prior (ANDP) distribution and explored some of its features through simulation. We then described the Weighted Adapted Nested Dirichlet Process mixture of Plackett–Luce models (WAND). This took the ANDP as the prior distribution and used it to mix over Weighted Plackett–Luce models. Using the stick-breaking representation of the ANDP prior and truncating the infinite dimensional aspect we were able to derive a complete set of full conditional distributions. Additional label switching moves were then introduced to improve mixing over cluster labels. The resulting algorithm was a Metropolis-within-Gibbs sampling scheme. Adopting the (conditional) sampling approach outlined here does however have some drawbacks, the most notable of which is that we can only obtain samples from an approximate posterior distribution. A further issue is that it is difficult to determine what *a priori* choices of the truncation parameters lead to a reasonable approximation to the true posterior. Ideally, as for the one-way clustering in Chapter 3, we would avoid truncation methods and appeal to a marginal approach to inference. In the section that follows we discuss how to improve the MCMC scheme to avoid such approximations by implementing a two-way marginal sampling scheme.

4.6 A marginal sampling approach

Implementing marginal sampling methods for two-way clustering models which use Dirichlet processes can be somewhat challenging. For example, the construction of the standard NDP results in a “fully” marginal scheme being computationally infeasible (Rodriguez et al., 2008). The real crux when designing such a marginal sampling scheme is obtaining (posterior) realisations of the top level cluster labels, \mathbf{c} . For the standard Nested Dirichlet process, sampling c_i requires the evaluation of $\pi(\mathbf{x}_i|G_s)$ (for all s), that is, the likelihood of observation \mathbf{x}_i given it is in cluster s . The issue here is that G_s is itself a Dirichlet process (recall that under the NDP the atoms of the top level Dirichlet process are themselves Dirichlet processes). It follows that obtaining a value of $\pi(\mathbf{x}_i|G_s)$ requires the evaluation of an infinite sum – something which is clearly problematic. Given this, it is perhaps now

clear that a “single truncation” sampling scheme could be designed to obtain posterior realisations under the NDP prior. If the low level Dirichlet processes G_s are truncated at a *finite* value, say N_2 , then the evaluation of $\pi(\mathbf{x}_i|G_s)$ is straightforward. The resulting posterior sampling scheme would then comprise a marginal scheme (based on a Pólya urn) to sample the top level cluster indicators \mathbf{c} , and a conditional sampling approach to approximate the posterior over the low level indicators D ; see Rodriguez (2007) for full details. Such a sampling scheme would not only increase computational efficiency but also reduce the approximation to the posterior distribution in comparison to a “double truncation” approach akin to that discussed in Section 4.5. Further, the additional label switching moves would no longer be required for the ranker cluster labels.

Given the Adapted Nested Dirichlet process is inherently related to the NDP it follows that we too could implement a single truncation approach. It transpires, however, that a fully marginal approach is possible as a result of the adaptation made to the prior distribution. Recall that for the NDP, unless the low level DPs are truncated, a marginal approach to sample the indicator variables c_i is infeasible due to the need to evaluate $\pi(\mathbf{x}_i|G_s)$. However, for the ANDP we are instead required to evaluate $\pi(\mathbf{x}_i|\boldsymbol{\lambda}_s)$ where $\boldsymbol{\lambda}_s \sim G_s$, that is, $\boldsymbol{\lambda}_s$ is a *realisation* from a Dirichlet process and not a DP itself (as it is for the NDP). In other words, the subtle difference which allows for marginal schemes under the ANDP is that the top level Dirichlet process is fairly standard, with the atoms being parameter vectors which are *realisations* from (independent) DPs, whereas, for the NDP, the atoms of the top level DP are themselves (infinite dimensional) discrete distributions. The evaluation of $\pi(\mathbf{x}_i|\boldsymbol{\lambda}_s)$ is therefore trivial – it is simply the (Plackett–Luce) likelihood of ranking i conditional on the skill parameters for ranker cluster s .

In what follows we develop a marginal sampling algorithm which can be used to generate posterior samples under the Bayesian WAND model. Although the conditional sampling algorithm outlined in Section 4.5.7 is capable of generating approximate posterior samples (subject to the addition of label switching moves) it is still advantageous to appeal to a marginal approach – not least to reduce the computational burden when performing inference but also so that we can generate realisations from the true posterior distribution. As discussed in Section 3.4.1, marginal sampling algorithms typically involve the use of a Pólya urn scheme which enables the marginalization of the infinite dimensional distribution (Escobar and West (1995), MacEachern and Müller (1998)) and thus avoid approximations. We discussed in Section 4.5 how conditional samplers rely on truncation parameters to define the maximum size of the state space (N_1, N_2 in our case). When performing inference this entire state space needs to be updated at each iteration and causes a substantial amount of redundant prior sampling. To produce our marginal sampler we again appeal to Neal’s Algorithm 8 (Neal, 2000). Recall that this algorithm only performs updates for the unique components (clusters) which are currently populated and the re-

maining (theoretically infinite) unpopulated clusters are not considered to be within the sample space. Each observation is then assigned to either a component which is currently in use or to one of m *auxiliary* components which are independent draws from the prior distribution. Our model has nested clustering and so under the conditional method, the state space for the skill parameters Λ is dimension $N_1 \times N_2$. If we let N^r and N_s^e be the number of ranker clusters and the number of entity clusters within ranker cluster s respectively (for $s = 1, \dots, N^r$) then the size of our state space is reduced to $\sum_{s=1}^{N^r} N_s^e \ll N_1 \times N_2$ under marginal methods. It is clear therefore that marginal methods have the potential to reduce the number of operations required per iteration quite substantially. Neal's Algorithm 8 is designed to sample from a single DP mixture and so we design a nested version which will enable inference to be performed under the Bayesian WAND model.

Recall that the WAND model comprises an infinite mixture of Weighed Plackett–Luce models with the ANDP chosen as the prior distribution. As in Section 4.4, the main components of the model are

$$\begin{aligned} \mathbf{X}_i | \boldsymbol{\lambda}_i, w_i &\sim \text{PLW}(\boldsymbol{\lambda}_i, w_i), \quad i = 1, \dots, n, \\ (\boldsymbol{\lambda}_1, \dots, \boldsymbol{\lambda}_n) | Q &\sim Q \\ Q | \alpha, \gamma, G_0 &\sim \text{ANDP}(\alpha, \gamma, G_0), \end{aligned}$$

where the stick-breaking representation of the ANDP prior is as in Section 4.3.

4.6.1 Sampling marginally from the ANDP prior

We begin by first outlining how we can (marginally) simulate a realisation from the ANDP prior distribution via a Pólya urn scheme. To outline this process concisely we make use of the latent cluster indicators introduced in Section 4.5.1. Recall that $c_i = j$ denotes that ranking i is associated with parameter vector $\boldsymbol{\lambda}_j$ and $\mathbf{c} = (c_1, \dots, c_n)$ is defined to be the collection of these latent ranker cluster labels. For our entity clustering we let $d_{ij} = \ell$ denote that entity j within parameter vector i is allocated to entity cluster ℓ and D be the collection of all latent entity cluster labels. Note that, unlike for the conditional sampling approach, under the marginal sampling approach the cluster labels are required to be labelled incrementally from 1, that is, we require $c_i \in \{1, \dots, N^r\}$, (for $i = 1, \dots, n$) where N^r denotes the number of ranker clusters and also $d_{sj} \in \{1, \dots, N_s^e\}$, (for $j = 1, \dots, K$) where N_s^e denotes the number of entity clusters within ranker cluster $s \in \{\mathbf{c}\}$. A prior realisation from the ANDP is obtained using the following process.

- Choose $\alpha > 0$ or sample from an appropriate prior distribution.
- Set $c_1 = 1$ and the (current) number of ranker clusters $N^r = 1$.

- For $i = 2, \dots, n$ simulate the allocation of ranker i to a ranker cluster according to

$$\Pr(c_i = j|c_1, \dots, c_{i-1}) = \frac{n_{ij}^r}{\alpha + i - 1}, \quad \text{for } j = 1, \dots, N^r,$$

$$\Pr(c_i = N^r + 1|c_1, \dots, c_{i-1}) = \frac{\alpha}{\alpha + i - 1},$$

where n_{ij}^r denotes the number of rankers currently within ranker cluster j (at iteration i), and $N^r \rightarrow N^r + 1$ if $c_i = N^r + 1$.

- For each ranker cluster $s = 1, \dots, N^r$
 - Choose $\gamma = \gamma_s > 0$ for all s or sample from an appropriate prior distribution (note the choice of prior distribution *must* be exchangeable with respect to s).
 - Set $d_{s1} = 1$ and the current number of entity clusters within ranking cluster s $N_s^e = 1$.
 - For $k = 2, \dots, K$ simulate the allocation of entity k to an entity cluster according to

$$\Pr(d_{sk} = j|d_{s1}, \dots, d_{s,k-1}) = \frac{n_{kj,s}^e}{\gamma_s + k - 1}, \quad \text{for } j = 1, \dots, N_s^e,$$

$$\Pr(d_{sk} = N_s^e + 1|d_{s1}, \dots, d_{s,k-1}) = \frac{\gamma_s}{\gamma_s + k - 1},$$

where $n_{kj,s}^e$ denotes the number of entities currently within entity cluster j (at iteration k in ranker cluster s), and $N_s^e \rightarrow N_s^e + 1$ if $d_{si} = N_s^e + 1$.

- Simulate $\lambda_{sj} \stackrel{\text{indep}}{\sim} G_0$ for $s = 1, \dots, N^r$, $j = 1, \dots, N_s^e$.

4.6.2 Simulating data from the WAND model

In Section 4.5.1 we outlined how data can be generated from the WAND model using a conditional sampling approach. The mechanism for the marginal approach is almost identical and differs only in how we generate a realisation (of the cluster structure) from the ANDP prior. It follows that if the clustering structure is chosen explicitly, that is, it is not drawn from the ANDP prior, then the data generating process is the same as that outlined in Section 4.5.1. However, if we instead wish to sample a cluster structure from the prior distribution then using the marginal method given in Section 4.6.1 is advantageous as it allows the simulation of *exact* realisations from the prior distribution (in contrast to the approximation provided by the conditional sampling approach). Conditional on a prior sample (generated using the method in Section 4.6.1) a collection of n complete rankings of K entities is generated using the same method as in Section 4.5.1:

For $i = 1, \dots, n$

- Sample $\nu_{ij} \stackrel{\text{indep}}{\sim} \text{Exp}(\lambda_{c_i, d_{c_i, j}}^{w_i})$ for $j = 1, \dots, K$.
- Set $x_{ij} = \underset{q \in S_{ij}}{\text{argmin}} \nu_{iq}$ where $S_{ij} = \mathcal{K} \setminus \{x_{i1}, \dots, x_{ij-1}\}$ for $j = 1, \dots, K$.

Alternative types of rankings for example, a top-5 ranking, can be obtained from the complete rankings simulated using the same process as discussed in Section 2.2.5.

4.6.3 Prior specification and latent variables

Before we perform Bayesian inference we must first of course choose a suitable prior specification for our model. We consider the same prior specification as when we considered the conditional sampling approach in Section 4.5.2. We let $G_0 = \text{Ga}(a, 1)$ which gives $\lambda_{st} \stackrel{\text{indep}}{\sim} \text{Ga}(a, 1)$ *a priori* (recall that the rate parameter is not likelihood identifiable and therefore chosen to be 1). Again we define Λ to be the collection of all unique skill parameters and note that the dimension of the parameter space (over Λ) is $\sum_{s=1}^{N^r} N_s^e$ and therefore is now dependent on the number of both ranker and entity clusters. It follows that the prior distribution over Λ is

$$\pi(\Lambda | \mathbf{c}, D) = \prod_{s=1}^{N^r} \prod_{t=1}^{N_s^e} \frac{\lambda_{st}^{a-1} e^{-\lambda_{st}}}{\Gamma(a)}.$$

Furthermore we choose the prior on the latent binary ability indicators to be $w_i \stackrel{\text{indep}}{\sim} \text{Bern}(p_i)$ with $p_i \in (0, 1]$ for $i = 1, \dots, n$. Recall that the prior choice of $p_i = 0$ is not allowed; in this scenario ranker i has constant likelihood and therefore this ranker has no information about the parameters and so should not be considered. The prior distributions on the concentration parameters are $\alpha \sim \text{Ga}(a_\alpha, b_\alpha)$ and $\gamma_s \stackrel{\text{indep}}{\sim} \text{Ga}(a_\gamma, b_\gamma)$ for $s = 1, \dots, N^r$.

Of course, the likelihood under the Bayesian WAND model is the same irrespective of the sampling method used. However, we note that when augmenting the likelihood it is helpful for the distribution of the skill parameters to be of dimension $\sum_{s=1}^{N^r} N_s^e$ rather than of dimension $N_1 \times N_2$ as in the conditional approach. The likelihood, conditional on the latent cluster indicators, is

$$\pi(\mathcal{D} | \Lambda, \mathbf{c}, D, \mathbf{w}) = \prod_{i=1}^n \prod_{j=1}^{n_i} \frac{\left(\lambda_{c_i, d_{c_i, x_{ij}}} \right)^{w_i}}{\sum_{m=j}^{n_i} \left(\lambda_{c_i, d_{c_i, x_{im}}} \right)^{w_i} + \sum_{m \in \mathcal{U}_i} \left(\lambda_{c_i, d_{c_i, m}} \right)^{w_i}}. \quad (4.26)$$

Again it is useful to use latent variables as in (4.5) as these give semi-conjugate updates for the skill parameters. The latent variables are

$$z_{ij} | \mathcal{D}, \Lambda, \mathbf{c}, D, \mathbf{w} \stackrel{\text{indep}}{\sim} \text{Exp} \left(\sum_{m=j}^{n_i} \left(\lambda_{c_i, d_{c_i, x_{im}}} \right)^{w_i} + \sum_{m \in \mathcal{U}_i} \left(\lambda_{c_i, d_{c_i, m}} \right)^{w_i} \right) \quad (4.27)$$

for $i = 1, \dots, n$, $j = 1, \dots, n_i$.

Before we construct the density of all stochastic quantities (and subsequently derive the full conditional distributions for each random variable within the model) it is useful to define the complete data likelihood as

$$\begin{aligned} \pi(\mathcal{D}, Z | \Lambda, \mathbf{c}, D, \mathbf{w}, \alpha, \gamma) &= \pi(\mathcal{D} | \Lambda, \mathbf{c}, D, \mathbf{w}) \pi(Z | \mathcal{D}, \Lambda, \mathbf{c}, D, \mathbf{w}) \\ &= \prod_{i=1}^n \prod_{j=1}^{n_i} \left(\lambda_{c_i, d_{c_i, x_{ij}}} \right)^{w_i} \exp \left\{ - \left(\sum_{m=j}^{n_i} \left(\lambda_{c_i, d_{c_i, x_{im}}} e \right)^{w_i} + \sum_{m \in \mathcal{U}_i} \left(\lambda_{c_i, d_{c_i, m}} \right)^{w_i} \right) z_{ij} \right\} \\ &= \prod_{s=1}^{N^r} \prod_{t=1}^{N_s^e} \lambda_{st}^{\beta_{st}} \prod_{i=1}^n \prod_{j=1}^{n_i} \exp \left\{ - \left(\sum_{m=j}^{n_i} \left(\lambda_{c_i, d_{c_i, x_{im}}} \right)^{w_i} + \sum_{m \in \mathcal{U}_i} \left(\lambda_{c_i, d_{c_i, m}} \right)^{w_i} \right) z_{ij} \right\} \\ &= \prod_{s=1}^{N^r} \prod_{t=1}^{N_s^e} \lambda_{st}^{\beta_{st}} \exp \left(- \lambda_{st} \sum_{i=1}^n w_i \sum_{j=1}^{n_i} \zeta_{ij}(s, t) z_{ij} \right), \end{aligned} \quad (4.28)$$

where

$$\beta_{st} = \sum_{i=1}^n w_i \mathbb{I}(c_i = s) \sum_{j=1}^{n_i} \mathbb{I}(d_{c_i, x_{ij}} = t),$$

is the same as that under the conditional approach (4.7) and gives the number of times that the random variable λ_{st} is assigned to an entity within an informative ranking, and

$$\zeta_{ij}(s, t) = \mathbb{I}(c_i = s) \left(\sum_{m=j}^{n_i} \mathbb{I}(d_{c_i, x_{im}} = t) + \sum_{m \in \mathcal{U}_i} \mathbb{I}(d_{c_i, m} = t) \right),$$

is also as defined under the conditional approach in (4.8) and gives the number of times that the random variable λ_{st} represents an entity which is ranked no higher than j th in the i th ranking.

4.6.4 Full conditional distributions

The posterior distribution is formed by applying Bayes' Theorem. The posterior distribution $\pi(\Lambda, Z, \mathbf{c}, D, \mathbf{w}, \alpha, \gamma | \mathcal{D})$ is now a joint distribution of the latent random variables Z , the collection of unique skill parameters Λ , the binary indicator variables \mathbf{w} , the latent

cluster indicators \mathbf{c} , D and the DP concentration parameters α, γ . Posterior realisations of the latent indicators can be sampled from their respective full conditional distribution via a nested version of Neal's Algorithm 8 (Neal, 2000) which is described in the next section. Further the full conditional distribution for the DP concentration parameters are akin to those in Section 3.4.2 and are given in Section 4.6.5 when we provide a complete outline of the MCMC scheme used to generate posterior samples. Conditional on the latent cluster indicator variables and DP concentration parameters, the density of all remaining stochastic quantities is

$$\begin{aligned}
 \pi(\Lambda, \mathcal{D}, Z, \mathbf{w} | \mathbf{c}, D, \alpha, \gamma) &= \pi(\Lambda, \mathcal{D}, Z, \mathbf{w} | \mathbf{c}, D) \\
 &= \pi(Z | \mathcal{D}, \Lambda, \mathbf{c}, D, \mathbf{w}) \pi(\mathcal{D} | \Lambda, \mathbf{c}, D, \mathbf{w}) \pi(\Lambda | \mathbf{c}, D) \pi(\mathbf{w}) \\
 &= \prod_{s=1}^{N^r} \prod_{t=1}^{N_s^e} \lambda_{st}^{\beta_{st}} \exp \left(-\lambda_{st} \sum_{i=1}^n w_i \sum_{j=1}^{n_i} \zeta_{ij}(s, t) z_{ij} \right) \\
 &\quad \times \prod_{s=1}^{N^r} \prod_{t=1}^{N_s^e} \frac{\lambda_{st}^{a-1} e^{-\lambda_{st}}}{\Gamma(a)} \times \prod_{i=1}^n p_i^{w_i} (1-p_i)^{1-w_i} \\
 &= \prod_{s=1}^{N^r} \prod_{t=1}^{N_s^e} \frac{\lambda_{st}^{\beta_{st}+a-1}}{\Gamma(a)} \exp \left\{ - \left(1 + \sum_{i=1}^n w_i \sum_{j=1}^{n_i} \zeta_{ij}(s, t) z_{ij} \right) \lambda_{st} \right\} \\
 &\quad \times \prod_{i=1}^n p_i^{w_i} (1-p_i)^{1-w_i}.
 \end{aligned}$$

The full conditional distributions are then as follows.

- Λ : For $s = 1, \dots, N^r$, $t = 1, \dots, N_s^e$,

$$\lambda_{st} | \mathcal{D}, \Lambda_{-st}, Z, \mathbf{c}, D, \mathbf{w}, \alpha, \gamma \stackrel{\text{indep}}{\sim} \text{Ga} \left(a + \beta_{st}, 1 + \sum_{i=1}^n w_i \sum_{j=1}^{n_i} \zeta_{ij}(s, t) z_{ij} \right).$$

- Z : The latent variables are defined by their full conditional distribution (4.27). Hence for $i = 1, \dots, n$, $j = 1, \dots, n_i$,

$$z_{ij} | \mathcal{D}, \Lambda, Z_{-ij}, \mathbf{c}, D, \mathbf{w}, \alpha, \gamma \stackrel{\text{indep}}{\sim} \text{Exp} \left(\sum_{m=j}^{n_i} \left(\lambda_{c_i, d_{c_i, x_{im}}} \right)^{w_i} + \sum_{m \in \mathcal{U}_i} \left(\lambda_{c_i, d_{c_i, m}} \right)^{w_i} \right)$$

- \mathbf{w} : This has the same FCD as under the conditional approach. Hence for $i = 1, \dots, n$, w_i follows the discrete distribution given by

$$\begin{aligned} \Pr(w_i = 1 | \mathbf{w}_{-i}, \dots) &\propto \Pr(w_i = 1) \pi(\mathcal{D} | \mathbf{w}_{-i}, w_i = 1, \dots) \pi(Z | \mathbf{w}_{-i}, w_i = 1, \dots) \\ &\propto p_i \prod_{j=1}^{n_i} \lambda_{c_i, d_{c_i, x_{ij}}} \exp \left\{ -z_{ij} \left(\sum_{m=j}^{n_i} \lambda_{c_i, d_{c_i, x_{im}}} + \sum_{m \in \mathcal{U}_i} \lambda_{c_i, d_{c_i, m}} \right) \right\}. \end{aligned}$$

$$\begin{aligned} \Pr(w_i = 0 | \mathbf{w}_{-i}, \dots) &\propto \Pr(w_i = 0) \pi(\mathcal{D} | \mathbf{w}_{-i}, w_s = 0, \dots) \pi(Z | \mathbf{w}_{-i}, w_i = 0, \dots) \\ &\propto (1 - p_i) \exp \left\{ - \sum_{j=1}^{n_i} z_{ij} (K_i - j + 1) \right\}. \end{aligned}$$

Therefore the full conditional is

$$w_i | \dots \stackrel{\text{indep}}{\sim} \text{Bern}(\rho_i),$$

for $i = 1, \dots, n$ where

$$\rho_i = \frac{\Pr(w_i = 1 | \mathbf{w}_{-i}, \dots)}{\Pr(w_i = 1 | \mathbf{w}_{-i}, \dots) + \Pr(w_i = 0 | \mathbf{w}_{-i}, \dots)}$$

is the probability that ranking i is informative (given the other quantities).

4.6.5 MCMC – a marginal sampler

We are now in a position to describe the algorithm used for sampling from the posterior distribution $\pi(\Lambda, Z, \mathbf{c}, D, \mathbf{w}, \alpha, \gamma | \mathcal{D})$ under the WAND model. Recall that $N^r = |\{c_i\}_{i=1,\dots,n}|$ is the current number of ranker clusters and $N_s^e = |\{d_{sj}\}_{j=1,\dots,K}|$ denotes the number of entity clusters within ranker cluster s . The state of the Markov chain then consists of $\mathbf{c} = (c_i)$, $D = (d_{sl})$, $\Lambda = (\lambda_{st})$, $Z = (z_{ij})$, $\mathbf{w} = (w_i)$, $\gamma = (\gamma_s)$ and α for $s = 1, \dots, N^r$, $t = 1, \dots, N_s^e$, $i = 1, \dots, n$, $j = 1, \dots, n_i$ and $l = 1, \dots, K$. Now if we first define the contribution to the complete data likelihood from ranker i to be

$$f(\mathbf{x}_i, \mathbf{z}_i | \Lambda, \mathbf{c}, D, \mathbf{w}, \alpha, \gamma) = \prod_{j=1}^{n_i} \lambda_{c_i, d_{c_i, x_{ij}}}^{w_i} \exp \left\{ - \left(\sum_{m=j}^{n_i} \lambda_{c_i, d_{c_i, x_{im}}}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_i, d_{c_i, m}}^{w_i} \right) z_{ij} \right\}$$

then the updates proceed as follows.

- For $i = 1, \dots, n$: Let q^{r-} be the number of distinct c_j for $j \neq i$ and $h^r = q^{r-} + m^r$. Label these c_j values in $\{1, \dots, q^{r-}\}$. If $c_i = c_j$ for some $j \neq i$, draw $\boldsymbol{\lambda}_c \stackrel{\text{indep}}{\sim} \text{DP}(\gamma_c, G_0)$ for $q^{r-} < c \leq h^r$. If $c_i \neq c_j \forall j \neq i$, let c_i have the label $q^{r-} + 1$, and draw $\boldsymbol{\lambda}_c \stackrel{\text{indep}}{\sim} \text{DP}(\gamma_c, G_0)$ for $q^{r-} + 1 < c \leq h^r$.

Draw a new value for c_i from $\{1, \dots, h^r\}$ using probabilities

$$\begin{aligned} \Pr(c_i = c | \mathcal{D}, \Lambda, Z, \mathbf{c}_{-i}, D, \mathbf{w}, \alpha, \boldsymbol{\gamma}) \\ = \begin{cases} b n_{-i,c}^r f(\mathbf{x}_i, \mathbf{z}_i | \Lambda, \mathbf{c}_{-i}, c_i = c, D, \mathbf{w}, \alpha, \boldsymbol{\gamma}), & 1 \leq c \leq q^{r-}, \\ b \frac{\alpha}{m^r} f(\mathbf{x}_i, \mathbf{z}_i | \Lambda, \mathbf{c}_{-i}, c_i = c, D, \mathbf{w}, \alpha, \boldsymbol{\gamma}), & q^{r-} < c \leq h^r, \end{cases} \end{aligned}$$

where $n_{-i,c}^r$ is the number of c_j for $j \neq i$ that are equal to c , and b is the appropriate normalising constant. Change the state to contain only those λ_c that are now associated with one or more observations. i.e. let $\Lambda = (\lambda_c : c \in \{c_1, \dots, c_n\})$.

- Relabel \mathbf{c} so that $c_i \in \{1, \dots, N^r\}$ for $i = 1, \dots, n$.
- For $s = 1, \dots, N^r$, $i = 1, \dots, K$: Let q_s^{e-} be the number of distinct d_{sj} for $j \neq i$ and $h_s^e = q_s^{e-} + m^e$. Label these d_{sj} values in $\{1, \dots, q_s^{e-}\}$. If $d_{si} = d_{sj}$ for some $j \neq i$, draw $\lambda_d \stackrel{\text{indep}}{\sim} G_0$ for $q_s^{e-} < d \leq h_s^e$. If $d_{si} \neq d_{sj} \forall j \neq i$, let d_{si} have the label $q_s^{e-} + 1$, and draw $\lambda_d \stackrel{\text{indep}}{\sim} G_0$ for $q_s^{e-} + 1 < d \leq h_s^e$.

Draw a new value for d_{si} from $\{1, \dots, h_s^e\}$ using probabilities

$$\begin{aligned} \Pr(d_{si} = d | \mathcal{D}, \Lambda, Z, \mathbf{c}, D_{-si}, \mathbf{w}, \alpha, \boldsymbol{\gamma}) \\ = \begin{cases} b n_{s,-i,d}^e \prod_{i \in R} f(\mathbf{x}_i, \mathbf{z}_i | \Lambda, \mathbf{c}, D_{-si}, D_{si} = d, \mathbf{w}, \alpha, \boldsymbol{\gamma}), & 1 \leq d \leq q_s^{e-}, \\ b \frac{\gamma_s}{m^e} \prod_{i \in R} f(\mathbf{x}_i, \mathbf{z}_i | \Lambda, \mathbf{c}, D_{-si}, D_{si} = d, \mathbf{w}, \alpha, \boldsymbol{\gamma}), & q_s^{e-} < d \leq h^e, \end{cases} \end{aligned}$$

where $n_{s,-i,d}^e$ is the number of d_{sj} for $j \neq i$ that are equal to d , $R = \{i : c_i = s\}$ and b is the appropriate normalising constant. Change the state to contain only those λ that are now associated with one or more observations, that is, let $\Lambda = (\lambda_{st} : s = 1, \dots, N^r, t \in \{d_{s1}, \dots, d_{sK}\})$.

- For $s = 1, \dots, N^r$ relabel \mathbf{d}_s such that $d_{sj} \in \{1, \dots, N_s^e\}$ for $j = 1, \dots, K$.
- For $s = 1, \dots, N^r$, $t = 1, \dots, N_s^e$ sample

$$\lambda_{st} | \mathcal{D}, \Lambda_{-st}, Z, \mathbf{c}, D, \mathbf{w}, \alpha, \boldsymbol{\gamma} \stackrel{\text{indep}}{\sim} \text{Ga} \left(a + \beta_{st}, 1 + \sum_{i=1}^n w_i \sum_{j=1}^{n_i} \zeta_{ij}(s, t) z_{ij} \right),$$

- For $i = 1, \dots, n$, $j = 1, \dots, n_i$ sample

$$z_{ij} | \mathcal{D}, \Lambda, Z_{-ij}, \mathbf{c}, D, \mathbf{w}, \alpha, \boldsymbol{\gamma} \stackrel{\text{indep}}{\sim} \text{Exp} \left(\sum_{m=j}^{n_i} \lambda_{c_i, d_{c_i, x_{im}}}^{w_i} + \sum_{m \in \mathcal{U}_i} \lambda_{c_i, d_{c_i, m}}^{w_i} \right).$$

- For $i = 1, \dots, n$, sample w_i from the discrete distribution given by

$$\begin{aligned}\Pr(w_i = 1 | \mathcal{D}, \Lambda, Z, \mathbf{c}, D, \mathbf{w}_{-i}, \alpha, \gamma) &\propto p_i f(\mathbf{x}_i, \mathbf{z}_i | \Lambda, \mathbf{c}, D, \mathbf{w}_{-i}, w_i = 1, \alpha, \gamma), \\ \Pr(w_i = 0 | \mathcal{D}, \Lambda, Z, \mathbf{c}, D, \mathbf{w}_{-i}, \alpha, \gamma) &\propto (1 - p_i) f(\mathbf{x}_i, \mathbf{z}_i | \Lambda, \mathbf{c}, D, \mathbf{w}_{-i}, w_i = 0, \alpha, \gamma) \\ &\propto (1 - p_i) \exp \left\{ - \sum_{j=1}^{n_i} z_{ij}(K_i - j + 1) \right\}.\end{aligned}$$

- Rescale

- Sample $\Lambda^\dagger \sim \text{Ga} \left(a \sum_{s=1}^{N^r} N_s^e, 1 \right)$.
- Calculate $\Sigma = \sum_{s=1}^{N^r} \sum_{t=1}^{N_s^e} \lambda_{st}$.
- For $s = 1, \dots, N^r$, $t = 1, \dots, N_s^e$, let $\lambda_{st} \rightarrow \lambda_{st} \Lambda^\dagger / \Sigma$.

Conditional on the prior distribution described in Section 4.6.3, the concentration parameters can be sampled from the following mixtures

- Sample

$$\alpha | \dots \sim \pi \text{Ga}(a_\alpha + N^r, b_\alpha - \log \eta) + (1 - \pi) \text{Ga}(a_\alpha + N^r - 1, b_\alpha - \log \eta),$$

where

$$\frac{\pi}{(1 - \pi)} = \frac{a_\alpha + N^r - 1}{n(b_\alpha - \log \eta)}, \quad \text{and} \quad \eta | \dots \sim \text{Beta}(\alpha + 1, n).$$

- For $s = 1, \dots, N^r$ sample

$$\gamma_s | \dots \stackrel{\text{indep}}{\sim} \pi_s \text{Ga}(a_\gamma + N_s^e, b_\gamma - \log \eta_s) + (1 - \pi_s) \text{Ga}(a_\gamma + N_s^e - 1, b_\gamma - \log \eta_s),$$

where

$$\frac{\pi_s}{(1 - \pi_s)} = \frac{a_\gamma + N_s^e - 1}{K(b_\gamma - \log \eta_s)}, \quad \text{and} \quad \eta_s | \dots \stackrel{\text{indep}}{\sim} \text{Beta}(\gamma_s + 1, K).$$

4.7 Simulation studies

In this section we present two illustrative simulation studies based on simulated data which highlight the flexibility of the Bayesian WAND for the analysis of ranked data.

4.7.1 Study 1

In this study we consider two (new) datasets with *complete* rankings of $K = 20$ entities. The first dataset (Dataset 3) contains complete rankings from $n = 40$ informative ($w_i = 1$) rankers in a single ranker group ($N^r = 1$ and $c_i = 1$). The distinct “skill” parameters λ_k were sampled from the prior distribution with $\gamma_1 = 1$ and base distribution $G_0 = \text{Ga}(1, 1)$. This simulation gave six unique entity clusters ($N_1^e = 6$). The entity clusters are $\{1\} \succ \{2\} \succ \{3, 4\} \succ \{5, 6\} \succ \{7-16\} \succ \{17-20\}$ where \succ means “is preferred to” and the distinct skill parameters are 3.07, 1.83, 1.47, 0.85, 0.45, 0.04 for each cluster respectively. The simulated data are given in Table B.3 within the appendices. Note that, for ease of interpretation, we applied a permutation to the entity labels so that the λ_k were decreasing, with the most preferred entity being labelled 1 and the least preferred labelled 20.

The second dataset (Dataset 4) contains complete rankings from $n = 50$ rankers, consisting of those in Dataset 3 and an additional set from 10 uninformative rankers, with $w_{41:50} = 0$. The addition of these random rankings will allow us to investigate the extent to which the model both identifies and handles uninformative rankings. These additional (random) rankings are given within the appendices in Table B.4.

We investigate the effect of incomplete rankings by comparing the analysis of the complete rankings with those of top- M rankings for $M = 5, 10, 15$. That is, we analyse the data assuming rankers see $K_i = K = 20$ entities and report their top $n_i = 5, 10, 15, 20$ entities. These analyses will allow us to investigate the level of uncertainty introduced by only observing truncations of the rankings. It will also be interesting to explore whether a collection of complete rankings is required if we only wish to infer, say, the top-5 entities. Another scenario we consider is that of a so-called “restricted” analysis in which any entity not ranked by any ranker is removed from the dataset. The intuition behind this scenario is that our beliefs about the orderings of entities that appear in at least one ranking should not be affected by whether non-appearing entities are included or not. One example of this is in Dataset 3 where, under the top-5 and top-10 scenarios, entities 17, 18 and 20 do not appear in any of the rankings. Thus we also consider restricted top-5 and top-10 analyses of Dataset 3 which use $K_i = K = 17$.

We will adopt the same base distribution as for previous analyses and take $G_0 = \text{Ga}(1, 1)$. Further we choose the prior distributions for the concentration parameters to be $\alpha \sim$

$\text{Ga}(1, 1)$ and $\gamma_s \sim \text{Ga}(3, 3)$ for $s \in \mathbb{N}$, noting that these are common choices within the literature (Rodriguez et al., 2008). Here we consider two scenarios of an exchangeable prior for the ranker reliability parameters \mathbf{w} : one in which we are unsure about their ability (taking $p_i = 0.5$) and the other where we are quite confident that they are informative (taking $p_i = 0.9$).

To generate realisations from the posterior distribution (for each analysis) we implemented the (marginal) sampling algorithm outlined in Section 4.6.5 with $m^r = 2$ and $m^e = 3$ auxiliary (ranker and entity) variables. Each Markov chain was initialised at a random draw from the prior distribution. To obtain 10K (almost) un-autocorrelated realisations from the posterior distribution we allowed each chain a burn-in period of 10K iterations then ran the scheme for a further 1M iterations and thinned the output by a factor of 100 in each case. The most computationally expensive analyses were those of Dataset 4 with $p_i = 0.9$ and the computational time required to perform inference was (approximately) 7, 16, 21 and 26 minutes for the top-5, top-10, top-15 and complete cases respectively. The mixing of the MCMC chains was assessed by inspecting trace plots and convergence was assessed by initialising numerous chains at differing starting values and verifying that the resulting posterior distributions were equivalent (up to stochastic noise).

Figure 4.3 shows the posterior probability $\Pr(w_i = 1 | \mathcal{D})$ that ranker i is informative for each information scenario and for both choices of prior probabilities $p_i = \Pr(w_i = 1)$. It is not surprising that the restricted analyses, due to the loss of information, produce results which are less consistent with the “known” abilities of the rankers ($w_{1:40} = 1$ and $w_{41:50} = 0$) than the full (unrestricted) analyses which consider all entities. This finding is clear for both choices of the p_i but is more noticeable for the $p_i = 0.5$ case. The results correctly show that the the majority of rankers have been identified as informative (middle row). Unsurprisingly this identification becomes clearer as more comprehensive rankings are used (going from top-5 up to complete). It is also interesting to see that the most data-poor (top-5) analysis does reasonably well.

The analyses for Dataset 4 show that the uninformative (random) rankers have been identified quite clearly, particularly for the $p_i = 0.5$ case where the posterior probability that rankers 41 to 50 are informative are very close to zero (bottom left plot). The probabilities for rankers 1 to 40 remain similar to those found when analysing Dataset 3. This (and other analyses not given here) suggests that the identification of informative rankers using this model is fairly robust to the addition of rankings from uninformative rankers. The bottom row of plots show the influence of the choice of the p_i in the prior distribution. Here we see that having a high level of confidence that uninformative rankers are informative can potentially mask their identification. It is clear here that the posterior probabilities $\Pr(w_i = 1 | \mathcal{D})$ for the uninformative rankers are well separated from those for

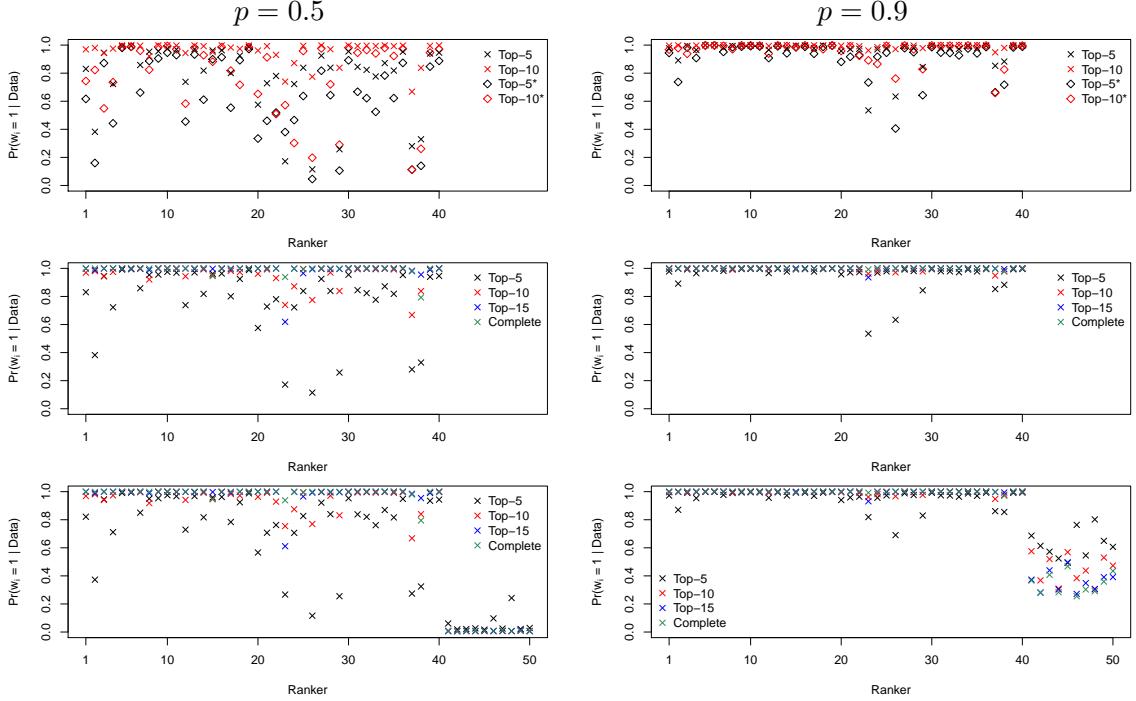


Figure 4.3: Plots of the posterior probability $\Pr(w_i = 1 | \mathcal{D})$ that ranker i is informative for both scenarios of prior on their ability: $p_i = 0.5$ (left column) and $p_i = 0.9$ (right column). The top row of plots show the comparison between the restricted (*) and full (unrestricted) analyses for Dataset 3. Plots in the middle row are those for the full analyses using Dataset 3, with the corresponding plots using Dataset 4 in the bottom row.

informative rankers but nevertheless they might be identified as informative if this choice were made by thresholding these posterior probabilities at say 0.5 or even higher. This suggests that the analyst should use a fairly conservative choice of the p_i and should be careful about expressing over confidence in ranker abilities *a priori*.

Table 4.1 gives the posterior distribution of the number of ranker clusters N^r under each analysis. For Dataset 3, we see much more posterior support for a single ranker group under the full (unrestricted) analyses compared to their restricted equivalents, with little dependence on the choice of the p_i . Also, for the analyses of Dataset 4, the posterior support for a single ranker group reduces, particularly for the $p_i = 0.9$ case. Indeed for this case, the high prior confidence that all rankers are informative changes the modal number of ranker clusters from one to two, though this comes with a higher level of posterior uncertainty. Also, as before, the probability of the correct number of ranker clusters increases as more comprehensive rankings are included within the analysis.

In Chapter 3 we noted that determining the allocation of rankers to ranker groups can be problematic. For example, conditioning on the modal number of ranker clusters and allocating ranker i to ranker cluster k using only an MCMC estimate of $\Pr(c_i = k | N^r, \mathcal{D})$ can

$p = 0.5$						$p = 0.9$					
Dataset 3	1	2	3	4	≥ 5	Dataset 3	1	2	3	4	≥ 5
Top-5*	0.73	0.20	0.05	0.02	0.00	Top-5*	0.65	0.23	0.08	0.02	0.01
Top-10*	0.88	0.11	0.01	0.00	0.00	Top-10*	0.88	0.11	0.01	0.00	0.00
Top-5	0.87	0.11	0.02	0.00	0.00	Top-5	0.87	0.11	0.02	0.00	0.00
Top-10	0.98	0.02	0.00	0.00	0.00	Top-10	0.99	0.01	0.00	0.00	0.00
Top-15	0.99	0.01	0.00	0.00	0.00	Top-15	1.00	0.00	0.00	0.00	0.00
Complete	1.00	0.00	0.00	0.00	0.00	Complete	1.00	0.00	0.00	0.00	0.00
Dataset 4	1	2	3	4	≥ 5	Dataset 4	1	2	3	4	≥ 5
Top-5	0.74	0.20	0.05	0.01	0.00	Top-5	0.06	0.29	0.29	0.19	0.10
Top-10	0.88	0.10	0.02	0.00	0.00	Top-10	0.09	0.38	0.30	0.16	0.06
Top-15	0.89	0.10	0.01	0.00	0.00	Top-15	0.15	0.36	0.28	0.14	0.05
Complete	0.89	0.10	0.01	0.00	0.00	Complete	0.16	0.35	0.28	0.14	0.06

Table 4.1: Posterior distribution of the number of ranker clusters N^r for restricted (*) and full (unrestricted) analyses. Numbers in bold indicate modal values.

fail to give an adequate description of the joint posterior distribution of the allocations \mathbf{c} , particularly if there is not overwhelming posterior support for the choice of N^r . Instead we prefer not to condition on a particular N^r and to use the full MCMC output to look at co-clustering probabilities $\Pr(c_i = c_j | \mathcal{D})$. We define a dissimilarity matrix $\Delta = (\Delta_{ij})$, where $\Delta_{ij} = \Pr(c_i \neq c_j | \mathcal{D})$ measures how dissimilar rankers i and j are, and then consider the corresponding dendrogram when forming the allocation of rankers to ranker clusters, whilst also accounting for the posterior distribution of N^r . We use the complete linkage method, also known as furthest neighbour clustering, as this tends to produce more densely packed clusters and does not suffer from “chaining”.

The posterior distribution for the number of ranker clusters in the Dataset 3 analysis gives overwhelming support for the true number $N^r = 1$ for each case, and particularly for the complete, top-15 and top-10 cases. Therefore the allocation of rankers to ranker clusters is trivial. However, the allocation is not quite as straightforward in the Dataset 4 analysis. Note that here we have retained the same prior for the upper level concentration parameter α rather than amend it to reflect the known heterogeneity in ranker beliefs in Dataset 4.

Figure 4.4 gives the complete linkage dendograms for both $p_i = 0.5$ and $p_i = 0.9$ cases in the (complete ranking) analysis of Dataset 4. The method clearly picks out the informative and homogeneous rankers (numbered 1–40) and puts them into a single ranker group. The posterior probabilities that these rankers are co-clustered is 0.997 and 0.979 respectively – these probabilities are straightforward to obtain from the (posterior) realisations of cluster allocations and are given by $\Pr(c_1 = c_2 = \dots = c_{40} | \mathcal{D})$. When $p_i = 0.5$ we see that the uninformative ranker most “similar” to the informative group is ranker 42. This ranker

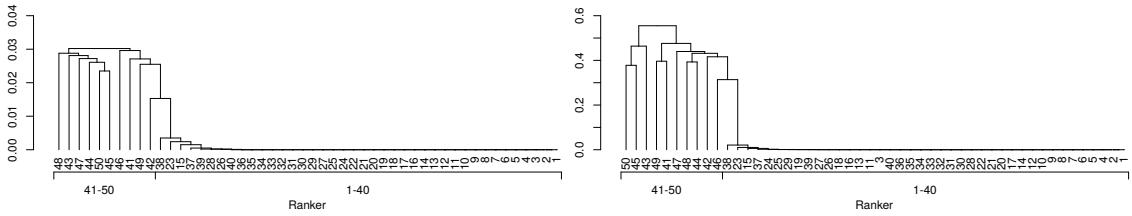


Figure 4.4: Dendrograms for ranker clustering within Dataset 4 under a complete analysis for $p_i = 0.5$ (left plot) and $p_i = 0.9$ (right plot).

is co-clustered with each of the informative rankers at least 98.5% of the time. The other uninformative rankers also have high co-clustering probabilities and this is consistent with the very high posterior support for a single group of rankers, $\Pr(N^r = 1 | \mathcal{D}, p_i = 0.5) = 0.89$. This result occurs as a consequence of the model down-weighting the influence of the uninformative rankers: $\Pr(w_i = 1 | \mathcal{D}, p_i = 0.5) \ll 0.1$ for $i = 41, \dots, 50$. On the other hand when we are much more confident in the ability of the rankers (with $p_i = 0.9$) the most ‘‘similar’’ uninformative ranker to the informative group is co-clustered with informative rankers at least 68.6% of the time, a much smaller proportion than in the $p_i = 0.5$ case. Also the uninformative rankers do not separate themselves into a single distinct cluster, with $0.311 \leq \Delta_{ij} \leq 0.555$ for any $i \neq j \in \{41-50\}$, that is, any pair of uninformative rankers are co-clustered between 44.5%–68.9% of the time. It is perhaps not surprising that the model is not able to detect significant similarities between any pair of the ten uninformative rankers as their rankings are random permutations and they are few in number.

Table 4.2 gives the marginal posterior distributions of the number of entity clusters conditional on a single ranker cluster (for all analyses of each dataset). Note that the $p_i = 0.9$ analysis of Dataset 4 gives very low posterior support for a single ranker cluster, with $\Pr(N^r = 1 | \mathcal{D}, p_i = 0.9) = 0.16$, and so later in Section 4.7.1 we look at results when conditioning on two ranker clusters (the posterior modal number). The table also shows that posterior support for the correct number of entity clusters ($N_1^e = 6$) increases as the information provided within each ranking increases. The cost of performing a restricted analysis is especially visible in the top–5 case. As was the case for ranker clustering, the inclusion of complete rankings in comparison to top–10 rankings does not have a significant effect on our posterior beliefs.

Figure 4.5 shows dendrograms of the entity grouping structure for each of the complete analyses, conditional on a single ranker cluster. Note that the entity clusters are similar under both $p_i = 0.5$ and $p_i = 0.9$ analyses, particularly for entities 1–6 and 15–20. The other entities (on the right hand side of each dendrogram) also have a similar grouping

		$p = 0.5$									
Dataset 3		1	2	3	4	5	6	7	8	9	≥ 10
Top-5*		0.00	0.11	0.23	0.27	0.18	0.12	0.06	0.02	0.01	0.00
Top-10*		0.00	0.00	0.05	0.19	0.28	0.23	0.15	0.06	0.02	0.02
Top-5		0.00	0.00	0.15	0.27	0.24	0.17	0.09	0.04	0.02	0.02
Top-10		0.00	0.00	0.03	0.13	0.22	0.25	0.19	0.11	0.05	0.03
Top-15		0.00	0.00	0.00	0.07	0.21	0.29	0.22	0.13	0.05	0.03
Complete		0.00	0.00	0.00	0.09	0.24	0.29	0.21	0.11	0.04	0.02
Dataset 4		1	2	3	4	5	6	7	8	9	≥ 10
Top-5		0.00	0.00	0.14	0.26	0.25	0.17	0.10	0.05	0.02	0.01
Top-10		0.00	0.00	0.03	0.12	0.22	0.24	0.19	0.12	0.05	0.03
Top-15		0.00	0.00	0.00	0.08	0.22	0.29	0.23	0.12	0.05	0.01
Complete		0.00	0.00	0.00	0.10	0.24	0.28	0.21	0.11	0.04	0.02
		$p = 0.9$									
Dataset 3		1	2	3	4	5	6	7	8	9	≥ 10
Top-5*		0.00	0.10	0.28	0.29	0.20	0.08	0.03	0.02	0.00	0.00
Top-10*		0.00	0.00	0.04	0.18	0.28	0.25	0.15	0.07	0.02	0.01
Top-5		0.00	0.00	0.21	0.31	0.24	0.14	0.06	0.02	0.01	0.01
Top-10		0.00	0.00	0.03	0.11	0.22	0.24	0.20	0.12	0.05	0.03
Top-15		0.00	0.00	0.00	0.08	0.22	0.28	0.23	0.13	0.05	0.01
Complete		0.00	0.00	0.00	0.08	0.22	0.30	0.22	0.11	0.05	0.02
Dataset 4		1	2	3	4	5	6	7	8	9	≥ 10
Top-5		0.00	0.00	0.17	0.24	0.25	0.18	0.08	0.05	0.02	0.01
Top-10		0.00	0.00	0.02	0.10	0.24	0.24	0.21	0.11	0.05	0.03
Top-15		0.00	0.00	0.00	0.06	0.22	0.28	0.25	0.13	0.05	0.01
Complete		0.00	0.00	0.01	0.07	0.23	0.28	0.23	0.12	0.05	0.01

Table 4.2: Posterior distribution of the number of entity clusters, conditional on a single ranker cluster, for restricted (*) and full (unrestricted) analyses. Numbers in bold indicate modal values.

structure with only minor discrepancies in the order that entity pairs cluster together (and clustering taking place at similar levels of the dissimilarity measure Δ_{ij}). Thus the dendrograms are fairly robust to the addition of the uninformative rankings. This can partly be explained by the dendrograms being conditional on a single ranker cluster and the Dataset 4 analysis correctly identifying the uninformative rankers (with $w_{41:50} = 0$).

We can explore our posterior distribution further by investigating where specific entities are likely to be ranked. Consider the posterior probability that a specific entity is ranked at most i th, that is, $P_i = \Pr(\text{entity in top } i | \mathcal{D})$. Figure 4.6 displays P_5 for all analyses, P_{10} for all except the top-5 case and P_{15} for the top-15 and complete analyses. Interestingly the posterior probabilities under the restricted top-5 and top-10 analyses are very similar to those under the unrestricted analysis (especially when $p_i = 0.5$). This suggests that, for these data, this aspect of the posterior distribution is robust to whether or not the unobserved entities (in the restricted analysis) are included in the analysis. The two left

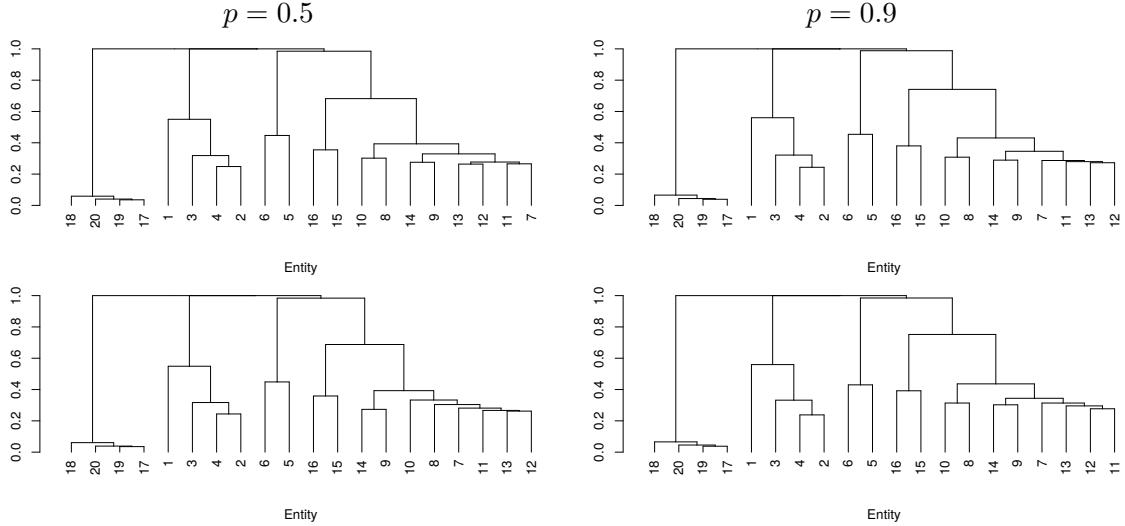


Figure 4.5: Dendrograms for entity clustering for Dataset 3 (top) and Dataset 4 (bottom) conditional on a single ranker cluster under both prior specifications for the complete analyses.

hand plots within Figure 4.6 shows considerable similarity between the full (unrestricted) analyses of the two datasets (when $p_i = 0.5$). Here the WAND model has been able to identify the so-called spam rankings within Dataset 4 (see Figure 4.3) and so these rankings have little effect on the analysis. However, this is not the case when we take $p_i = 0.9$. In this case, the high level of confidence that the rankers are informative results in the WAND model being reluctant to classify any rankers as uninformative; again see Figure 4.3. This leads to the uninformative rankers contaminating the posterior distribution of the entity λ -parameters.

Earlier we questioned whether a collection of complete rankings is required if we only wish to infer, say, the top-5 entities. Looking at the results for Dataset 3 (in the top plots in Figure 4.6), if we consider P_5 we can see how the top-5 analysis is able to detect that entities 1–4 are highly likely to be within the top-5, however there is some doubt about which is the 5th strongest entity. In contrast, the other analyses (top-10, top-15 and complete) indicate that entity 6 is much more likely to be within the top-5 in comparison to the remaining entities. Furthermore notice how P_5 decreases significantly for entities 7–20 under the top-15 and complete analyses in comparison to the top-10 scenario. Similar results are obtained for P_{10} and P_{15} , that is, as we increase the information contained within the rankings, we become more certain about those entities which are the most preferred. In conclusion, although it is possible to identify the most preferred entities without complete rankings, it has been advantageous to incorporate as much information as possible.

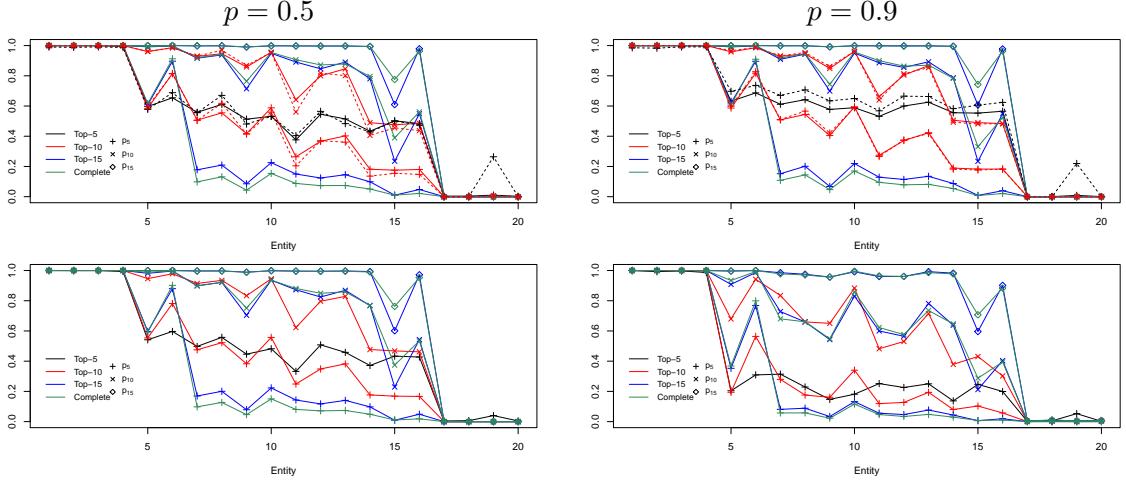


Figure 4.6: Posterior probabilities P_5 for all analyses, P_{10} for all except the top-5 case and P_{15} for the top-15 and complete analyses. The analyses of Dataset 3 and Dataset 4 are shown on the upper and lower row respectively for each prior choice of p .

Dataset 4 analysis conditional on two ranker clusters

Here we revisit the analysis of Dataset 4 under the choice of $p_i = 0.9$ *a priori*. In the previous section we looked at the posterior entity clustering structure conditional on a single ranker cluster. However there was little posterior support for a single ranker cluster, with $\Pr(N^r = 1 | \mathcal{D}, p_i = 0.9) \leq 0.16$ under all analyses considered, and the modal posterior number of ranker clusters was two. Therefore we now look at the (posterior) clustering structure conditional on this modal number of ranker groups (as would be done if we had no knowledge of the generating mechanism for these data).

Table 4.3 gives the marginal posterior distributions of the number of entity clusters within each ranker group (conditional on two ranker clusters) for all analyses. Note that, within ranker cluster 1, the posterior support for the correct number of entity clusters ($N_1^e = 6$) increases as the information provided within each ranking increases – this was also observed when conditioning on a single ranker group. For ranker cluster 2 we see increased uncertainty within the marginal posteriors (in comparison to ranker cluster 1) with only two or three entity clusters being most probable under all analyses. Clearly the rankers in ranker cluster 2 are less able to distinguish between the entities – and this is perhaps not surprising as this cluster typically houses the uninformative rankers.

Figure 4.7 shows dendrograms of the entity grouping structure within ranker clusters 1 and 2 (left and right respectively) for the complete analysis of Dataset 4 with $p_i = 0.9$, conditional on two ranker clusters. Note that the entity clusters within ranker cluster 1 are very similar to those when conditioning on a single ranker cluster; see Figure 4.5. This

Dataset 4	Cluster	1	2	3	4	5	6	7	8	9	≥ 10
Top-5	1	0.01	0.01	0.16	0.26	0.25	0.17	0.09	0.03	0.02	0.00
	2	0.14	0.23	0.22	0.18	0.12	0.06	0.03	0.01	0.01	0.00
Top-10	1	0.00	0.00	0.02	0.11	0.21	0.23	0.21	0.12	0.06	0.04
	2	0.11	0.21	0.24	0.19	0.13	0.07	0.03	0.01	0.01	0.00
Top-15	1	0.00	0.00	0.00	0.08	0.21	0.28	0.23	0.12	0.06	0.02
	2	0.18	0.25	0.21	0.16	0.10	0.05	0.03	0.01	0.01	0.00
Complete	1	0.00	0.00	0.00	0.08	0.25	0.28	0.20	0.12	0.05	0.02
	2	0.20	0.24	0.23	0.15	0.10	0.05	0.02	0.01	0.00	0.00

Table 4.3: Posterior distribution of the number of entity clusters, conditional on two ranker clusters, for each analysis of Dataset 4 with $p_i = 0.9$. Numbers in bold indicate modal values.

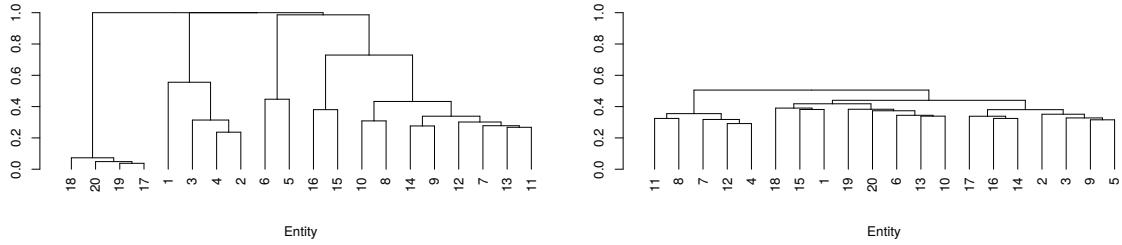


Figure 4.7: Dendrograms of entity clustering (conditional on 2 ranker clusters) in ranker cluster 1 (left) and ranker cluster 2 (right) for the analysis of Dataset 4 with $p_i = 0.9$.

is probably due to ranker cluster 1 containing all informative rankers in both cases (of one or two ranker groups). For ranker cluster 2 we see that any two entities are clustered together at least 49% of the time ($\Delta_{ij} < 0.51$). Also entity clusters are formed at similar levels of dissimilarity, again highlighting the uncertainty on the entity clustering within this ranker group.

4.7.2 Study 2

In study 2 we look at a single dataset with $n = 40$ *complete* rankings of $K = 20$ entities from informative ($w_i = 1$) rankers. We also simulate the cluster allocations (for both rankers and entities) marginally from the prior. The distinct skill parameters and cluster allocations were simulated using $\alpha = \gamma_s = 1$ for $s \in \mathbb{N}$, and $a = 1$ so that our base distribution is $G_0 = \text{Ga}(1, 1)$. The simulation gave three ranker clusters ($N^r = 3$) containing 24, 12 and 4 rankers, which we label as rankers 1–24, 25–36 and 37–40. Also the ranker clusters contained 8, 6 and 3 entity clusters ($N_1^e = 8$, $N_2^e = 6$, $N_3^e = 3$).

Table 4.4 shows the entity clustering (within each ranker cluster) along with the associated true values of the skill parameters on the log scale. For ease of interpretation, the entities are labelled according to the size of their “true aggregate” skill parameter, largest to

C^r	Rankers	Entity cluster							
		1	2	3	4	5	6	7	8
1	1–24	1	6	10,13	3,4,7,9,12,15	2	5,11,14,17–20	16	8
		2.47	0.65	0.52	0.40	0.34	0.24	0.02	0.01
2	25–36	2–5,8	1,7,9,11	14,16	12	6,10,15,17	13,18–20		
		1.72	0.76	0.68	0.42	0.21	0.15		
3	37–40	2,6,10	7	1,3–5,8,9,11–20					
		1.54	1.16	0.64					

Table 4.4: True allocation of entities in to clusters, along with the corresponding true parameter value for each of the entity clusters.

smallest, so that they are labelled with the most preferred entity overall first, down to the least preferred entity overall last. Here the true aggregate values are an average of the true parameter values within each ranker cluster, weighted by the size of the ranker clusters. The complete (simulated) rankings analysed within this study can be found in Table B.5 within the appendices.

The purpose of this study is to investigate the ability of our WAND model to (correctly) identify different ranker groups and the associated preferences therein. The analysis given here uses the same base distribution and prior distribution for the entity concentration parameters as in Section 4.7.1, that is, $G_0 = \text{Ga}(1, 1)$ and $\gamma_s \sim \text{Ga}(3, 3)$ for $s \in \mathbb{N}$. To reflect the known ranker heterogeneity within these data we now take $a_\alpha = b_\alpha = 3$, that is, $\alpha \sim \text{Ga}(3, 3)$. We also consider the case where we have only moderate confidence in our rankers being informative by taking $p_i = 0.5$.

Realisations from the posterior distribution were obtained using the (marginal) sampling algorithm outlined in Section 4.6.5 with $m^r = 2$ and $m^e = 3$ auxiliary (ranker and entity) variables. The Markov chain was initialised at a random draw from the prior distribution. To obtain 10K (almost) un-autocorrelated realisations from the posterior distribution we performed a burn-in period of 10K iterations and then ran the scheme for a further 1M iterations and thinned the output by a factor of 100. The computational time required to perform inference was (approximately) 17 minutes. The mixing of the MCMC chain was assessed by inspecting trace plots and convergence was assessed by initialising numerous chains at differing starting values and verifying that the resulting posterior distributions were equivalent (up to stochastic noise).

The left plot in Figure 4.8 shows the posterior probabilities, $\Pr(w_i = 1 | \mathcal{D})$, that ranker i is informative. The plot shows that, in general, the rankers in (true) ranker clusters 1 and 2 (rankers 1–36) are well identified to be informative. However the rankers in (true) ranker cluster 3 are identified as uninformative. The reason for this misidentification is perhaps due to the (true) entity clustering structure present within ranker cluster 3. Table 4.4 shows that this ranker cluster contains only 3 entity clusters, with one of these containing 16 out of the 20 entities, and so it is very likely that rankings in this cluster

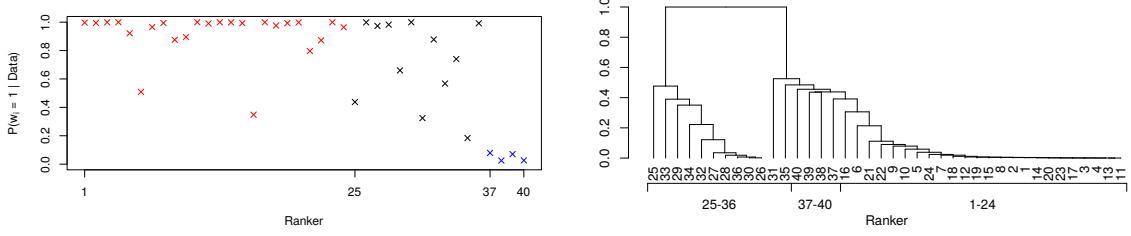


Figure 4.8: Plot of the posterior probability $\Pr(w_i = 1 | \mathcal{D})$ that ranker i is informative (left), colours distinguish between the “true” ranker clusters. Dendrogram (complete linkage) computed using the dissimilarity Δ_{ij} between rankers i and j (right).

resemble a random permutation of the K entities.

The right plot in Figure 4.8 shows the complete linkage dendrogram determined using dissimilarities Δ_{ij} . The dendrogram suggests there are two ranker clusters (taking dissimilarity $\in (0.53, 1)$) which separates those rankers numbered $\{25 - 30, 32, 33, 34, 36\}$ from the remaining rankers. That there are two ranker clusters is supported further by the marginal posterior distribution of the number of ranker clusters: $\Pr(N^r = i | \mathcal{D}) = 0.68, 0.25, 0.06, 0.01$ for $i = 2, 3, 4, 5$. It is not surprising that the analysis has not identified the third ranker cluster as this cluster only contains rankers whose rankings are virtually indistinguishable from random permutations; rather the model prefers to deem such rankers as uninformative and place them within clusters of informative rankers.

Table 4.5 gives the marginal posterior distribution for the number of entity clusters within each ranker cluster, conditional on the posterior modal number of ranker clusters. The modal number of entity clusters within ranker clusters 1 and 2 is six and four respectively (the corresponding true values are eight and six). Here the analysis has correctly identified that ranker cluster 1 is the stronger cluster, in that these rankers are more able to distinguish between entities. The dendograms in Figure 4.9 suggest that there are five entity clusters within ranker cluster 1 (taking dissimilarity $\in (0.58, 0.83)$) and three entity clusters in ranker cluster 2 (taking dissimilarity $\in (0.50, 0.69)$). Notice that in ranker cluster 1, the most preferred entity in this cluster (entity 1) has its own cluster, and entities 16 and 8 (in true entity clusters 7 and 8) also form a single cluster; perhaps these are not surprising given the “true” values of the skill parameters for these entities within this ranker

Cluster	1	2	3	4	5	6	7	8	9	≥ 10
1	0.00	0.00	0.00	0.07	0.20	0.25	0.21	0.14	0.08	0.05
2	0.00	0.11	0.25	0.26	0.18	0.11	0.05	0.02	0.01	0.01

Table 4.5: Posterior distribution of the number of entity clusters, conditional on two ranker clusters.

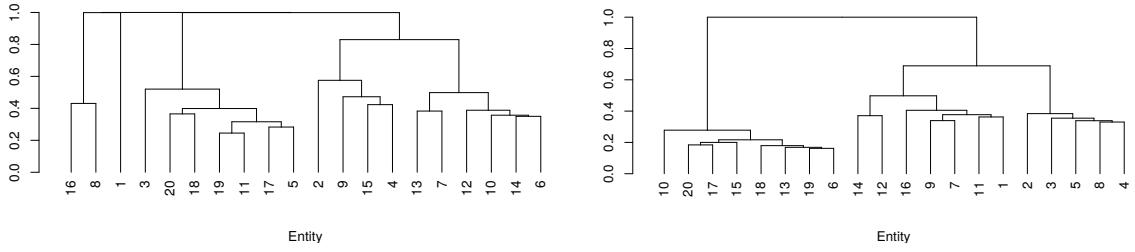


Figure 4.9: Dendrograms showing the dissimilarity between entities within ranker clusters 1 (left) and 2 (right), conditional on two ranker clusters ($N^r = 2$).

cluster (see Table 4.4). True entity cluster 6 is fairly well identified with only entity 14 not being included and entity 3 (from true cluster 4) joining the cluster. The remaining two entity clusters identified by the dendrogram house the other entities from true entity clusters 2–5. Within ranker cluster 2 the “true” entity clustering structure from which the data were simulated is largely preserved but the inferred clusters are groups of the “true” clusters, with all entities in “true” cluster 1 being clearly identified in one cluster and those in clusters 2, 3 and 4 in another cluster and those in clusters 5 and 6 in another cluster. That these entity clusters have merged is perhaps not too surprising given the true values (see Table 4.4) and the limited number of rankings observed.

We now investigate the preference ordering of the entities within each ranker group and an overall preference ordering; see Table 4.6. Here the preference ordering within each ranker group has been determined by the posterior mean of the “skill” parameters, averaged over both the entity clustering and the allocation of rankers to each ranker group. The overall preference ordering has been further averaged over all ranker clusters. Comparing these preference orderings with the truth (in Table 4.4) we see that the WAND model has performed fairly well in recovering the true preferences expressed in ranker clusters 1 and 2, especially for those entities which are the most and least preferred within each ranker group. Not surprisingly there is an increased level of misidentification in the middle ranks of the preference ordering for both ranker clusters, and particularly so for ranker cluster 1. This is perhaps due, in part, to the true values of the skill parameters in entity clusters 2–6 within each ranker cluster being fairly similar, with those in ranker cluster 1 being the most similar; see Table 4.4.

The entities in Table 4.6 are listed in order of their overall “true” skill parameter. Even though the WAND model has allowed for differences between rankers, the inferred overall ordering is very different from the “true” order. That said, the inferred orderings within the ranker clusters are very similar to the “true” orderings and give a much better account of the heterogeneity within the model underpinning the data. This illustrates how inferring preference orderings using overall (population level) summaries of heterogeneous rankers

Rank	Entity	C_1^r		C_2^r		Aggregate	
		Mean (SD)	Entity	Mean (SD)	Entity	Mean (SD)	Entity
1	1	3.54 (1.57)	2	1.91 (1.11)	1	2.93 (1.15)	
2	7	1.02 (0.54)	8	1.77 (1.08)	7	1.13 (0.47)	
3	13	1.02 (0.54)	5	1.73 (1.07)	4	1.02 (0.41)	
4	10	0.91 (0.46)	4	1.71 (1.06)	2	0.97 (0.40)	
5	14	0.89 (0.45)	3	1.65 (1.05)	14	0.96 (0.40)	
6	6	0.87 (0.44)	16	1.51 (1.02)	12	0.95 (0.39)	
7	12	0.83 (0.42)	1	1.44 (0.99)	9	0.89 (0.38)	
8	15	0.76 (0.39)	7	1.41 (0.98)	3	0.82 (0.35)	
9	4	0.75 (0.40)	9	1.36 (0.96)	13	0.80 (0.39)	
10	9	0.70 (0.38)	11	1.31 (0.95)	5	0.78 (0.34)	
11	2	0.60 (0.35)	12	1.23 (0.92)	10	0.73 (0.33)	
12	3	0.49 (0.28)	14	1.13 (0.87)	6	0.69 (0.31)	
13	20	0.44 (0.24)	10	0.25 (0.21)	11	0.64 (0.29)	
14	18	0.44 (0.24)	17	0.22 (0.17)	15	0.62 (0.29)	
15	17	0.41 (0.21)	15	0.22 (0.16)	8	0.52 (0.31)	
16	5	0.41 (0.21)	20	0.22 (0.16)	16	0.47 (0.29)	
17	11	0.37 (0.19)	13	0.21 (0.15)	20	0.39 (0.18)	
18	19	0.37 (0.18)	19	0.21 (0.15)	18	0.38 (0.18)	
19	16	0.05 (0.07)	6	0.21 (0.15)	17	0.36 (0.16)	
20	8	0.03 (0.08)	18	0.20 (0.14)	19	0.33 (0.14)	

Table 4.6: Posterior preference orderings within ranker clusters 1 and 2 (conditional on two ranker clusters) and the overall/aggregate ranking, with mean (and standard deviation) of their skill parameters.

can be very misleading even when knowing the skill parameters, let alone when attempting to infer their values.

4.8 Summary

In this chapter we have described the Adapted Nested Dirichlet process prior which facilitates two-way clustering on both rankers and entities (within ranker groups). We then used this prior to form the WAND model by taking the underlying ranking distribution to be the Weighted Plackett–Luce model. Two approaches to inference for the WAND model were then considered. In Section 4.5 we appealed to a conditional sampling approach. Although intuitive, this approach to inference for DP mixtures comes with drawbacks when compared to marginal sampling schemes, as discussed in Chapter 3. In Section 4.6 we discussed how a marginal scheme for posterior sampling can be constructed for this adaptation of the NDP. The marginal posterior sampling scheme we outlined in Section 4.6.5 allows for fast and efficient inference under our WAND model.

We saw through the simulation studies in Section 4.7 that reasonable inferences can be made under the WAND model even when only limited (partial) information is available.

The richness of information in the posterior distribution allows us to infer many details of the structure both between ranker groups and between entity groups (within ranker groups). The high dimension of the posterior distribution can make the production of insightful but simple summaries quite difficult and we have explored different approaches, ranging from conditioning on modal number of groups to adopting a classification based on calculations from a dissimilarity matrix summary.

In the next chapter we consider two real datasets that have been analysed in the literature, and compare their conclusions with those obtained from fitting the WAND model.

Chapter 5

Real data analyses

5.1 Roskam's data set

In this section we consider a dataset originally collected in 1968 by Roskam, more recently studied by de Leeuw (2006). The data are available in the R package *homals* (de Leeuw and Mair, 2009) and are also given in Table B.6 within the appendices. The data consist of rankings obtained from $n = 39$ psychologists within the Psychology Department at the University of Nijmegen (Netherlands). Each ranker gives a complete ranking of $K = 9$ sub-areas (entities), listed according to how appropriate the sub-areas are to their work. The sub-areas are: SOC - Social Psychology, EDU - Educational and Developmental Psychology, CLI - Clinical Psychology, MAT - Mathematical Psychology and Psychological Statistics, EXP - Experimental Psychology, CUL - Cultural Psychology and Psychology of Religion, IND - Industrial Psychology, TST - Test Construction and Validation, and PHY - Physiological and Animal Psychology.

The heterogeneity within these data has been analysed by de Leeuw (2006) using a non-linear principal component analysis to detect groupings within the rankings. Their analysis supported the idea that there are two groups of rankings: one group which favours the qualitative fields and the other favouring the quantitative fields of psychology. A homogeneity analysis was later performed by de Leeuw and Mair (2009) which exposed groupings of entities within the rankings. More recently Choulakian (2016) performed a Taxicab correspondence analysis to look at structure both between the rankings and the entities within ranker groups. Their results support the conclusions of de Leeuw (2006) and suggest that the psychologists comprise two homogeneous groups with 23 and 16 members respectively. Within the larger ranker group they obtain the entity clustering $\{\text{MAT}, \text{EXP}\} \succ \{\text{IND}, \text{TST}\} \succ \{\text{PHY}, \text{SOC}, \text{EDU}\} \succ \text{CLI} \succ \text{CUL}$, where \succ means “is preferred to”, and quantitative areas of psychology appear to be preferred. The corresponding clus-

tering of entities for the other ranker group is $\{\text{EDU}, \text{CLI}, \text{SOC}\} \succ \{\text{CUL}, \text{MAT}, \text{EXP}\} \succ \{\text{TST}, \text{IND}\} \succ \text{PHY}$, and here qualitative areas of psychology appear to be preferred. They also conclude that the larger ranker group is somewhat more homogeneous than the smaller group.

We now use our WAND model to investigate subgroup structure in these data and take our prior specification for the base distribution and concentration parameters to be $a = 1$ and $a_\alpha = b_\alpha = 1$, $a_\gamma = b_\gamma = 3$. These data contain orderings of individual preferences which we believe to be informative and so take $p_i = 0.75$. The posterior distribution is fairly robust to this choice; a sensitivity analysis follows in Section 5.1.1. We report the results from a typical run of our MCMC scheme initialised from the prior, with a burn-in of 10K iterations and then run for a further 1M iterations and thinned by 100 to obtain 10K (almost) un-autocorrelated realisations from the posterior distribution. Convergence was assessed by using multiple starting values, inspection of traceplots of parameters and the log complete data likelihood, and standard statistics available in the R package *coda* (Plummer et al., 2006). The MCMC scheme runs fairly quickly, with C code on a single thread of an Intel Core i7-4790S CPU (3.20GHz clock speed) taking around 5 minutes.

Table 5.1 shows both the prior and posterior distribution for the number of ranker clusters. The data clearly have been informative and suggest that it is likely that there are between two and four ranker groups, with two groups being most plausible. Note that there is almost no posterior support to suggest there is a single (homogeneous) ranker group and so an aggregate ranking from this dataset may be misleading. Figure 5.1 shows the dendrogram of rankers along with the posterior probability that each ranker is informative. The dendrogram suggests that there are two ranker groups (taking dissimilarity > 0.60), and this is consistent with the posterior distribution in Table 5.1 and the conclusions of previous analyses. We note that the data are consistent with most rankers being informative (with $\Pr(w_i = 1 | \mathcal{D}) \geq 0.8$), an increase from their prior probabilities ($p_i = 0.75$). Also the rankers whose probabilities have decreased (rankers 1, 5, 8, 10, 13, 14, 15, 31) are those with (slightly) different preferences and hence late to join the right-hand cluster in the dendrogram.

We now turn to the subgroup structure of entities within the ranker clusters, and here we condition on there being two ranker clusters. Figure 5.2 shows the (marginal) posterior distribution for the number of entity clusters within each ranker cluster together with the

	1	2	3	4	5	6	7	≥ 8
Posterior	0.00	0.43	0.33	0.16	0.06	0.02	0.00	0.00
Prior	0.20	0.18	0.16	0.13	0.10	0.08	0.05	0.10

Table 5.1: Prior and posterior distribution of the number of ranker clusters (to 2 d.p.).

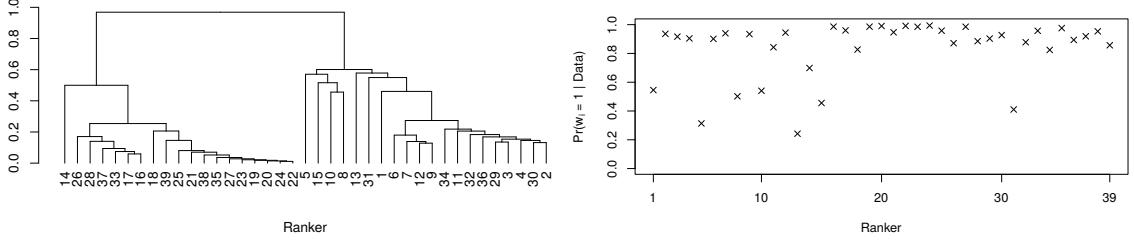


Figure 5.1: Roskam's dataset: Dendrogram (left) showing the ranker cluster structure along with the posterior probability, $\Pr(w_i = 1 | \mathcal{D})$, for each ranker i (right).

prior distribution. The dendograms in Figure 5.3 show the entity clustering structure in each ranker cluster. We define entity clusters at dissimilarities in ranges $(0.45, 0.95)$ and $(0.63, 0.89)$ for rankers groups 1 and 2 respectively and form a preference ordering of these entity clusters by examining the marginal posteriors for the skill parameters $\lambda_{c_i d_{c_i j}}$ within each ranker group c_i . Conditioning on these allocations to both ranker and entity groups and ordering by posterior mean, we obtain $\{\text{EXP, MAT}\} \succ \{\text{TST, PHY, IND}\} \succ \{\text{EDU, SOC, CLI}\} \succ \{\text{CUL}\}$ (with entity cluster means $3.02, 0.72, 0.22, 0.06$) in ranker cluster 1 and $\{\text{SOC, EDU, CLI, MAT}\} \succ \{\text{CUL, IND, EXP, TST}\} \succ \{\text{PHY}\}$ (with entity cluster means $1.96, 0.82, 0.12$) in ranker cluster two. These entity clusters (within ranker groups) are similar to those given by Choulakian (2016). Also if we use the average value of $\Pr(w_i = 1 | \mathcal{D})$ as a measure of homogeneity within a ranker cluster then we obtain 0.68 and 0.56 for clusters 1 and 2 respectively, which again agrees with the Choulakian (2016) conclusion that ranker cluster 1 is more homogeneous than ranker cluster 2. Note that, for this data analysis, we obtain a very similar entity ordering using marginal posterior means of the skill parameters within each ranker group (marginal over the distribution of entity clusters); see Table 5.2. Indeed the table suggests that the ranker groups almost have opposite (reverse) preferences to each other.

We looked at the sensitivity of the posterior distribution (and inferences) to modest changes to the prior distribution. The posterior distribution was fairly insensitive to

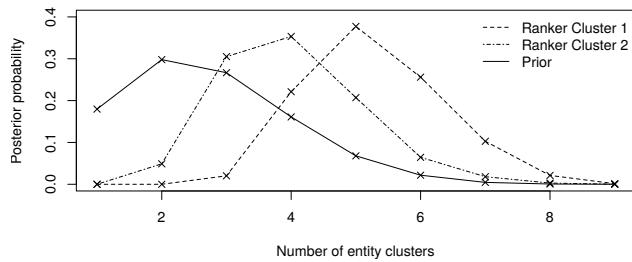


Figure 5.2: Prior and marginal posterior densities for the number of entity clusters within each ranker cluster (conditional on two ranker clusters).

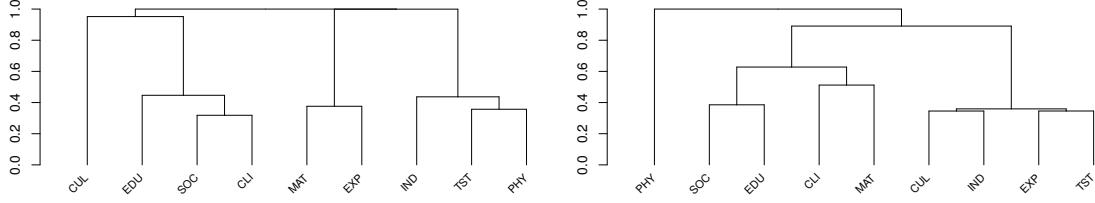


Figure 5.3: Roskam's dataset: Dendograms showing the entity clustering structure within ranker cluster 1 and 2 (left and right respectively) conditional on two ranker clusters.

Ranker cluster	Rank								
	1	2	3	4	5	6	7	8	9
1	EXP	MAT	TST	PHY	IND	EDU	SOC	CLI	CUL
	3.13	2.68	0.76	0.70	0.63	0.27	0.22	0.20	0.07
2	SOC	EDU	CLI	MAT	CUL	IND	EXP	TST	PHY
	1.95	1.75	1.49	1.32	0.94	0.90	0.87	0.87	0.10

Table 5.2: Roskam's dataset: entity rankings by posterior mean within ranker cluster (conditional on two ranker clusters). Rank 1 corresponds to the entity most preferred within each cluster.

changes in the index (a) of the gamma base distribution and to changes in the parameters (a_α , b_α , a_γ , b_γ) of the gamma prior distributions for the concentration parameters. The posterior distribution was most sensitive to changes in the prior probabilities (p_i) of rankers being informative. Not surprisingly most affected by such changes were their posterior equivalents $\Pr(w_i = 1|\mathcal{D})$ though the conclusion of two ranker groups and the membership of these groups was robust. The allocation of entities to groups (within each ranker cluster) was also fairly robust, with only a minor change in the allocation in the $p = 0.85$ case. Section 5.1.1 contains the (ranker and entity) dendograms and plots of $\Pr(w_i = 1|\mathcal{D})$ for $p_i = 0.65$ and $p_i = 0.85$ in addition to the choice $p_i = 0.75$ used in this analysis.

5.1.1 Prior sensitivity analysis

Here we look at the sensitivity of the posterior distribution to changes in the prior probability that a ranker is informative. We consider two alternative choices to the one used in our previous analysis ($p_i = 0.75$), namely $p_i = 0.65$ and $p_i = 0.85$. For ease of reference we also include the results for the $p_i = 0.75$ case.

Overall, we found that the posterior distribution was fairly robust to the choice of p_i *a priori*. Perhaps unsurprisingly the aspect of our posterior distribution most sensitive to changes in the p_i was their posterior equivalents $\Pr(w_i = 1|\mathcal{D})$; see Figure 5.4 (right column). However we note that the rankers whose informative probability decreases (prior

\rightarrow posterior) remain the same in each case: these are rankers $\{1, 5, 8, 10, 13, 14, 15, 31\}$. The dendrograms of the ranker clustering structure are similar for each prior choice and clearly indicate that there are two groups of rankers. Also the allocation of rankers to clusters is similar in each case; see Figure 5.4 (left column). Interestingly we observe increasing posterior support for two ranker clusters as the p_i decrease – this is also the posterior mode in each case; see Figure 5.5. In addition, conditional on there being two ranker clusters, the marginal posterior of the number of entity clusters N_s^e (within each ranker cluster $s = 1, 2$) remains fairly robust to the prior choice; see Figure 5.5. Also the dendrograms of the entity clustering structure are similar in each case; see Figure 5.6.

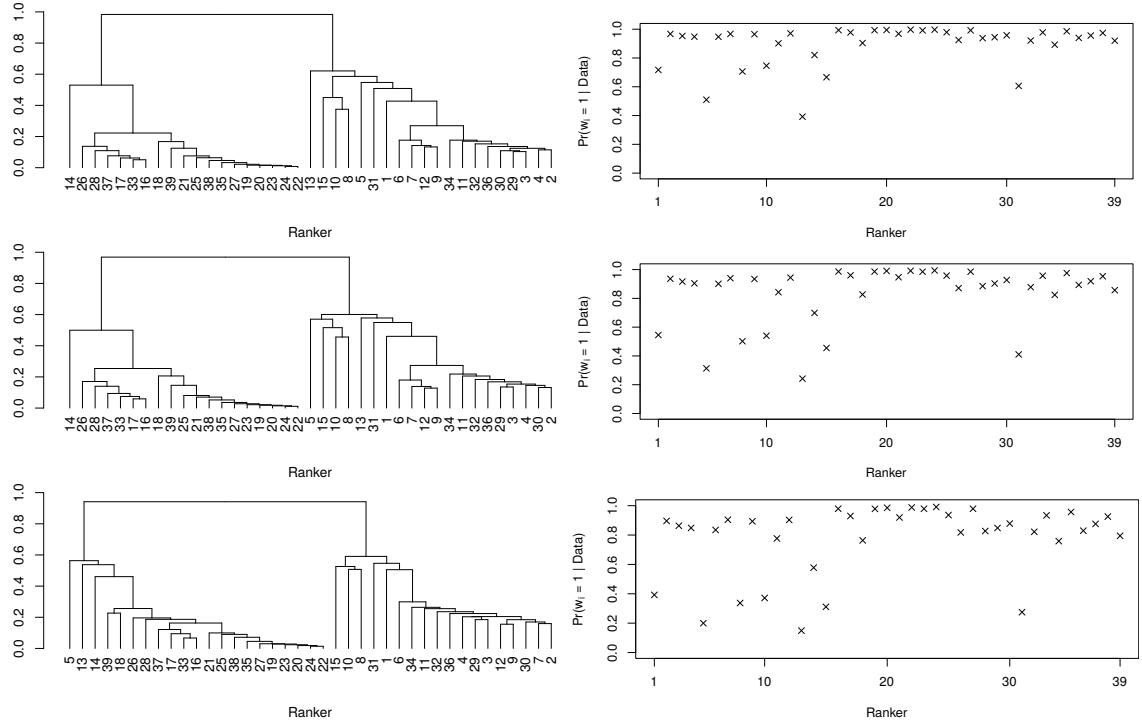


Figure 5.4: Roskam's dataset: Dendrogram (left) showing the cluster structure of the rankers along with the posterior probability $\Pr(w_i = 1 | \mathcal{D})$ for each ranking i (right) for $p_i = 0.85, 0.75, 0.65$ (from top to bottom respectively).

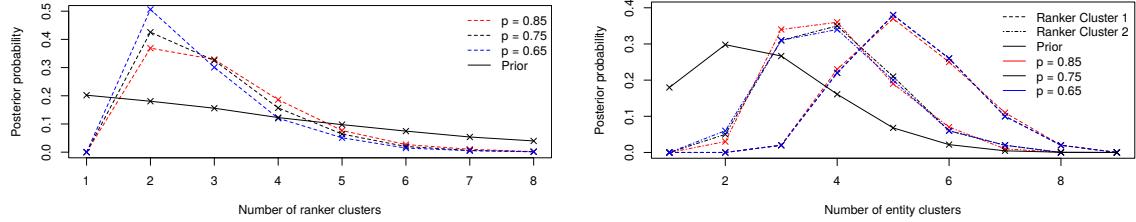


Figure 5.5: Roskam's dataset: Prior and marginal posterior densities for the number of ranker clusters (left plot) and the number of entity clusters within each ranker cluster, conditional on two ranker clusters, (right plot) for $p_i = 0.85, 0.75, 0.65$.

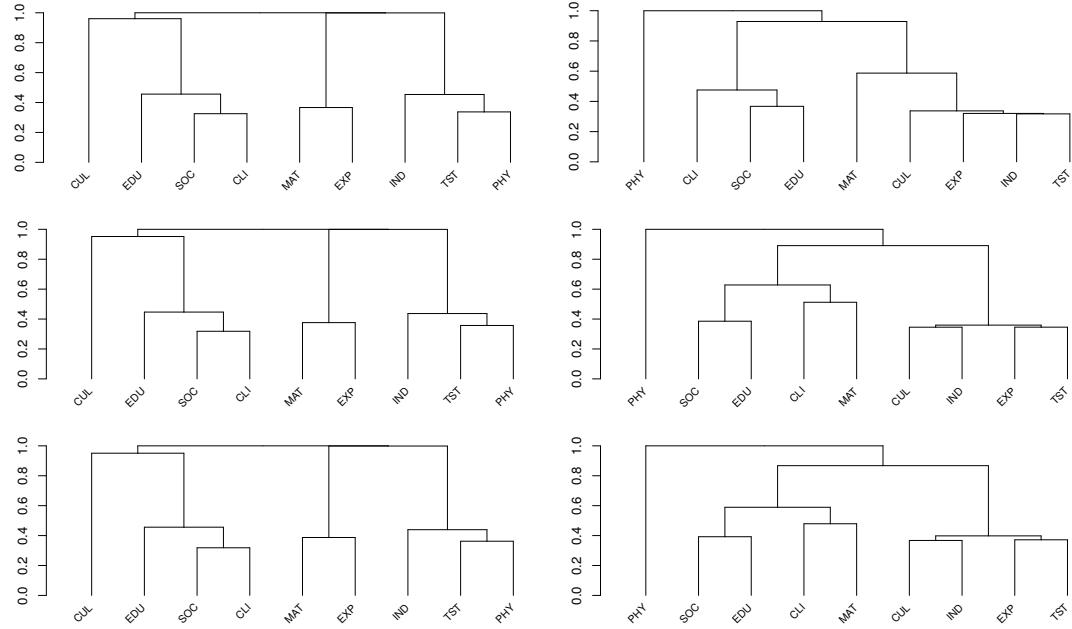


Figure 5.6: Roskam's dataset: Dendograms of entity clustering structure within ranker cluster 1 (left) and ranker cluster 2 (right). These are shown for each prior specification, $p_i = 0.85, 0.75, 0.65$, from top to bottom respectively.

5.2 NBA study

We now consider another dataset of ranks, studied by Deng et al. (2014) and involving rankings of NBA (National Basketball Association) teams. In their paper, Deng et al. propose a model (named “Bayesian Aggregation of Ranked Data”, BARD) which aims to aggregate rankings and identify the “relevant entities”. Their model also accommodates the possibility that rankings may not be equally reliable. One drawback of the BARD model is that it assumes that all rankings come from a single homogeneous group. We now investigate this assumption by using the WAND model and also produce an aggregate ranking to compare with the BARD aggregate ranking.

In 2011/12 the NBA league contained $K = 30$ teams (entities) and the dataset we consider has a ranking of these teams from each of $n = 34$ rankers. The first six complete rankings were obtained from odds given at “professional” websites and the other top–8 rankings obtained from amateurs. Further, each amateur was asked to classify themselves into one of the following groups: “Avid fans” (never missed an NBA game), “Fans” (watched NBA games frequently), “Infrequent watchers” (occasionally watched NBA games) and “Not interested” (never watched an NBA game). Each ranker considered all teams and so we have $K_i = K$ for $i = 1, \dots, n$. The rankers are numbered as follows: Professionals (1–6), Avid fans (7–12), Fans (13–18), Infrequent watchers (19–25) and Not interested (26–34). Therefore we have $n_i = K = 30$ for $i = 1, \dots, 6$ and $n_i = 8$ for $i = 7, \dots, n$. The data are given in Table B.7 within the appendices. Further details on how these data were collected can be found in Deng et al. (2014).

We now analyse these data using our WAND model and see whether it is plausible that these rankers are homogeneous or whether the self-assessed groups behave differently. We take the same prior for the base distribution ($a = 1$) as in the previous example. However, to reflect weak prior beliefs that there are several ranker groups, we take $a_\alpha = b_\alpha = 3$ in addition to the previous choice for entities, $a_\gamma = b_\gamma = 3$. The prior we choose for each ranker’s ability is based on how much attention they reportedly pay to the NBA, with professional rankers likely to be most informative, followed by the Avid fans, then Fans and so on. We do this by giving the same p_i -value for each ranker in the same “ability” group, with $p_i = 0.9$ for professionals, $p_i = 0.7$ for Avid fans, $p_i = 0.5$ for Fans, $p_i = 0.3$ for Infrequent watchers and $p_i = 0.1$ for Not interested. We appreciate that, in general, this type of information is unlikely to be available to the analyst and so in Section 5.2.1 we consider an analysis where $p_i = 0.5$ for each ranker (a sensible choice when no information is available). Generally, we found that the posterior distribution is fairly robust to the choice of p_i *a priori* which is perhaps not surprising given what we have seen in the previous simulation studies and (other) real data analysis. Unsurprisingly the aspect of the posterior distribution most sensitive to changes in the p_i was their posterior

equivalents $\Pr(w_i = 1 | \mathcal{D})$ although the inferences under each analysis were robust.

As in the previous analysis, we report the results from a typical run of our MCMC scheme initialised from the prior, with a burn-in of 10K iterations and then run for a further 1M iterations and thinned by 100 to obtain 10K (almost) un-autocorrelated realisations from the posterior distribution. Convergence was assessed by using multiple starting values, inspection of traceplots of parameters and the log complete data likelihood, and standard statistics available in the R package *coda*. Again the MCMC scheme runs reasonably quickly, with C code on a single thread of an Intel Core i7-4790S CPU (3.20GHz clock speed) taking just under 18 minutes.

Our analysis of the posterior realisations reveals very little posterior support for a single homogeneous group of rankers, with most support for two ranker groups ($\Pr(N^r = 1 | \mathcal{D}) = 0.00$, $\Pr(N^r = 2 | \mathcal{D}) = 0.80$ and $\Pr(N^r = 3 | \mathcal{D}) = 0.17$). Figure 5.7 (left) shows a dendrogram of the posterior clustering structure of rankers and confirms the conclusion that there are two distinct groups of rankers: one with rankers 1–10, 12, 15 and the other with rankers 11, 14, 17–26, 28 and 32. Nearly all the other rankers are classed as uninformative, with $\Pr(w_i = 1 | \mathcal{D}) < 0.25$, except informative ranker 16 who is (roughly) equally likely to be allocated to each cluster; see Figure 5.7 (right). Note that obtaining a clustering by using the MAP allocation would be misleading as the MAP allocation occurs in only 60 of the 10K iterations in the MCMC chain. Unsurprisingly, uninformative rankers are typically those who pay less attention to the NBA, with average values of $\Pr(w_i = 1 | \mathcal{D})$ for rankers in the self-certified groups (from professionals down to the not interested individuals) of 1, 1, 0.87, 0.88, 0.34 respectively. A similar conclusion was found under BARD through its ranking quality parameters; see Figure 8 in Deng et al. (2014).

Figure 5.8 shows the marginal posterior distribution for the number of entity clusters within each ranker cluster (conditional on there being two ranker clusters) together with the prior distribution. The posterior mean number of entity clusters for ranker clusters 1 and 2 is 8.88 and 4.58 respectively, with corresponding standard deviations 1.55 and 1.29.

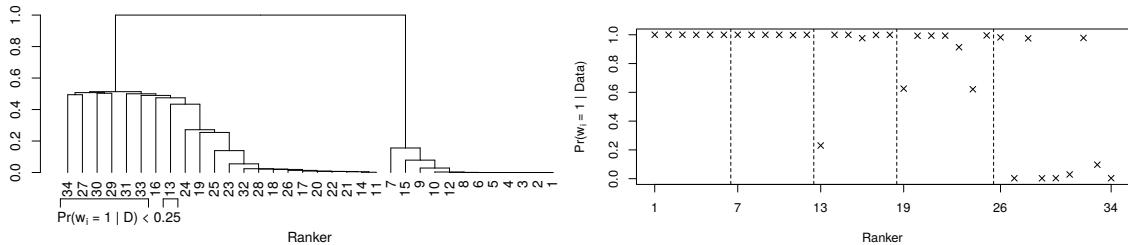


Figure 5.7: NBA dataset: Dendrogram (left) showing the clustering structure of rankers and highlighting those rankers with $\Pr(w_i = 1 | \mathcal{D}) < 0.25$. Plot (right) of the posterior probabilities $\Pr(w_i = 1 | \mathcal{D})$ for each ranker, with vertical lines separating the self-certified groups.

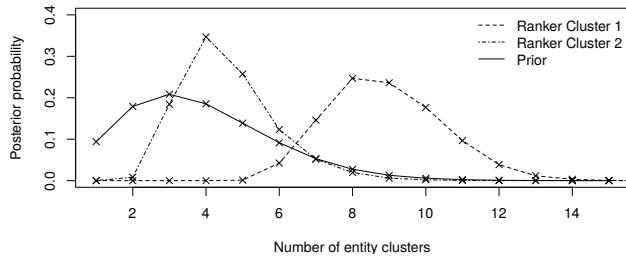


Figure 5.8: Prior and marginal posterior densities for the number of entity clusters within each ranker cluster (conditional on two ranker clusters).

These distributions suggest that rankers within cluster 1 are able to distinguish between many more entities than those in cluster 2. Again this should come as no surprise as ranker cluster 2 mainly consists of rankers who typically pay little attention to the NBA. The dendrograms in Figure 5.9 show the entity clustering in each ranker cluster, and suggest that there are six distinct entity clusters within ranker cluster 1 (taking dissimilarity > 0.81) and three entity clusters in ranker cluster 2 (taking dissimilarity > 0.61). We note that the MAP clustering gives six and two entity clusters respectively, though there are relatively few MCMC iterations contributing to the MAP allocation for either cluster.

It is also of interest to look at the differences in preferences between the two ranker clusters by examining the within-cluster aggregate rankings; see Table 5.3. As before, these are determined by the marginal posterior mean for each entity (within each ranker cluster). The horizontal lines in this table show the MAP entity clustering described above and the (quite small) number of occurrences of the MAP is also given. So that our results can be compared to those of Deng et al. (2014), the table also includes the overall aggregate ranking, determined by ordering the mean of the (fully) marginal posterior distribution for each entity (marginalised over ranker clusters).

The entity rankings in ranker cluster 1 strongly favour the Heat (entity 1) and Thunder (2), with the Bulls (10) as the 3rd most preferred team. Ranker cluster 2 also favours the Heat but differs in their preferences for second and third positions – here being the

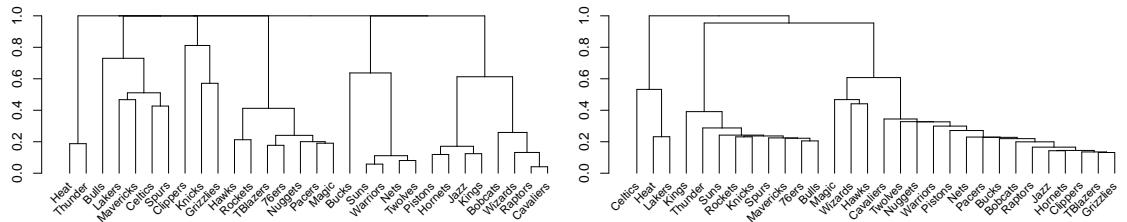


Figure 5.9: NBA dataset: Dendrograms showing the entity cluster structure within ranker clusters 1 and 2 (left and right respectively) conditional on two ranker clusters.

Rank	Ranker cluster 1		Ranker cluster 2		Aggregate	
	Entity	Mean (SD)	Entity	Mean (SD)	Entity	Mean (SD)
1	1	5.63 (2.17)	1	3.18 (1.73)	1	Heat 4.35 (1.38)
2	2	5.22 (2.38)	6	3.03 (1.77)	2	Thunder 2.61 (1.20)
3	10	1.48 (1.24)	4	2.23 (1.73)	6	Lakers 1.99 (0.95)
4	6	0.92 (0.54)	8	0.20 (0.16)	4	Celtics 1.52 (0.92)
5	9	0.86 (0.54)	10	0.20 (0.16)	10	Bulls 0.81 (0.57)
6	4	0.75 (0.44)	9	0.19 (0.16)	9	Mavericks 0.52 (0.26)
7	3	0.74 (0.43)	3	0.19 (0.15)	3	Spurs 0.46 (0.22)
8	5	0.53 (0.36)	18	0.19 (0.15)	5	Clippers 0.28 (0.17)
9	11	0.32 (0.23)	11	0.18 (0.15)	11	Knicks 0.26 (0.13)
10	12	0.20 (0.16)	20	0.18 (0.14)	8	76ers 0.13 (0.09)
11	7	0.05 (0.04)	2	0.17 (0.15)	18	Rockets 0.12 (0.08)
12	13	0.05 (0.04)	26	0.15 (0.13)	12	Grizzlies 0.12 (0.08)
13	14	0.05 (0.04)	14	0.12 (0.12)	20	Suns 0.10 (0.08)
14	17	0.05 (0.03)	27	0.10 (0.10)	14	Magic 0.09 (0.07)
15	8	0.04 (0.03)	15	0.09 (0.10)	26	Kings 0.08 (0.07)
16	15	0.04 (0.03)	29	0.07 (0.08)	15	Hawks 0.07 (0.06)
17	18	0.03 (0.02)	23	0.07 (0.08)	13	Nuggets 0.07 (0.05)
18	19	0.02 (0.02)	13	0.07 (0.08)	7	Pacers 0.06 (0.04)
19	20	0.00 (0.00)	22	0.06 (0.07)	27	Wizards 0.05 (0.06)
20	22	0.00 (0.00)	25	0.06 (0.07)	17	TBlazers 0.05 (0.03)
21	21	0.00 (0.00)	21	0.05 (0.06)	23	Twolves 0.04 (0.05)
22	23	0.00 (0.00)	7	0.05 (0.06)	29	Cavaliers 0.04 (0.05)
23	25	0.00 (0.00)	19	0.05 (0.06)	19	Bucks 0.04 (0.04)
24	24	0.00 (0.00)	30	0.05 (0.06)	22	Warriors 0.04 (0.04)
25	16	0.00 (0.00)	28	0.05 (0.05)	25	Pistons 0.04 (0.04)
26	26	0.00 (0.00)	16	0.04 (0.04)	21	Nets 0.03 (0.04)
27	27	0.00 (0.00)	24	0.04 (0.05)	30	Bobcats 0.03 (0.03)
28	28	0.00 (0.00)	5	0.04 (0.04)	28	Raptors 0.03 (0.03)
29	29	0.00 (0.00)	17	0.04 (0.04)	16	Jazz 0.03 (0.03)
30	30	0.00 (0.00)	12	0.04 (0.04)	24	Hornets 0.03 (0.03)

MAP: 24

MAP: 67

Table 5.3: NBA analysis: Posterior preference orderings within ranker clusters 1 and 2 (conditional on two ranker clusters) and the overall/aggregate ranking, with mean (and standard deviation) of their skill parameters. The horizontal lines indicate the MAP entity clustering within ranker clusters. The numbers at the bottom are the number of occurrences in which the MAP clustering was observed (out of 8038 iterations with two rankers clusters).

Lakers (6) and Celtics (4). There are many differences in preference orderings between the ranker clusters, for example, the Thunder and Bulls appear in positions 11 and 5 in ranker cluster 2.

The analysis given in Deng et al. (2014) looks at the so-called “relevant entities”, defined to be those entities within the top–16, and concludes that these are $\{1, 2, \dots, 15, 18\}$. The overall aggregate ranking reported under our WAND model gives the top–16 as $\{1, 2, \dots, 6, 8, \dots, 12, 14, 15, 18, 20, 26\}$; see Table 5.3. Perhaps surprisingly, despite the

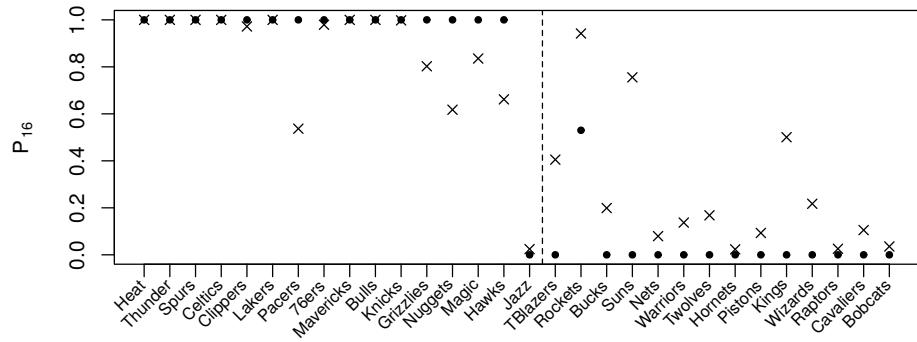


Figure 5.10: The probability P_{16} that each entity is in the top–16 under the WAND (\times) model, and the probabilities that each entity is a relevant entity under BARD (\cdot). The vertical line separates out the teams that actually reached the top–16 playoffs.

BARD analysis assuming only a single ranker cluster, there is considerable overlap between the WAND and BARD top–16 lists – the differences being that entities 20 and 26 feature in our list whereas entities 7 and 13 are omitted, with entities 7 and 13 just missing out from our top–16 and appearing in positions 18 and 17. However, this can be explained by the WAND overall aggregate ranking being formed by a consensus between the very discriminating ranker cluster 1 and the much less discriminating ranker cluster 2.

If we now compare the BARD top–16 with the rankings in the WAND ranker clusters, we see that the entity rankings in ranker cluster 1 are consistent with the BARD results, with the only differences being that entity 18 is ranked 17th and entity 17 moves into the top–16. The entity rankings in ranker cluster 2 are much less consistent with the BARD results, and this is partially explained by the larger uncertainty on entity positions within this cluster. The closeness of the entity posterior means (in ranker cluster 2) helps to explain this level of rank uncertainty as these rankers clearly struggle to distinguish between entities.

The BARD analysis also reports a probability for each entity being a relevant entity which is similar to the probability P_{16} that each entity is in the top–16 under the WAND model. The values for these probabilities under both BARD and WAND models are shown in Figure 5.10; the vertical dashed line separates entities 1–16 from the remainder, that is, separates the teams that actually reached the top–16 playoffs that season from the others. It is interesting to see that the WAND model places much more uncertainty on many top–16 teams than under BARD, in the sense that their P_{16} values are smaller – BARD values are essentially zero or one.

5.2.1 Prior sensitivity analysis

In Section 5.2 we used the (self declared) ranker abilities to form a relatively “informative” prior on the ranker weights w_i . In general this information is unlikely to be available to the analyst and so here we look at the sensitivity of the posterior distribution to changes in the prior probability that a ranker is informative. From our experience using WAND we have found that, in a scenario where there is little information regarding the ranker abilities, it is best to make conservative choices of p_i *a priori*. We consider an alternative choice to the “staggered” p_i used previously and let $p_i = 0.5$ for all i so that each ranker is equally likely to be informative/uninformative and compare the posterior under each analysis.

Generally, we found that the posterior distribution is fairly robust to the choice of p_i *a priori* which is perhaps not surprising given what we have seen in the previous simulation studies and (other) real data analysis. As before, it came as no surprise that the aspect of the posterior distribution most sensitive to changes in the p_i was their posterior equivalents $\Pr(w_i = 1|\mathcal{D})$; see Figure 5.11 (right column). Unsurprisingly the largest discrepancies are for those rankers whose prior p_i has been increased the most (rankers 19–25 and 26–34). We note however that the rankers who obtain $\Pr(w_i = 1|\mathcal{D}) < 0.25$ are similar under both analyses with only rankers 31 and 33 no longer under this threshold for the $p_i = 0.5$ analysis (note $\Pr(w_{31} = 1|\mathcal{D}) = 0.26$). The dendograms of the ranker clustering structure are similar for each prior choice and clearly indicate that there are two groups of rankers. Also the allocation of rankers to clusters is similar in each case; see Figure 5.11 (left column). This is further supported by the marginal posterior distribution of the number of ranker groups which shows that these data have clearly been informative and suggest 2 ranker groups under both prior choices; see Figure 5.12 (left). Interestingly, conditional on there being two ranker clusters, the marginal posterior of the number of entity clusters N_s^e (within each ranker cluster $s = 1, 2$) suggests there are slightly more entity clusters within ranker group 2 under the $p_i = 0.5$ analysis; see Figure 5.12 (right). This is perhaps an artifact of additional information being available within this cluster as $\Pr(w_i = 1|\mathcal{D})$ has increased for some of the rankers within this group. The dendograms of the entity clustering structure in ranker group 1 are similar in each case; see Figure 5.13 (left column). Although the corresponding dendograms of the entity clustering for ranker group 2 may appear to be slightly different for each analysis upon closer inspection it becomes clear that only the Magic and the Wizards have changed association from the right hand group to the central group for the $p_i = 0.5$ analysis.

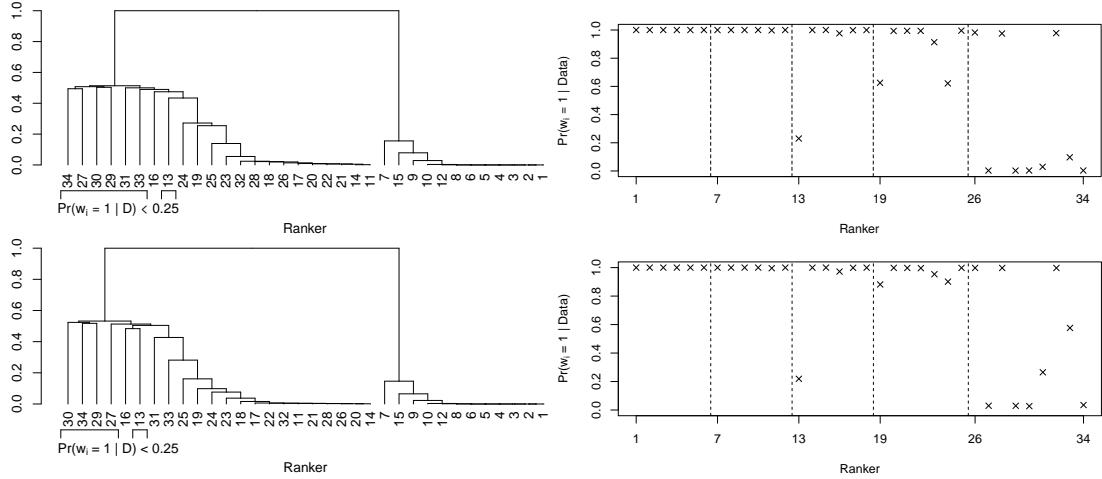


Figure 5.11: NBA dataset: Dendrogram (left) showing the clustering structure of rankers and highlighting those rankers with $\Pr(w_i = 1 | \mathcal{D}) < 0.25$. Plot (right) of the posterior probabilities $\Pr(w_i = 1 | \mathcal{D})$ for each ranker, with vertical lines separating the self-certified groups. The top row shows the results for the “staggered” choice of p_i and the bottom row shows the corresponding results when $p_i = 0.5$ for all rankers.

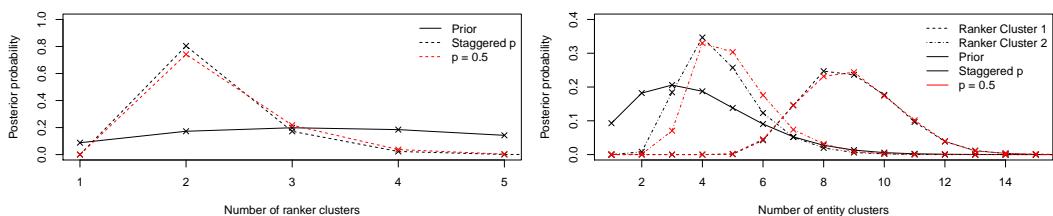


Figure 5.12: NBA dataset: Prior and marginal posterior densities for the number of ranker clusters (left plot) and the number of entity clusters within each ranker cluster, conditional on two ranker clusters, (right plot) for “staggered” choice of p_i and $p_i = 0.5$.

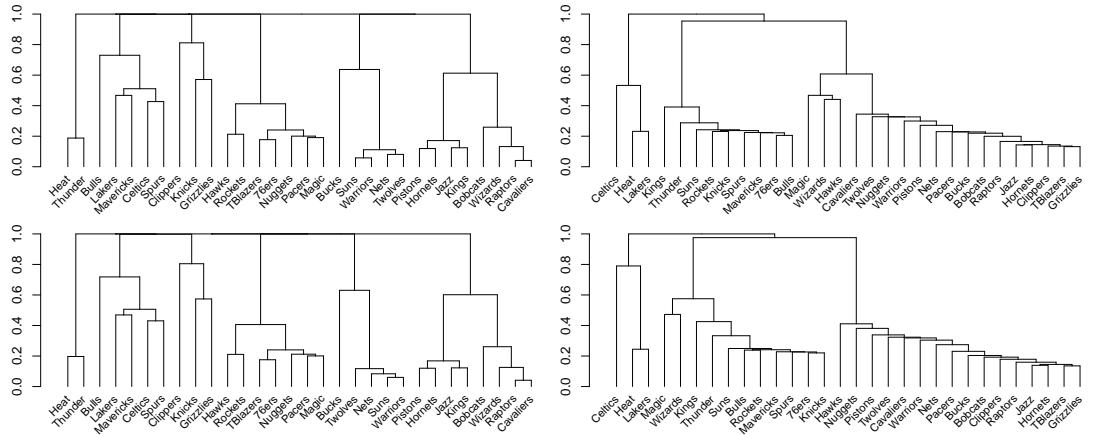


Figure 5.13: NBA dataset: Dendograms showing the entity cluster structure within ranker clusters 1 and 2 (left and right respectively) conditional on two ranker clusters for “staggered” choice of p_i and $p_i = 0.5$, top and bottom respectively.

5.3 Summary

In this chapter we fitted the WAND model to two real datasets that have been previously analysed in the literature. In general we found that the inferences under the WAND model were similar to those obtained under other models. However, the richness of the information within the posterior distribution (under WAND) allows us to infer additional information about the structure between both ranker and entity groups. Our analysis of the NBA data also revealed strong signs of heterogeneity between the rankers’ beliefs about the entities. It follows that the BARD model may not be well suited to these data given the underlying (ranker) homogeneity assumption.

In the next chapter we consider relaxing the assumption of an explicit ranking process by appealing to the Extended Plackett–Luce model (Mollica and Tardella, 2014).

Chapter 6

The Extended Plackett–Luce model

6.1 Introduction

In this chapter we revert to considering homogeneous ranked data and consider the Extended Plackett–Luce model proposed by Mollica and Tardella (2014). This model is an extension to the standard Plackett–Luce model which relaxes the *a priori* assumption of an explicit ranking process. Recall that the standard Plackett–Luce model (and also the Weighted Plackett–Luce model) assumes that each ranker forms their ranking using the *forward ranking process*; see Section 2.2. Here each ranker forms their ranking by first allocating their most preferred entity, then their second most preferred entity and so on until their least preferred entity is allocated last. This is a rather strong assumption. It is easy to imagine a scenario where an individual ranker might assign entities to positions in an alternative way. For example, it is quite plausible that rankers may find it easier to identify their most and least preferred entities first rather than those entities they place in the middle positions of their ranking. In such a scenario rankers might form their ranking by first assigning their most and then least preferred entities to a rank before completing their ranking (by filling out the middle positions) through a process of elimination using the remaining (unallocated) entities, that is, they use a different *ranking process*. The effect of the (assumed) underlying ranking process is somewhat unknown with, to the best of our knowledge, only the standard (forward ranking) Plackett–Luce model and the (backward ranking) Reverse Plackett–Luce model receiving significant attention within the literature. The Extended Plackett–Luce model allows the underlying ranking process to be further explored as it instead allows for all possible ranking processes and allows the data to inform us which is most plausible.

The remainder of this chapter is outlined as follows. We begin with a discussion of the Extended Plackett–Luce model and describe the associated data generating process. In Section 6.2.2 we consider the identifiability of the ranking process and provide some insight as to where the information about the ranking process is contained within the data. The remaining sections focus on inference for the Extended Plackett–Luce model and we consider both maximum likelihood and Bayesian approaches with efficient inference algorithms presented in both cases. Throughout this chapter we perform several simulation studies to demonstrate how insightful inferences can be obtained using the Extended Plackett–Luce model.

6.2 The Extended Plackett–Luce model

Mollica and Tardella (2014) propose the Extended Plackett–Luce (EPL) model which allows the *a priori* assumption of an implicit ranking process to be relaxed. It follows that for this model we can learn about both the (possibly unobserved) underlying ranking process and the parameters of the entities. Before we describe the Extended Plackett–Luce model it is natural to recast the underlying ranking process in terms of a “*choice order*” where the choice order is the order in which rankers assign entities to positions/ranks. For example, a choice order of $(1, K, 2, 3, \dots, K - 1)$ corresponds to the ranking process where a ranker first assigns their most preferred entity, then their least preferred entity before then assigning the remaining entities in rank order from 2nd down. In other words, the rankers choose their most and least preferred entities and then assign the remaining entities using the forward ranking process. Note that the choice order is just a permutation of the ranks 1 to K . The EPL model is defined through the introduction of an additional (free) parameter to represent the choice order within the Plackett–Luce model. Suppose there are K entities and let σ denote the (possibly unknown) choice order then the probability of a particular ranking under the Extended Plackett–Luce model is

$$\Pr(\mathbf{X}_i = \mathbf{x}_i | \boldsymbol{\lambda}^*, \sigma) = \prod_{j=1}^K \frac{\lambda_{x_{i\sigma_j}}^*}{\sum_{m=j}^K \lambda_{x_{i\sigma_m}}^*} \quad (6.1)$$

where $\boldsymbol{\lambda}^* \in \mathbb{R}_{>0}^K$ are the entity parameters and $\sigma \in \mathcal{S}_K$, the set of all possible permutations of the ranks 1 to K . Note that, to maintain notational consistency, we have adopted an alternative representation to that of Mollica and Tardella (2014): here $\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{iK})$ represents the *preference* of the entities reported by ranker i and so, as before, entity x_{i1} is their most preferred entity, x_{i2} is the second most preferred entity and so on. In what follows, the rankings \mathbf{x}_i are often referred to as *preference orderings* to make it clear that we are considering the preference of the entities expressed by ranker i .

irrespective of the choice order. Further, under the Extended Plackett–Luce model the parameters have a different interpretation from those under the standard (forward ranking) Plackett–Luce model (discussed below) and so we let λ^* be the parameters of the entities to make this distinction clear. We also note that the Extended Plackett–Luce model is only well defined for *complete rankings* (Mollica and Tardella, 2014) and so each ranker must provide a preference ordering of *all* the entities. Hence for the Extended Plackett–Luce model we require $n_i = K$ for $i = 1, \dots, n$.

The form of the Extended Plackett–Luce probability is naturally quite similar to that of the standard Plackett–Luce probability. Indeed, the standard and reverse Plackett–Luce models are special cases of the EPL. The standard (forward ranking) Plackett–Luce model is recovered from the EPL model when σ is the identity permutation, that is, when $\sigma_j = j$ for $j = 1, \dots, K$. Also, the (backward ranking) Reverse Plackett–Luce model is obtained when $\sigma = (K, K - 1, \dots, 1)$. Given this it is perhaps now clear that although σ is nominally a model parameter, each unique $\sigma \in \mathcal{S}_K$ defines a *different* Plackett–Luce model.

A key aspect of analysing ranked data using a Plackett–Luce model is the interpretation of the parameters λ^* . Although perhaps not obvious, for the Plackett–Luce model the interpretation of the parameters depends on the underlying ranking process. It follows that for the Extended Plackett–Luce model, the interpretation of the parameters depends on the choice order parameter σ . This becomes clear if we consider an example conditional on known (fixed) choice orders: suppose we have two entities (labelled i, j) with parameters λ_i^* and λ_j^* where $\lambda_i^* > \lambda_j^*$. For the standard (forward ranking) Plackett–Luce model ($\sigma = (1, 2, \dots, K)$) the interpretation is that entity i is preferred to entity j . However, for the (backward ranking) Reverse Plackett–Luce model ($\sigma = (K, K - 1, \dots, 1)$) these parameters are interpreted as entity j being preferred to entity i . It follows that the preference order of entities must be read *with respect to* the choice order. In general, the entity with the largest parameter is the entity most likely to be ranked in position σ_1 . Also, conditional on an entity being assigned to rank σ_1 , the entity with the largest parameter of those remaining is that most likely to be assigned rank σ_2 . Although for the forward and backward ranking processes this leads to a natural interpretation of the skill parameters, the interpretation for other ranking processes can be tricky. For example, again suppose that $\lambda_i^* > \lambda_j^*$, and now consider the choice order to be $\sigma = (5, 3, 2, 1, 4)$. It follows that, for this choice order, entity i is more likely to be ranked fifth than entity j . Further, if another entity $\ell \neq i, j$, is assigned rank 5 then entity i is preferred for rank 3 (σ_2) over entity j . This interpretation is not exactly intuitive. Given the Extended Plackett–Luce model considers all $\sigma \in \mathcal{S}_K$ it would be helpful if we could devise a method for consistently interpreting the preference of entities (via the parameters) irrespective of the choice order.

In what follows we discuss how the probability for the EPL model can be rewritten in such a way so that the parameters maintain a preference order interpretation irrespective of the choice order. Recall that the probability of a particular *preference ordering* \mathbf{x}_i for the EPL model is

$$\Pr(\mathbf{X}_i = \mathbf{x}_i | \boldsymbol{\lambda}^*, \boldsymbol{\sigma}) = \prod_{j=1}^K \frac{\lambda_{x_i \sigma_j}^*}{\sum_{m=j}^K \lambda_{x_i \sigma_m}^*}.$$

Note that the numerator of the j th product $\lambda_{x_i \sigma_j}^*$ is the parameter for the entity in rank σ_j of preference order i . However, to maintain the preference ordering interpretation of the parameters we require that the numerator of the j th product corresponds to the parameter for the entity in rank j as opposed to that in rank σ_j (as above). If we let a ‘permuted ranking’ be $\mathbf{x}^* = \mathbf{x} \circ \boldsymbol{\sigma}$, that is, $x_{ij}^* = x_{i\sigma_j}$ for $j = 1, \dots, K$ then it follows that the probability of a preference ordering \mathbf{x} under the EPL model can be written in terms of the permuted ranking \mathbf{x}^* as

$$\begin{aligned} \Pr(\mathbf{X}_i = \mathbf{x}_i | \boldsymbol{\lambda}^*, \boldsymbol{\sigma}) &= \prod_{j=1}^K \frac{\lambda_{x_i \sigma_j}^*}{\sum_{m=j}^K \lambda_{x_i \sigma_m}^*} \\ &= \prod_{j=1}^K \frac{\lambda_{x_{ij}^*}^*}{\sum_{m=j}^K \lambda_{x_{im}^*}^*}, \end{aligned} \tag{6.2}$$

and this is simply the standard (forward ranking) Plackett–Luce probability defined over the permuted rankings. Indeed we have already seen a special case of this result in Chapter 2 when we noted that the Reverse Plackett–Luce model is equivalent to the forward ranking PL model applied to rankings which have been permuted in to reverse order. Note however that under this representation the parameters $\boldsymbol{\lambda}^*$ must still be interpreted with respect to the choice order as we are analysing the permuted rankings and so the entity in rank 1 of the permuted rankings (the entity most likely to be chosen first) is that which has preference σ_1 . It follows that although the probability has been rewritten the skill parameter $\lambda_{x_{ij}^*}^*$ is still the parameter for the entity in position σ_j of the preference ordering. The consequence of this is that, for this model, the largest parameter will correspond to the entity which has preference σ_1 . Ideally the entity with the largest parameter would be that which has preference 1 (as in the standard Plackett–Luce model). This can be achieved by considering the inverse permutation $\boldsymbol{\sigma}^{-1}$ which is defined so that

$$\boldsymbol{\sigma} \circ \boldsymbol{\sigma}^{-1} = \boldsymbol{\sigma}^{-1} \circ \boldsymbol{\sigma} = \boldsymbol{\sigma}_{\mathcal{I}}$$

where $\boldsymbol{\sigma}_{\mathcal{I}}$ is the identity permutation, that is, $\boldsymbol{\sigma}_{\mathcal{I}} = (1, 2, \dots, K)$. It is therefore clear that $\sigma_{\sigma_1}^{-1} = 1$ by definition. Now let $\boldsymbol{\lambda} = (\lambda_1, \dots, \lambda_K)$ with $\lambda_k = \lambda_{\sigma_k}^*$ for $k = 1, \dots, K$ be a collection of *skill* parameters then, by construction, we have $\lambda_k^* = \lambda_{\sigma_k^{-1}}$ for $k = 1, \dots, K$.

It follows that $\lambda_{x_{ij}^*}^* = \lambda_{\sigma_{x_{ij}^*}^{-1}}^{-1}$ and so we can write the probability of a preference ordering for the Extended Plackett–Luce model in terms of the skill parameters $\boldsymbol{\lambda}$ as

$$\begin{aligned}\Pr(\mathbf{X}_i = \mathbf{x}_i | \boldsymbol{\lambda}, \boldsymbol{\sigma}) &= \prod_{j=1}^K \frac{\lambda_{x_{ij}^*}^*}{\sum_{m=j}^K \lambda_{x_{im}^*}^*} \\ &= \prod_{j=1}^K \frac{\lambda_{\sigma_{x_{ij}^*}^{-1}}^{-1}}{\sum_{m=j}^K \lambda_{\sigma_{x_{im}^*}^{-1}}^{-1}} \\ &= \prod_{j=1}^K \frac{\lambda_{\sigma_{x_{i\sigma_j}}^{-1}}^{-1}}{\sum_{m=j}^K \lambda_{\sigma_{x_{i\sigma_m}}^{-1}}^{-1}},\end{aligned}\tag{6.3}$$

so that the largest skill parameter will be for the entity which has preference 1, as desired. Hence by considering the skill parameters $\boldsymbol{\lambda}$ (as opposed to $\boldsymbol{\lambda}^*$) and using formulation (6.3) we maintain the preference order interpretation of the skill parameters, that is, $\lambda_i > \lambda_j$ indicates that entity i is preferred to entity j (irrespective of the choice order). That said, we make it clear that, although λ_k represents the preference of entity k , our usual intuition about the probabilities they specify over the ranking no longer holds; λ_k no longer represents the probability that entity k is given rank 1, unless $\sigma_1 = 1$.

6.2.1 Simulating data from the Extended Plackett–Luce model

The method for generating data from the Extended Plackett–Luce model is similar to that for the standard (forward ranking) Plackett–Luce model. However there is a subtle but yet important difference. For the standard PL model, the entity which is chosen first is also that which is the most preferred. However, for the Extended Plackett–Luce model this is no longer the case and the entity that is chosen first is instead considered to have rank σ_1 . Recall that $\mathbf{x}^* = \mathbf{x} \circ \boldsymbol{\sigma}$ denotes a permuted ranking where \mathbf{x} is the corresponding preference ordering and $\boldsymbol{\sigma}$ is the choice order. Now noting that, by construction, a permuted ranking \mathbf{x}^* is an ordering of entities according to the *order in which they were chosen* (and not the preference of the entities) it follows that we can simulate these orderings from the standard PL model as they follow the forward ranking process by definition. Then given a (simulated) permuted ranking \mathbf{x}^* we can obtain the corresponding preference order trivially by recalling that $\mathbf{x} = \mathbf{x}^* \circ \boldsymbol{\sigma}^{-1}$.

When simulating data consistent with the Extended Plackett–Luce model additional attention must be placed on the choice of the skill parameters. Although λ_k still corresponds to the strength/ability of entity k , our usual intuition about the probabilities they specify over the ranking no longer holds. Recall that for the standard Plackett–Luce model the probability that entity k is given rank 1 is proportional to λ_k . Now given that the EPL

model is actually the standard Plackett–Luce model defined over the permuted rankings with parameters $\boldsymbol{\lambda}^*$ it follows that the parameter λ_k^* is proportional to the probability that entity k is ranked first within \mathbf{x}^* , or equivalently, the probability that entity k is ranked in position σ_1 of the preference ordering \mathbf{x} is $\Pr(x_{\sigma_1} = k) \propto \lambda_k^*$ for $k = 1, \dots, K$.

A collection $X^* = \{\mathbf{x}_i^*\}_{i=1}^n$ of n permuted rankings using permutation $\boldsymbol{\sigma}$ are generated from the standard (forward ranking) Plackett–Luce data generating mechanism conditional on the parameters $\boldsymbol{\lambda}^*$ as follows.

For $i = 1, \dots, n$,

1. Sample $\nu_{ij} \stackrel{\text{indep}}{\sim} \text{Exp}(\lambda_{ij}^*)$ for $j = 1, \dots, K$.
2. Set $x_{ij}^* = \underset{q \in S_{ij}}{\operatorname{argmin}} \nu_{iq}$ where $S_{ij} = \mathcal{K} \setminus \{x_{i1}^*, \dots, x_{ij-1}^*\}$ for $j = 1, \dots, K$.

The preference orderings are then obtained by letting $\mathbf{x}_i = \mathbf{x}_i^* \circ \boldsymbol{\sigma}^{-1}$ for $i = 1, \dots, n$.

Example

To provide additional clarity we now consider a brief example which shows how the data generating mechanism works in practice. Suppose we have $K = 5$ entities and let $\mathcal{K} = \{a, b, c, d, e\}$ denote the collection of all entities. Further, let $\boldsymbol{\lambda} = (\lambda_a, \lambda_b, \lambda_c, \lambda_d, \lambda_e) = (10, 8, 6, 4, 2)$ and so entity a is most preferred, entity b is second most preferred and so on. Note that, for the standard (forward ranking) Plackett–Luce model, $\boldsymbol{\sigma} = (1, 2, 3, 4, 5)$, and with this choice of skill parameters, the *optimal ranking* that maximises the standard PL probability is $\hat{\mathbf{x}}_1 = (a, b, c, d, e)$. Now let $\boldsymbol{\sigma} = (5, 3, 2, 1, 4)$ so that rankers first choose their least preferred entity, then choose their 3rd, 2nd, 1st and 4th most preferred entities respectively. Recall that $\lambda_k^* = \lambda_{\sigma_k^{-1}}$ for $k = 1, \dots, K$ and so $\boldsymbol{\lambda}^* = (4, 6, 8, 2, 10)$ given $\boldsymbol{\sigma}^{-1} = (4, 3, 2, 5, 1)$. It follows that, with respect to $\boldsymbol{\lambda}^*$, entity e is the “strongest” entity and therefore the most likely to be selected first. Therefore, given $\boldsymbol{\lambda}^*$, the optimal permuted ranking is $\hat{\mathbf{x}}^* = (e, c, b, a, d)$ and so the corresponding optimal preference order is $\hat{\mathbf{x}}_2 = \hat{\mathbf{x}}^* \circ \boldsymbol{\sigma}^{-1} = (a, b, c, d, e)$, that is, the optimal ranking is the same as that for the standard forward ranking process given $\boldsymbol{\lambda}$, which seems sensible. The permuted rankings \mathbf{x}_i^* are then generated by repeatedly performing Steps 1 and 2 above and the corresponding preference orderings are given by $\mathbf{x}_i = \mathbf{x}_i^* \circ \boldsymbol{\sigma}^{-1}$.

6.2.2 Identifiability of the ranking process

In this section we discuss the identifiability of the ranking process. Indeed, it is perhaps not obvious that the data contain any information to identify the choice order parameter $\boldsymbol{\sigma}$. Again suppose that we have $K = 5$ entities with $\mathcal{K} = \{a, b, c, d, e\}$ and skill parameter

vector $\boldsymbol{\lambda} = (\lambda_a, \lambda_b, \lambda_c, \lambda_d, \lambda_e)$. If we now consider a single preference ordering $\mathbf{x}_1 = (1, 2, 3, 4, 5) \equiv (a, b, c, d, e)$ and two different choice orderings $\boldsymbol{\sigma}_1 = (1, \dots, K)$ and $\boldsymbol{\sigma}_2 = (K, \dots, 1)$ then the probability of the preference ordering for each choice order is

$$\begin{aligned}\Pr(\mathbf{x}_1 | \boldsymbol{\lambda}, \boldsymbol{\sigma}_1) &= \frac{\lambda_a}{\lambda_a + \lambda_b + \lambda_c + \lambda_d + \lambda_e} \times \frac{\lambda_b}{\lambda_b + \lambda_c + \lambda_d + \lambda_e} \times \frac{\lambda_c}{\lambda_c + \lambda_d + \lambda_e} \times \frac{\lambda_d}{\lambda_d + \lambda_e} \times \frac{\lambda_e}{\lambda_e}. \\ \Pr(\mathbf{x}_1 | \boldsymbol{\lambda}, \boldsymbol{\sigma}_2) &= \frac{\lambda_a}{\lambda_a + \lambda_b + \lambda_c + \lambda_d + \lambda_e} \times \frac{\lambda_b}{\lambda_b + \lambda_c + \lambda_d + \lambda_e} \times \frac{\lambda_c}{\lambda_c + \lambda_d + \lambda_e} \times \frac{\lambda_d}{\lambda_d + \lambda_e} \times \frac{\lambda_e}{\lambda_e}.\end{aligned}$$

Therefore, in this scenario, there is clearly no information in the single preference ordering \mathbf{x}_1 with which to identify the choice order as the probability of \mathbf{x}_1 is the same for all $\boldsymbol{\lambda} \in \mathbb{R}_{>0}^K$ under both choice orders. However, let us instead consider the likelihood of several preference orders. Given that the preference orders are conditionally independent, the likelihood under the Extended Plackett–Luce model is

$$\begin{aligned}\pi(\mathcal{D} | \boldsymbol{\lambda}, \boldsymbol{\sigma}) &= \prod_{i=1}^n \Pr(\mathbf{x}_i | \boldsymbol{\lambda}, \boldsymbol{\sigma}) \\ &= \prod_{i=1}^n \prod_{j=1}^K \frac{\lambda_{\sigma_{x_i \sigma_j}^{-1}}}{\sum_{m=j}^K \lambda_{\sigma_{x_i \sigma_m}^{-1}}}\end{aligned}\tag{6.4}$$

where \mathcal{D} denotes a collection of preference orders $\{\mathbf{x}_i\}_{i=1}^n$. Now suppose that $n = 2$ and the additional preference order is $\mathbf{x}_2 = (3, 2, 1, 4, 5) \equiv (c, b, a, d, e)$ then from (6.4) it follows that the probability of $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2\}$ under the choice order $\boldsymbol{\sigma}_1$ is

$$\begin{aligned}\pi(\mathcal{D} | \boldsymbol{\lambda}, \boldsymbol{\sigma}_1) &= \frac{\lambda_a}{\lambda_a + \lambda_b + \lambda_c + \lambda_d + \lambda_e} \times \frac{\lambda_b}{\lambda_b + \lambda_c + \lambda_d + \lambda_e} \times \frac{\lambda_c}{\lambda_c + \lambda_d + \lambda_e} \times \frac{\lambda_d}{\lambda_d + \lambda_e} \times \frac{\lambda_e}{\lambda_e} \\ &\quad \times \frac{\lambda_c}{\lambda_c + \lambda_b + \lambda_a + \lambda_d + \lambda_e} \times \frac{\lambda_b}{\lambda_b + \lambda_a + \lambda_d + \lambda_e} \times \frac{\lambda_a}{\lambda_a + \lambda_d + \lambda_e} \times \frac{\lambda_d}{\lambda_d + \lambda_e} \times \frac{\lambda_e}{\lambda_e},\end{aligned}$$

and under the choice order $\boldsymbol{\sigma}_2$ is

$$\begin{aligned}\pi(\mathcal{D} | \boldsymbol{\lambda}, \boldsymbol{\sigma}_2) &= \frac{\lambda_a}{\lambda_a + \lambda_b + \lambda_c + \lambda_d + \lambda_e} \times \frac{\lambda_b}{\lambda_b + \lambda_c + \lambda_d + \lambda_e} \times \frac{\lambda_c}{\lambda_c + \lambda_d + \lambda_e} \times \frac{\lambda_d}{\lambda_d + \lambda_e} \times \frac{\lambda_e}{\lambda_e} \\ &\quad \times \frac{\lambda_a}{\lambda_a + \lambda_b + \lambda_e + \lambda_d + \lambda_c} \times \frac{\lambda_b}{\lambda_b + \lambda_e + \lambda_d + \lambda_c} \times \frac{\lambda_e}{\lambda_e + \lambda_d + \lambda_c} \times \frac{\lambda_d}{\lambda_d + \lambda_c} \times \frac{\lambda_c}{\lambda_c}.\end{aligned}$$

It is clear that the two probabilities are only equal when $\lambda_a = \lambda_c = \lambda_e$ and so

$$\pi(\mathcal{D} | \boldsymbol{\lambda}, \boldsymbol{\sigma}_1) = \pi(\mathcal{D} | \boldsymbol{\lambda}, \boldsymbol{\sigma}_2) \iff \lambda_a = \lambda_c = \lambda_e.$$

As more (unique) preference orderings are introduced into the likelihood it is clear that additional constraints on the skill parameters will be required in order for the likelihood

Rank	Entity				
	a	b	c	d	e
1	0.33	0.27	0.20	0.13	0.07
2	0.28	0.26	0.22	0.16	0.08
3	0.21	0.23	0.24	0.20	0.12
4	0.13	0.17	0.22	0.28	0.20
5	0.04	0.07	0.12	0.23	0.54

Table 6.1: Probabilities that each entity is assigned to a specific rank for the standard Plackett–Luce model with $\boldsymbol{\lambda} = (5, 4, 3, 2, 1)$.

to be the same under different choice orders. Indeed for sufficiently large n we have that

$$\pi(\mathcal{D}|\boldsymbol{\lambda}, \boldsymbol{\sigma}_i) = \pi(\mathcal{D}|\boldsymbol{\lambda}, \boldsymbol{\sigma}_j) \iff \lambda_k = \lambda$$

for $i \neq j$ and $k = 1, \dots, K$. Mollica and Tardella (2014) provide a proof to this effect which shows that each choice order $\boldsymbol{\sigma}_i \in \mathcal{S}_K$ defines a *unique* distribution over rankings. It follows that the choice order is identifiable given a reasonably large number of rankings n . Unfortunately providing theoretical justification for a sufficient value of n is difficult as this will not only depend on the number of entities K but also on the positions of entities within the preference orderings. We suppose however that in the scenario where $n \gg K$ the choice order is likely to be identifiable.

We now take this opportunity to provide some insight into where the information about the choice order is contained within the data (preference orderings). It turns out that it is the variation within the positions of the preference orderings that allows us to determine the choice order. To see this it is useful to first consider the probabilities of each entity being assigned to a specific rank for the standard (forward ranking) Plackett–Luce model. Suppose we have $K = 5$ entities with skill parameters $\boldsymbol{\lambda} = (5, 4, 3, 2, 1)$. Again we denote the complete set of entities as $\mathcal{K} = \{a, b, c, d, e\}$ and so entities a and e are the most and least preferred entities respectively. We can describe the variation within the preference orderings by calculating the probabilities that each entity is assigned to a specific rank; see Table 6.1. Note that Table 6.1 is not symmetric and so the probability of the most preferred entity being ranked last is not the same as the probability of the least preferred entity being ranked first. Interestingly we also see that the least preferred entity (e) is ranked last the majority of the time (probability 0.54). In contrast, the most preferred entity (a) is only ranked first a third of the time. This suggests that there is more uncertainty about which entities are assigned to the ranks at the beginning of the preference ordering in comparison to the latter ranks with this choice of $\boldsymbol{\lambda}$. In other words, the weaker entities are often clearly identified and appear near the bottom of the preference orderings whereas

Rank	Entity					Rank	Entity				
	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>e</i>		<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>e</i>
≤ 1	0.33	0.27	0.20	0.13	0.07	≥ 1	1.00	1.00	1.00	1.00	1.00
≤ 2	0.61	0.53	0.42	0.29	0.15	≥ 2	0.66	0.73	0.80	0.87	0.94
≤ 3	0.82	0.76	0.46	0.49	0.27	≥ 3	0.38	0.47	0.58	0.71	0.86
≤ 4	0.95	0.93	0.88	0.77	0.47	≥ 4	0.17	0.24	0.34	0.51	0.74
≤ 5	1.00	1.00	1.00	1.00	1.00	≥ 5	0.04	0.07	0.12	0.23	0.54

Table 6.2: Cumulative probabilities of each entity being ranked no lower than (left) and no higher than (right) or equal to each position. Entry i, j corresponds to $\Pr(j \in x_{1:i})$ (left) and $\Pr(j \in x_{i:K})$ (right).

the stronger entities are more susceptible to appearing lower than they perhaps should in the preference orderings. What we are actually seeing here is an artifact of the forward ranking process. Recall that the forward ranking process dictates that a ranker first assigns an entity to rank 1 and so they choose their most preferred entity from the full set \mathcal{K} , that is, from the K entities that are available for selection when they make this choice. The ranker then chooses an entity for rank 2 where this choice is made conditional on the entity they placed in rank 1 no longer being available for selection and so, at this stage of the ranking process, there are only $K - 1$ possible entities from which to choose. It is clear that at each step in the ranking process there is one fewer entity to choose from than in the previous step. It follows that when a ranker is allocating their least preferred entities, most of the other entities will have already been allocated and so there are only a few to choose from and this results in the reduced variation on which entities are assigned to these ranks. The level of variation within each rank is further highlighted in Table 6.2 which shows the cumulative probabilities of each entity being ranked no lower than (left) and no higher than (right) each rank.

We now look at the probabilities that entities are assigned a specific rank for the Extended Plackett–Luce model. Suppose the choice order is $\sigma = (5, 3, 2, 1, 4)$ and as before let the skill parameters be $\lambda = (5, 4, 3, 2, 1)$, that is, we keep the preference of entities as before, with entity *a* being the most preferred and entity *e* the least preferred. Table 6.3 (top left) shows the probabilities that each entity is assigned to a specific rank (within the preference ordering). By comparing these probabilities to those from the standard (forward ranking) Plackett–Luce model in Table 6.1 it is clear that the distribution (of entities) over the ranks is not the same for the Extended Plackett–Luce model, that is, the EPL model defines a different ranking distribution. However, as we shall now see, these distributions are inherently related. Recall that when outlining the data generating mechanism for the EPL model we noted that the skill parameters λ are no longer proportional to the probability that an entity is ranked first within the preference ordering and instead the

		Entity							Entity				
Rank		<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>e</i>	Rank		<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>e</i>
1		0.28	0.22	0.17	0.20	0.13	σ_1		0.13	0.20	0.27	0.07	0.33
2		0.20	0.24	0.23	0.12	0.21	σ_2		0.16	0.22	0.26	0.08	0.28
3		0.16	0.22	0.26	0.08	0.28	σ_3		0.20	0.24	0.23	0.12	0.21
4		0.23	0.12	0.07	0.54	0.04	σ_4		0.28	0.22	0.17	0.20	0.13
5		0.13	0.20	0.27	0.07	0.33	σ_5		0.23	0.12	0.07	0.54	0.04

		Entity				
Rank		<i>e</i>	<i>c</i>	<i>b</i>	<i>a</i>	<i>d</i>
σ_1		0.33	0.27	0.20	0.13	0.07
σ_2		0.28	0.26	0.22	0.16	0.08
σ_3		0.21	0.23	0.24	0.20	0.12
σ_4		0.13	0.17	0.22	0.28	0.20
σ_5		0.04	0.07	0.12	0.23	0.54

Table 6.3: Probabilities of each entity being ranked within each position. Entry i, j corresponds to $\Pr(x_i = j)$, that is, the probability entity j receives rank i .

“original” parameters λ_k^* are proportional to the probability that entity k is ranked in position σ_1 . Given this it seems sensible to instead consider the probabilities that each entity is assigned rank σ_i . Table 6.3 (top right) shows these probabilities and we note these are obtained by simply applying the permutation $\boldsymbol{\sigma}$ to the rows of the table in Table 6.3 (top left). Note that these probabilities are actually the probabilities that each entity is ranked first within the permuted rankings \boldsymbol{x}^* . Now by recalling that the choice order is $\boldsymbol{\sigma} = (5, 3, 2, 1, 4)$ it follows that the entity most likely to be first in \boldsymbol{x}^* is that which is the fifth (σ_1 th) preferred entity. If we therefore permute the columns of the table so that, with respect to \boldsymbol{x}^* , the entities are the most to least preferred from left to right then the probabilities for the Extended Plackett–Luce model become the same as those for the standard (forward ranking) Plackett–Luce model; see Table 6.3 (bottom) and Table 6.1. It is therefore clear that the data contain information in which to identify the choice order $\boldsymbol{\sigma}$. In the following section we examine how informative the likelihood function is about the choice order $\boldsymbol{\sigma}$ before we then consider Bayesian analysis in Section 6.4.

6.3 Likelihood information about the choice order

Before we consider a fully Bayesian analysis of the Extended Plackett–Luce model, we examine how informative the likelihood function is about the choice order $\boldsymbol{\sigma}$. In this section we verify that the choice order is not only identifiable but the likelihood function can also be quite informative. We examine this issue by looking at the maximised likelihood

function as a function of the choice order $\boldsymbol{\sigma}$. The likelihood is maximised at the maximum likelihood estimate $\hat{\boldsymbol{\lambda}}(\boldsymbol{\sigma})$ and this estimate will depend on the choice order.

Recall from Section 2.2.2 that the standard Plackett–Luce probability is invariant to scalar multiplication of the skill parameters. Also, we saw in Section 6.2 that the Extended Plackett–Luce model is simply the standard Plackett–Luce model evaluated over the permuted rankings $\mathbf{x}^* = \mathbf{x} \circ \boldsymbol{\sigma}$ (with parameters $\boldsymbol{\lambda}^*$). Thus the EPL probability is also invariant to scalar multiplication of the parameters $\boldsymbol{\lambda}^*$ (and therefore also the skill parameters $\boldsymbol{\lambda}$), that is, $\Pr(\mathcal{D}|\boldsymbol{\lambda}, \boldsymbol{\sigma}) = \Pr(\mathcal{D}|C\boldsymbol{\lambda}, \boldsymbol{\sigma})$ for any $C \in \mathbb{R}_{>0}$ and so $\hat{\boldsymbol{\lambda}}$ cannot be identified. However, this issue is resolved by constraining the skill parameters so that $\sum_k \lambda_k = 1$, that is, placing the skill parameter vector $\boldsymbol{\lambda}$ on the $(K - 1)$ -dimensional simplex .

We now examine the information in the likelihood function by looking at the maximised log-likelihood for a given choice order $\boldsymbol{\sigma}$. Hunter (2004) proposed a Minorisation/Maximisation (MM) algorithm which enables the maximum likelihood estimate of the parameters of the standard Plackett–Luce model to be obtained. Here, by again noting that the Extended Plackett–Luce model is simply the standard Plackett–Luce model given the permuted rankings $\mathbf{x}^* = \mathbf{x} \circ \boldsymbol{\sigma}$ and with parameters $\boldsymbol{\lambda}^*$, it follows that we can obtain the MLE $\hat{\boldsymbol{\lambda}}^*$ given a fixed choice order $\boldsymbol{\sigma}$ by applying the standard MM algorithm to the permuted preference orderings. Note that, if desired, the MLE $\hat{\boldsymbol{\lambda}}$ of the skill parameters is straightforward to obtain with $\hat{\lambda}_k = \hat{\lambda}_{\sigma_k}^*$ for $k = 1, \dots, K$. It follows that, for any $\boldsymbol{\sigma} \in \mathcal{S}_k$ we can let $\mathcal{D} = \{\mathbf{x}_i^*\}_{i=1}^n$ be the collection of permuted rankings and use the MM algorithm to obtain $\hat{\boldsymbol{\lambda}}^*$ so that the standard Plackett–Luce likelihood

$$\pi_{\text{PL}}(\mathcal{D}|\boldsymbol{\lambda}^*) = \prod_{i=1}^n \prod_{j=1}^K \frac{\lambda_{x_{ij}^*}^*}{\sum_{m=j}^K \lambda_{x_{im}^*}^*} \quad (6.5)$$

is maximised. We now describe the MM algorithm which enables us to obtain the MLE $\hat{\boldsymbol{\lambda}}^*$ of the parameters given a *fixed* choice order.

6.3.1 MM Algorithm

The collection of skill parameters $\hat{\boldsymbol{\lambda}}$ which maximise the Extended Plackett–Luce likelihood for a given choice order $\boldsymbol{\sigma}$ can be obtained from the parameters $\hat{\boldsymbol{\lambda}}^*$ that maximise the standard (forward ranking) Plackett–Luce model given a collection of permuted rankings $X^* = \{\mathbf{x}_i \circ \boldsymbol{\sigma}\}_{i=1}^n$. Specifically $\hat{\lambda}_k = \hat{\lambda}_{\sigma_k}^*$ for $k = 1, \dots, K$ where $\hat{\boldsymbol{\lambda}}^*$ is obtained as follows.

1. Initialise: Let $t = 0$ and $\boldsymbol{\lambda}^{*(0)} = (\lambda_1^{*(0)}, \lambda_2^{*(0)}, \dots, \lambda_K^{*(0)})$

$$2. \text{ Let } \lambda_k^{*(t)} = \frac{w_k}{\sum_{i=1}^n \sum_{j=1}^K \delta_{ij}(k) \left(\sum_{m=j}^K \lambda_{x_{im}^*}^{*(t-1)} \right)^{-1}} \text{ for } k = 1, \dots, K.$$

3. Rescale:

- calculate $\Sigma^{(t)} = \sum_{k=1}^K \lambda_k^{*(t)}$.
- let $\lambda_k^{*(t)} \rightarrow \lambda_k^{*(t)} / \Sigma^{(t)}$ for $k = 1, \dots, K$.

4. Set $t = t + 1$ and return to Step 2.

Here the quantity $w_k = \sum_{i=1}^n \sum_{k=1}^K \mathbb{I}(x_{ij}^* = k)$ is the number of times the parameter λ_k^* represents an entity in the collection of permuted rankings and $\delta_{ij}(k) = \mathbb{I}(k \in \{x_{ij}^*, \dots, x_{in_i}^*\})$ is an indicator variable over the event that entity k appears no higher than position j in the permuted ranking i . Note that $w_k = n$ given we only consider complete rankings for the Extended Plackett–Luce model.

The number of iterations to perform is an important issue to consider when implementing the above MM algorithm. Hunter (2004) shows that in the limit as $t \rightarrow \infty$ the (true) MLE $\hat{\boldsymbol{\lambda}}^*$ is obtained almost surely. Of course, in practice we must consider only a finite number of iterations so that $\boldsymbol{\lambda}^{*(t)}$ is a reasonable approximation to the (true) MLE $\hat{\boldsymbol{\lambda}}^*$. Choosing a finite number of iterations to perform *a priori* is difficult as there is no guarantee that the algorithm will have converged by this point. To avoid making this choice it is sensible to instead implement a *stopping rule*. A stopping rule is a (mathematical) logical statement which is evaluated at each iteration and the algorithm proceeds until this logical statement is satisfied. Although there is a wealth of possible stopping rules we could consider, for the purpose of this thesis, we implement the straightforward stopping rule used by Hunter (2004) and take $\boldsymbol{\lambda}^{*(t)}$ to be the MLE when the L_2 -norm of the change in the value of the parameter vector is less than 10^{-9} , that is, when

$$\|\boldsymbol{\lambda}^{*(t)} - \boldsymbol{\lambda}^{*(t-1)}\|_2 = \sqrt{\sum_{k=1}^K (\lambda_k^{*(t)} - \lambda_k^{*(t-1)})^2} < 10^{-9} \quad (6.6)$$

is satisfied.

6.3.2 Simulation study

In this study we look at a single dataset with $n = 5000$ *complete* rankings (preference orderings) of $K = 5$ entities. We consider a large number of preference orderings so that the MLE $\hat{\boldsymbol{\lambda}}^*$ for each of the $K! = 120$ different possible choice orders is obtained after relatively

few iterations of the MM algorithm. The preference orderings were simulated from the data generating mechanism described in Section 6.2.1 with skill parameters $\boldsymbol{\lambda} = (5, 4, 3, 2, 1)$ and choice order $\boldsymbol{\sigma} = (5, 3, 2, 1, 4)$. The primary focus of this study is to verify that the choice order is identifiable and to investigate maximised values of the log-likelihood for different choice orders. Here the actual values of the MLEs $\hat{\boldsymbol{\lambda}}^*$ (under the different choice orders) is not a primary concern but nevertheless these parameters must be interpreted with respect to the choice order $\boldsymbol{\sigma}$ as we are working with the standard (forward ranking) Plackett–Luce model given permuted rankings.

Recall from Section 6.3 that we can use the MM algorithm to obtain the MLE $\hat{\boldsymbol{\lambda}}_j^*$ that maximises $\pi_{PL}(\mathcal{D} = \{\mathbf{x}_i^*\}_{i=1}^n | \boldsymbol{\lambda}^*)$ for any permutation $\boldsymbol{\sigma}_j$, and so obtain the MLEs $\hat{\boldsymbol{\lambda}}_j$ given $\boldsymbol{\sigma}_j$ for all possible permutations $j = 1, \dots, K!$. Each MM algorithm is initialised with $\lambda_k^* = \lambda^* = 1/K$ and proceeds until the stopping rule (6.6) is satisfied. Figure 6.1 shows $\log \pi(\mathcal{D} | \hat{\boldsymbol{\lambda}}_j, \boldsymbol{\sigma}_j)$, that is, the log-likelihood of the Extended Plackett–Luce model evaluated at the MLE of the skill parameters $\hat{\boldsymbol{\lambda}}_j$ for each choice order $\boldsymbol{\sigma}_j$. Clearly the log-likelihood is not constant and indeed the assumed choice order can have a large affect on the overall log-likelihood. Table 6.4 shows (a subset of) the ranking of choice orders (permutations) based on the value of the log-likelihood under the corresponding MLE for the skill parameters. The Kendall-tau distance between each choice order and the true permutation $\boldsymbol{\sigma} = (5, 3, 2, 1, 4)$ is also given. Interestingly each of the top 5 choice orders have $\sigma_4 = 1$ and $\sigma_5 = 4$ and so we have clearly identified that the most preferred entity is chosen 4th and the fourth preferred entity is chosen last. Also, perhaps surprisingly, the reverse choice order ($\boldsymbol{\sigma} = (4, 1, 2, 3, 5)$) to the one these data were simulated from results in the ninth largest log-likelihood. By definition this permutation is the furthest in (Kendall-tau) distance from the “true” choice order and so it is clear that local modes can be separated by large distances within permutation space. This is a key observation and

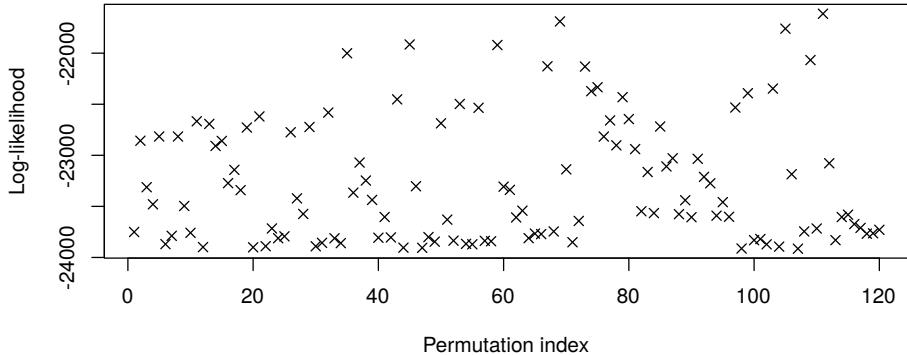


Figure 6.1: $\pi(\mathcal{D} | \hat{\boldsymbol{\lambda}}_j, \boldsymbol{\sigma}_j)$: maximised log-likelihood given each choice order $\boldsymbol{\sigma}_j$ and the respective MLE $\hat{\boldsymbol{\lambda}}_j$ for $j = 1, \dots, K!$.

Rank	Log-likelihood	Choice order	Kendall-tau distance
1	-21613.95	(5,3,2,1,4)	0
2	-21689.21	(3,5,2,1,4)	1
3	-21760.05	(5,2,3,1,4)	1
4	-21914.95	(2,5,3,1,4)	2
5	-21920.27	(3,2,5,1,4)	2
:	:	:	
9	-22131.35	(4,1,2,3,5)	10
:	:	:	
84	-23750.71	(1,2,3,4,5)	7
:	:	:	
116	-23900.67	(1,5,2,4,3)	5
117	-23904.19	(2,5,1,4,3)	4
118	-23905.65	(2,5,4,1,3)	5
119	-23912.46	(5,1,2,4,3)	4
120	-23914.28	(5,2,4,1,3)	4

Table 6.4: A subset of the ranking of choice orders (permutations) based on the value of the log-likelihood evaluated at the corresponding MLE for the skill parameters ($\pi(\mathcal{D}|\hat{\lambda}_j, \sigma_j)$).

will play an important role when we consider a Bayesian inference scheme for the EPL model in the following section.

We note in passing that the choice order $\sigma = (5, 3, 2, 1, 4)$ from which these data were simulated is that which gives the largest maximised log-likelihood, that is, it is also the MLE for σ for these data. However, in general, it is problematic to determine the MLE for σ as this requires a search over all possible ($K!$) choice orders, and this may be prohibitive when K is not small.

6.4 Inference – a Bayesian approach

In this section we consider a Bayesian approach to inference as opposed to the maximum likelihood approach taken previously. By taking a Bayesian approach we are able to obtain the posterior distribution $\pi(\sigma|\mathcal{D})$ and, as a consequence, quantify the uncertainty on the choice order parameter in a principled manner. To the best of our knowledge, the only current Bayesian solution is given by Mollica and Tardella (2018) but this relies on a restricted sample space for σ . Here we aim to develop MCMC methods capable of exploring the entire sample space. This is not straightforward given that $\sigma \in \mathcal{S}_K$ and so particular attention must be placed on how we can construct a Markov chain Monte Carlo sampling scheme capable of effectively exploring this space and targeting the correct posterior distribution.

6.4.1 Prior specification and latent variables

Before we perform Bayesian inference we must first choose a suitable prior specification for our model. We consider the same prior specification for the skill parameters as when we considered the standard Plackett–Luce model in Chapter 2, that is, we let $\lambda_k \stackrel{\text{indep}}{\sim} \text{Ga}(a_k, 1)$ *a priori* (recall that the rate parameter is not likelihood identifiable and therefore chosen to be 1). It follows that the prior distribution over $\boldsymbol{\lambda}$ is

$$\pi(\boldsymbol{\lambda}) = \prod_{k=1}^K \frac{\lambda_k^{a_k-1} e^{-\lambda_k}}{\Gamma(a_k)}.$$

Note that here we are free to choose a unique shape parameter a_k for each entity as we do not consider entity clustering (and so there are no exchangeability constraints). However, for the Extended Plackett–Luce model, additional attention must be placed on the choice of a_k as the ranking distribution is not solely specified by $\boldsymbol{\lambda}$ but also by the choice order $\boldsymbol{\sigma}$. Recall from Section 6.2.1 that the probability of entity k receiving rank σ_1 in the preference ordering is proportional to λ_k^* , that is, $\Pr(x_{\sigma_1} = k) \propto \lambda_k^*$ where $\lambda_k^* = \lambda_{\sigma_k^{-1}}$ for $k = 1, \dots, K$. It follows that specifying an informative prior for the skill parameters is tricky unless the choice order $\boldsymbol{\sigma}$ is fixed and so in practice it is often useful to instead let $a_k = a$ so that all preference orders are equally likely *a priori* (irrespective of the choice order $\boldsymbol{\sigma}$).

We must also choose a suitable prior distribution over the possible choice orderings. Each choice ordering $\boldsymbol{\sigma}_i$ is an element of \mathcal{S}_K and so there are $K!$ possible choice orderings given K entities. It follows that the prior distribution over the choice orderings is a discrete distribution with $K!$ possible values. We assume that each choice order (permutation) is equally likely *a priori* and so

$$\Pr(\boldsymbol{\sigma} = \boldsymbol{\sigma}_i) = \frac{1}{K!}$$

for $i = 1, \dots, K!$. Note that, if desired, it is possible to consider a subset of all the possible choice orderings by making an appropriate choice of prior probabilities.

Recall from Section 6.2.2 that the likelihood under the Extended Plackett–Luce model is

$$\pi(\mathcal{D}|\boldsymbol{\lambda}, \boldsymbol{\sigma}) = \prod_{i=1}^n \prod_{j=1}^K \frac{\lambda_{\sigma_{x_i \sigma_j}^{-1}}}{\sum_{m=j}^K \lambda_{\sigma_{x_i \sigma_m}^{-1}}}$$

and so is of similar form to that of the standard Plackett–Luce model. It follows from what we have seen previously that the form of the likelihood does not admit conjugate Bayesian inference. The implementation of a Gibbs sampler to maintain computational efficiency without the need for multiple tuning parameters is however highly desirable.

To facilitate this we appeal to the same technique as for previous models, that is, use data augmentation. A Gibbs sampling solution can be obtained using a straightforward generalisation of the latent variables in Caron and Doucet (2012), namely

$$z_{ij} | \mathcal{D}, \boldsymbol{\lambda}, \boldsymbol{\sigma} \stackrel{\text{indep}}{\sim} \text{Exp} \left(\sum_{m=j}^K \lambda_{\sigma_{x_i \sigma_m}^{-1}} \right)$$

for $i = 1, \dots, n$, $j = 1, \dots, K$.

Using these latent variables, the complete data likelihood is

$$\begin{aligned} \pi(\mathcal{D}, Z | \boldsymbol{\lambda}, \boldsymbol{\sigma}) &= \pi(\mathcal{D} | \boldsymbol{\lambda}, \boldsymbol{\sigma}) \pi(Z | \mathcal{D}, \boldsymbol{\lambda}, \boldsymbol{\sigma}) \\ &= \prod_{i=1}^n \prod_{j=1}^K \lambda_{\sigma_{x_i \sigma_j}^{-1}} \exp \left\{ -z_{ij} \sum_{m=j}^K \lambda_{\sigma_{x_i \sigma_m}^{-1}} \right\}. \end{aligned} \quad (6.7)$$

Further, given our choice of prior distribution, the density of all stochastic quantities is

$$\begin{aligned} \pi(\boldsymbol{\lambda}, \mathcal{D}, Z, \boldsymbol{\sigma}) &= \pi(\mathcal{D} | \boldsymbol{\lambda}, \boldsymbol{\sigma}) \pi(Z | \mathcal{D}, \boldsymbol{\lambda}, \boldsymbol{\sigma}) \pi(\boldsymbol{\lambda}) \pi(\boldsymbol{\sigma}) \\ &= \prod_{i=1}^n \prod_{j=1}^K \lambda_{\sigma_{x_i \sigma_j}^{-1}} \exp \left\{ -z_{ij} \sum_{m=j}^K \lambda_{\sigma_{x_i \sigma_m}^{-1}} \right\} \times \prod_{k=1}^K \frac{\lambda_k^{a_k-1} e^{-\lambda_k}}{\Gamma(a_k)} \times \frac{1}{K!}. \end{aligned} \quad (6.8)$$

6.4.2 Full conditional distributions for $\boldsymbol{\lambda}$, Z

From (6.8) we can obtain the full conditional distributions of the skill parameters $\boldsymbol{\lambda}$ and the latent variables Z as

- $\boldsymbol{\lambda}$: For $k = 1, \dots, K$,

$$\lambda_k | \dots \stackrel{\text{indep}}{\sim} \text{Ga} \left(a_k + \beta_k, 1 + \sum_{i=1}^n \sum_{j=1}^K \delta_{ij}(k) z_{ij} \right)$$

where

$$\beta_k = \sum_{i=1}^n \sum_{j=1}^K \mathbb{I}(\sigma_{x_i \sigma_j}^{-1} = k) \quad \text{and} \quad \delta_{ij}(k) = \sum_{m=j}^K \mathbb{I}(\sigma_{x_i \sigma_j}^{-1} = k)$$

are the number of times λ_k represents an entity within the data and an indicator variable over the event that λ_k represents an entity which appears in a position no higher than j in permuted ranking i , respectively. Note that $\beta_k = n$ (for $k = 1, \dots, K$) given we only consider complete rankings for the Extended Plackett–Luce model.

- Z : For $i = 1, \dots, n, j = 1, \dots, K$,

$$z_{ij} | \dots \stackrel{\text{indep}}{\sim} \text{Exp} \left(\sum_{m=j}^K \lambda_{\sigma_{x_{i\sigma m}}^{-1}} \right).$$

6.4.3 Full conditional distribution for $\boldsymbol{\sigma}$

The full conditional distribution for the choice order $\boldsymbol{\sigma}$ is also straightforward and is the discrete distribution with probabilities

$$\Pr(\boldsymbol{\sigma} = \boldsymbol{\sigma}_i | \dots) \propto \pi(\mathcal{D}, Z | \boldsymbol{\lambda}, \boldsymbol{\sigma} = \boldsymbol{\sigma}_i) \Pr(\boldsymbol{\sigma} = \boldsymbol{\sigma}_i)$$

for $i = 1, \dots, K!$. Clearly sampling from this full conditional will require $K!$ complete data likelihood evaluations and so a Gibbs update for $\boldsymbol{\sigma}$ is probably only plausible if K is sufficiently small; perhaps not much greater than 5. Of course, the probabilities $\Pr(\boldsymbol{\sigma} = \boldsymbol{\sigma}_i | \dots)$ and $\Pr(\boldsymbol{\sigma} = \boldsymbol{\sigma}_j | \dots)$ are conditionally independent for $i \neq j$ and so could be computed in parallel which may facilitate this approach for slightly larger values of K . The computational burden will also increase with n due to the evaluation of $\pi(\mathcal{D}, Z | \boldsymbol{\lambda}, \boldsymbol{\sigma} = \boldsymbol{\sigma}_i)$. However, the complete data likelihood could also be computed in parallel (for each choice order) as a consequence of the preference orderings being conditionally independent. Although parallel computing can be beneficial this approach probably remains infeasible for even a modest number of entities and so in the next section we consider an alternative approach capable of generating posterior realisations when Gibbs sampling is infeasible.

6.4.4 Metropolis-Hastings proposals for $\boldsymbol{\sigma}$

Suppose we are in a scenario where performing a Gibbs update for $\boldsymbol{\sigma}$ is not computationally feasible. Of course, we still wish to obtain the posterior distribution $\pi(\boldsymbol{\sigma}, \boldsymbol{\lambda}, Z | \mathcal{D})$ and so we instead consider a Metropolis-Hastings update for $\boldsymbol{\sigma}$ which will allow us to target the posterior distribution (in addition to the full conditionals for $\boldsymbol{\lambda}$ and Z from Section 6.4.2). Given $\boldsymbol{\sigma} \in \mathcal{S}_K$, it follows that we must construct a suitable proposal mechanism that allows the Markov chain to efficiently explore the (discrete) space of all $K!$ possible permutations. From our investigation into the likelihood of the Extended Plackett–Luce model given different choice orders in Section 6.3.2 it is clear that $\pi(\mathcal{D} | \boldsymbol{\lambda}, \boldsymbol{\sigma})$ is multi-modal. Further, local modes can be large distances from one another within permutation space. Our proposal mechanism must therefore be capable of making large jumps within permutation space as only proposing small moves (in terms of distance) may result in our chain becoming stuck in a local mode and not exploring the entire space. With this in mind our proposal distribution $q(\boldsymbol{\sigma}^\dagger | \boldsymbol{\sigma})$ will comprise 5 alternative proposals mechanisms which occur with

probabilities $\mathbf{p}^{\text{prop}} = (p_1^{\text{prop}}, \dots, p_5^{\text{prop}})$ which are to be specified *a priori*. More formally our proposal distribution is the 5 component mixture distribution

$$q(\boldsymbol{\sigma}^\dagger | \boldsymbol{\sigma}) = \sum_{i=1}^5 p_i^{\text{prop}} q_i(\boldsymbol{\sigma}^\dagger | \boldsymbol{\sigma}) \quad (6.9)$$

where $q_i(\boldsymbol{\sigma}^\dagger | \boldsymbol{\sigma})$ denotes the i th component. We now describe each component of the mixture distribution in further detail.

Proposal 1 – the random swap

For our first proposal mechanism we consider a “random swap”. We sample (uniformly) at random, and with replacement, two positions $\rho_1, \rho_2 \in \{1, \dots, K\}$ and let the proposed choice order $\boldsymbol{\sigma}^\dagger$ be the current choice order $\boldsymbol{\sigma}$ where the elements in positions ρ_1 and ρ_2 have been swapped. More formally we sample two positions ρ_1 and ρ_2 from the discrete distributions

$$\begin{aligned} \Pr(\rho_1 = p_1) &= \frac{1}{K}, \quad \text{for } 1 \leq p_1 \leq K \\ \Pr(\rho_2 = p_2) &= \frac{1}{K}, \quad \text{for } 1 \leq p_2 \leq K \end{aligned}$$

and let

$$\sigma_{\rho_1}^\dagger = \sigma_{\rho_2}, \quad \sigma_{\rho_2}^\dagger = \sigma_{\rho_1}, \quad \text{and} \quad \sigma_i^\dagger = \sigma_i \quad \text{for } i \in \{1, \dots, K\} \setminus \{\rho_1, \rho_2\}.$$

The fact that the two positions are sampled with replacement may seem incidental however there is good reason for this. Recall that we require our proposal mechanism to be capable of proposing large jumps around permutation space. Clearly if we only consider swapping two positions within $\boldsymbol{\sigma}$ then the number of unique proposals will be small; specifically $({}^K C_2 + 1) \ll K!$ (including the “null swap” $\rho_1 = \rho_2$). Therefore to increase the number of possible proposals it is sensible to consider swapping two positions $s > 1$ times. However, by not allowing the null swap we limit the possible proposals substantially depending on whether s is chosen to be odd or even. For example, suppose $\boldsymbol{\sigma} = (1, 2, 3, 4, 5)$ and swapping positions $\rho_1 = \rho_2$ is not allowed. In this scenario it is impossible to obtain $\boldsymbol{\sigma}^\dagger = (2, 1, 3, 4, 5)$ if $s > 1$ is even, and, in contrast, if $s > 1$ is odd then $\boldsymbol{\sigma}^\dagger = (2, 1, 3, 5, 4)$ can not be obtained. However, by allowing the null swap this issue is avoided as swapping positions $\rho_1 = \rho_2$ effectively changes s from being odd or even without altering the current state of $\boldsymbol{\sigma}$; this notion is discussed further in Section 6.4.5.

A key feature of implementing a Metropolis-Hastings proposal is the so-called proposal ratio $q(\boldsymbol{\sigma} | \boldsymbol{\sigma}^\dagger) / q(\boldsymbol{\sigma}^\dagger | \boldsymbol{\sigma})$. Of course, the proposal ratio is going to involve the ratio of the

mixture distribution (6.9) however it is useful if we first consider each proposal in turn, that is, $q_i(\boldsymbol{\sigma}|\boldsymbol{\sigma}^\dagger)/q_i(\boldsymbol{\sigma}^\dagger|\boldsymbol{\sigma})$. Consider first the case when $s = 1$. For this proposal mechanism it is clear that there are two possible ways in which the proposal $\boldsymbol{\sigma}^\dagger$ can be formed; either by swapping positions (p_1, p_2) or by swapping positions (p_2, p_1) . The proposal ratio is formed by considering the probability of obtaining $\boldsymbol{\sigma}$ given the proposed value $\boldsymbol{\sigma}^\dagger$ along with the probability of the “reverse move”, that is, the probability of obtaining $\boldsymbol{\sigma}^\dagger$ given the current state $\boldsymbol{\sigma}$. It follows that, given

$$\Pr(\rho_1 = p_1, \rho_2 = p_2) = \frac{1}{K^2} = \Pr(\rho_1 = p_2, \rho_2 = p_1),$$

the proposal ratio is

$$\begin{aligned} \frac{q_1(\boldsymbol{\sigma}|\boldsymbol{\sigma}^\dagger)}{q_1(\boldsymbol{\sigma}^\dagger|\boldsymbol{\sigma})} &= \frac{\Pr(\rho_1 = p_2, \rho_2 = p_1) + \Pr(\rho_1 = p_1, \rho_2 = p_2)}{\Pr(\rho_1 = p_1, \rho_2 = p_2) + \Pr(\rho_1 = p_2, \rho_2 = p_1)} \\ &= \frac{2\Pr(\rho_1 = p_1, \rho_2 = p_2)}{2\Pr(\rho_1 = p_1, \rho_2 = p_2)} \\ &= 1 \end{aligned}$$

and so this proposal mechanism is clearly symmetric. A straightforward generalisation of this result shows that this proposal remains symmetric when considering swapping two positions $s > 1$ times; this is discussed further in Section 6.4.5.

Proposal 2 – the Poisson swap

The second proposal mechanism is what we refer to as the “Poisson swap”. As for the random swap (Proposal 1) we form the proposed choice order by swapping two positions (ρ_1, ρ_2) of the current permutation $\boldsymbol{\sigma}$, however, the positions we swap are no longer chosen uniformly at random. Here we instead consider swapping positions ρ_1 and $\rho_2 = \rho_1 + m$ where m follows a Poisson (mixture) distribution. The idea here is that swapping positions closer to one another (small m) will lead to greater acceptance rates and so we are able to tune this proposal mechanism through the choice of the distribution for m .

Formally, we sample the position ρ_1 uniformly at random, that is, from the discrete distribution

$$\Pr(\rho_1 = p_1) = \frac{1}{K}, \quad \text{for } 1 \leq p_1 \leq K.$$

Then, in contrast to Proposal 1, we swap position ρ_1 with a position that is a certain number of positions away, namely $m = (-1)^p f$ where $p \sim \text{Bern}(0.5)$, $f \sim \text{Po}(\tau)$ and so $\rho_2 = \rho_1 + m$. It follows that the parameter τ is considered to be a tuning parameter

which allows us to alter the distribution of the distance between proposed swaps so that we obtain reasonable acceptance rates. Of course, we require that $\rho_2 \in \{1, \dots, K\}$ and so we define the choice order to be cyclic, that is, we suppose that σ_1 is next to σ_K . It follows that we can let $\rho_2 \rightarrow \rho_2 \text{ Mod } K$ where Mod is the *upper modulus* given by

$$x \text{ Mod } K \equiv \{(x - 1) \bmod K\} + 1.$$

The proposed choice order $\boldsymbol{\sigma}^\dagger$ is then formed as for Proposal 1, that is

$$\sigma_{\rho_1}^\dagger = \sigma_{\rho_2}, \quad \sigma_{\rho_2}^\dagger = \sigma_{\rho_1}, \quad \text{and} \quad \sigma_i^\dagger = \sigma_i \quad \text{for } i \in \{1, \dots, K\} \setminus \{\rho_1, \rho_2\}.$$

Before we consider the proposal ratio for this swap it is useful to note that the distribution of m is symmetric about 0, that is, $\Pr(m = c) = \Pr(m = -c)$ for all $c \in \mathbb{N}$. The proposal ratio for this swap is therefore

$$\begin{aligned} \frac{q_2(\boldsymbol{\sigma} | \boldsymbol{\sigma}^\dagger)}{q_2(\boldsymbol{\sigma}^\dagger | \boldsymbol{\sigma})} &= \frac{\Pr(\rho_1 = p_2) \Pr(\rho_2 = p_1 | \rho_1 = p_2)}{\Pr(\rho_1 = p_1) \Pr(\rho_2 = p_2 | \rho_1 = p_1)} \\ &= \frac{K \Pr(\rho_1 + m = p_1 | \rho_1 = p_2)}{K \Pr(\rho_1 + m = p_2 | \rho_1 = p_1)} \\ &= \frac{\Pr(m = p_1 - \rho_1 | \rho_1 = p_2)}{\Pr(m = p_2 - \rho_1 | \rho_1 = p_1)} \\ &= \frac{\Pr(m = p_1 - p_2)}{\Pr(m = p_2 - p_1)} \\ &= \frac{\Pr(m = p_1 - p_2)}{\Pr(m = -(p_1 - p_2))} \\ &= 1, \end{aligned}$$

and so this swap is clearly symmetric. Again a straightforward generalisation of this result shows that this proposal mechanism remains symmetric when applying $s > 1$ swaps.

Proposal 3 – random insertion

The third proposal we consider is a “random insertion” proposal (Bezáková et al., 2006). In contrast to the previous (swap based) proposals here the proposed choice order $\boldsymbol{\sigma}^\dagger$ is instead formed by taking the value in position ρ_1 and inserting it back into the permutation so that it is instead in position ρ_2 . It follows that this mechanism moves numerous positions within the permutation unlike the swap moves considered previously. To see this it may be useful to consider permuting (“shuffling”) a deck of playing cards. Under this proposal mechanism the new permuted deck is formed by removing the card that is currently in position ρ_1 and then reinserting it into the deck so that it has position ρ_2 . Clearly the

cards in the positions between ρ_1 and ρ_2 must also move to accommodate this. These cards will move either up or down a position depending on the value of the pair (ρ_1, ρ_2) .

Formally, we sample two positions (ρ_1, ρ_2) from the discrete distributions

$$\begin{aligned}\Pr(\rho_1 = p_1) &= \frac{1}{K}, \quad \text{for } 1 \leq p_1 \leq K \\ \Pr(\rho_2 = p_2 | \rho_1 = p_1) &= \frac{1}{K-1}, \quad \text{for } 1 \leq p_2 \neq p_1 \leq K\end{aligned}$$

and let

$$\boldsymbol{\sigma}^\dagger = \begin{cases} (\sigma_1, \dots, \sigma_{\rho_1-1}, \sigma_{\rho_1+1}, \dots, \sigma_{\rho_2}, \sigma_{\rho_1}, \sigma_{\rho_2+1}, \dots, \sigma_K), & \text{if } \rho_1 < \rho_2 \\ (\sigma_1, \dots, \sigma_{\rho_2-1}, \sigma_{\rho_1}, \sigma_{\rho_2}, \dots, \sigma_{\rho_1-1}, \sigma_{\rho_1+1}, \dots, \sigma_K), & \text{otherwise.} \end{cases}$$

For example, suppose the current choice order is $\boldsymbol{\sigma} = (1, 2, 3, 4, 5)$ and we have $(\rho_1, \rho_2) = (2, 4)$ then the proposed choice order is $\boldsymbol{\sigma}^\dagger = (1, 3, 4, 2, 5)$, or, if instead $(\rho_1, \rho_2) = (4, 2)$ then the proposed choice order would be $\boldsymbol{\sigma}^\dagger = (1, 4, 2, 3, 5)$.

As for the previous proposal mechanisms the proposal ratio for this move is straightforward. Suppose the proposed permutation is formed by applying the moves above given $(\rho_1 = p_1, \rho_2 = p_2)$ then it is clear that the current choice order can be recovered from the proposed $\boldsymbol{\sigma}^\dagger$ by considering $(\rho_1 = p_2, \rho_2 = p_1)$ and so

$$\begin{aligned}\frac{q_3(\boldsymbol{\sigma} | \boldsymbol{\sigma}^\dagger)}{q_3(\boldsymbol{\sigma}^\dagger | \boldsymbol{\sigma})} &= \frac{\Pr(\rho_1 = p_2) \Pr(\rho_2 = p_1 | \rho_1 = p_2)}{\Pr(\rho_1 = p_1) \Pr(\rho_2 = p_2 | \rho_1 = p_1)} \\ &= \frac{K(K-1)}{K(K-1)} \\ &= 1.\end{aligned}$$

Proposal 4 – prior proposal

The fourth proposal mechanism we consider is a “prior proposal”. Here $\boldsymbol{\sigma}^\dagger$ is simply an independent draw from the prior distribution $\pi(\boldsymbol{\sigma})$. Formally this is an independence proposal as the proposal distribution is independent of the current state of the Markov chain, that is, $q(\boldsymbol{\sigma}^\dagger | \boldsymbol{\sigma}) \stackrel{d}{=} q(\boldsymbol{\sigma}^\dagger)$. It follows that the proposal ratio is straightforward and is given by

$$\begin{aligned}\frac{q_4(\boldsymbol{\sigma} | \boldsymbol{\sigma}^\dagger)}{q_4(\boldsymbol{\sigma}^* | \boldsymbol{\sigma})} &= \frac{\pi(\boldsymbol{\sigma})}{\pi(\boldsymbol{\sigma}^\dagger)} \\ &= \frac{\Pr(\boldsymbol{\sigma} = \boldsymbol{\sigma})}{\Pr(\boldsymbol{\sigma} = \boldsymbol{\sigma}^\dagger)} \\ &= 1\end{aligned}$$

as our prior distribution is uniform over all permutations. Of course, this proposal may not be symmetric if an alternative prior choice is specified.

Proposal 5 – reverse proposal

Our final proposal mechanism is the “reverse proposal”. Here we let $\boldsymbol{\sigma}^\dagger$ be the reverse ordering of the current permutation $\boldsymbol{\sigma}$, that is, if $\boldsymbol{\sigma} = (1, 2, 3, 4, 5)$ then the proposed choice order is $\boldsymbol{\sigma}^\dagger = (5, 4, 3, 2, 1)$. Note that, in general, the reverse permutation is *not* the inverse permutation. This proposal mechanism is motivated by our observation from the investigation into the likelihood of the EPL model (under all possible choice orders) in Section 6.3.2 where we saw that the reverse of the “true” choice order featured quite highly in the ranking of choice orders; see Table 6.4. Further, the reverse of the current choice order is unlikely to be proposed by any of the previous proposal mechanisms as, by construction, the reverse permutation is a large distance away from the current state. Formally we let

$$\boldsymbol{\sigma}^\dagger = \boldsymbol{\sigma}_{K:1} = (\sigma_K, \dots, \sigma_1).$$

with probability 1. It follows that the proposal distribution is $q_5(\boldsymbol{\sigma}^\dagger | \boldsymbol{\sigma}) = \delta_{\boldsymbol{\sigma}_{K:1}}$ where δ_x denotes the Dirac probability measure concentrated at x and so the proposal ratio is

$$\frac{q_5(\boldsymbol{\sigma} | \boldsymbol{\sigma}^\dagger)}{q_5(\boldsymbol{\sigma}^\dagger | \boldsymbol{\sigma})} = \frac{\delta_{\boldsymbol{\sigma}_{K:1}}}{\delta_{\boldsymbol{\sigma}_{K:1}}} = 1. \quad (6.10)$$

The acceptance probability

In general, the acceptance probability of $\boldsymbol{\sigma}^\dagger$ is $\min(1, A)$ where

$$A = \frac{\pi(\boldsymbol{\sigma}^\dagger | \dots)}{\pi(\boldsymbol{\sigma} | \dots)} \times \frac{q(\boldsymbol{\sigma} | \boldsymbol{\sigma}^\dagger)}{q(\boldsymbol{\sigma}^\dagger | \boldsymbol{\sigma})}.$$

From the density of all stochastic quantities (6.8) it is clear that

$$\begin{aligned} \pi(\boldsymbol{\sigma} | \dots) &\propto \pi(\mathcal{D} | \boldsymbol{\lambda}, \boldsymbol{\sigma}) \pi(Z | \mathcal{D}, \boldsymbol{\lambda}, \boldsymbol{\sigma}) \pi(\boldsymbol{\sigma}) \\ &= \pi(\mathcal{D}, Z | \boldsymbol{\lambda}, \boldsymbol{\sigma}) \pi(\boldsymbol{\sigma}), \end{aligned}$$

and so, unsurprisingly, the posterior distribution of $\boldsymbol{\sigma}$ is proportional to the complete data likelihood times the prior.

Recall our proposal distribution is the 5 component mixture distribution (6.9) and by noting that $q_i(\boldsymbol{\sigma}^\dagger|\boldsymbol{\sigma}) = q_i(\boldsymbol{\sigma}|\boldsymbol{\sigma}^\dagger)$ for $i = 1, \dots, 5$ it follows that the proposal ratio is

$$\frac{q(\boldsymbol{\sigma}|\boldsymbol{\sigma}^\dagger)}{q(\boldsymbol{\sigma}^\dagger|\boldsymbol{\sigma})} = \frac{\sum_{i=1}^5 p_i^{\text{prop}} q_i(\boldsymbol{\sigma}|\boldsymbol{\sigma}^\dagger)}{\sum_{i=1}^5 p_i^{\text{prop}} q_i(\boldsymbol{\sigma}^\dagger|\boldsymbol{\sigma})} = \frac{\sum_{i=1}^5 p_i^{\text{prop}} q_i(\boldsymbol{\sigma}^\dagger|\boldsymbol{\sigma})}{\sum_{i=1}^5 p_i^{\text{prop}} q_i(\boldsymbol{\sigma}^\dagger|\boldsymbol{\sigma})} = 1.$$

Clearly the acceptance probability simplifies substantially and can be written as $\min(1, A)$ where

$$A = \frac{\pi(\mathcal{D}, Z|\boldsymbol{\lambda}, \boldsymbol{\sigma}^\dagger) \Pr(\boldsymbol{\sigma} = \boldsymbol{\sigma}^\dagger)}{\pi(\mathcal{D}, Z|\boldsymbol{\lambda}, \boldsymbol{\sigma}) \Pr(\boldsymbol{\sigma} = \boldsymbol{\sigma})} = \frac{\pi(\mathcal{D}, Z|\boldsymbol{\lambda}, \boldsymbol{\sigma}^\dagger)}{\pi(\mathcal{D}, Z|\boldsymbol{\lambda}, \boldsymbol{\sigma})}$$

which is simply the ratio of the complete data likelihood given the proposed and the current permutation (assuming each choice order is chosen to be equally likely *a priori*).

6.4.5 Further considerations

Exploring large discrete spaces such as the set of all permutations \mathcal{S}_K with a Metropolis-Hastings algorithm is a non-trivial task. The proposal distribution discussed in Section 6.4.4 comprises a mixture of “local” (Proposals 1–3) and “global” (Proposals 4 and 5) proposal mechanisms for the choice order parameter in an attempt to facilitate effective exploration of this large discrete space. Although reasonably straightforward to implement, it is useful to first consider the proposal distribution in more detail as a naive implementation may result in an inefficient proposal distribution due to the subtleties involved within some of the proposal mechanisms; particularly the swap moves.

The first thing to note is that, by construction, both Proposal 4 and 5 have the potential to propose large jumps (in terms of distance) from the current permutation. Although useful for escaping local modes large jumps typically result in smaller acceptance probabilities. Given this it is perhaps sensible to only consider constructing $\boldsymbol{\sigma}^\dagger$ from either Proposal 4 or 5 relatively infrequently in comparison to the more local proposals. This can be achieved through an appropriate choice of the mixture component weights in the proposal distribution (6.9). At first glance we might also think that Proposal 3 could suffer from poor acceptance rates given the proposed permutation is formed by moving numerous positions within the current permutation. However the construction of this proposal mechanism dictates that much of the structure that is present in the current permutation is also present within the proposed permutation. It follows that, in terms of distance, the proposed permutation will not be that far away from the current permutation as much of the order is preserved.

Perhaps surprisingly it is Proposals 1 and 2 that have potential to cause issues within this proposal distribution. Note that by swapping only two positions within σ the number of unique proposals that can be generated is $(^K C_2 + 1) \ll K!$ (including the “null swap” $\rho_1 = \rho_2$) and so we could easily become stuck in a local mode. Therefore, as alluded to in Section 6.4.4, it is perhaps sensible to instead consider swapping two positions $s > 1$ times to allow our proposal mechanism to propose larger jumps around permutation space. Each of the s swaps should be performed iteratively. For example, suppose we have s pairs of positions to swap $(\rho_1, \rho_2)_1, \dots, (\rho_1, \rho_2)_s$ which are samples from the appropriate discrete distributions. The proposed choice order σ^\dagger is obtained by first forming a “*temporary proposal*” σ_1^\dagger by swapping positions $(\rho_1, \rho_2)_1$ within the current state σ , then by considering σ_1^\dagger to be the current state of the chain we obtain another temporary proposal σ_2^\dagger by swapping positions $(\rho_1, \rho_2)_2$ within σ_1^\dagger . This process continues and the proposed choice order is $\sigma^\dagger = \sigma_s^\dagger$. Naturally as s increases so does the number of possible proposals, N_p . The distance between the current and proposed choice orders will also typically increase with s and so the number of swaps is nominally a tuning parameter and can be adjusted to increase acceptance rates. Of course, we must consider how performing $s > 1$ swaps affects the proposal ratio. We consider this for Proposal 1 and note that similar arguments apply for Proposal 2. Recall that each of the s swaps is performed iteratively and so we can write the proposal distribution as

$$\begin{aligned} q_1(\sigma^\dagger | \sigma) &= q_1(\sigma_s^\dagger | \sigma_{s-1}^\dagger, \dots, \sigma_1^\dagger) \times \dots \times q_1(\sigma_1^\dagger | \sigma) \\ &= q_1(\sigma_s^\dagger | \sigma_{s-1}^\dagger) \times \dots \times q_1(\sigma_1^\dagger | \sigma) \end{aligned}$$

given the proposal distribution for each swap s is conditionally independent given the temporary proposal σ_{s-1}^\dagger . Further, it is also clear that

$$q_1(\sigma | \sigma^\dagger) = q_1(\sigma_{s-1}^\dagger | \sigma_s^\dagger) \times \dots \times q_1(\sigma | \sigma_1^\dagger)$$

and so the proposal ratio $q_1(\sigma | \sigma^\dagger) / q_1(\sigma^\dagger | \sigma) = 1$ for all $s \geq 1$ which follows from the result that $q_1(\sigma_i | \sigma_j) = q_1(\sigma_j | \sigma_i)$ for any $\sigma_i, \sigma_j \in \mathcal{S}_K$; see Section 6.4.4.

We now briefly return our attention to why it is sensible to allow the null swap when generating proposed permutations using a swapping mechanism (Proposals 1 and 2) as mentioned in Section 6.4.4. Naturally we expect the number of possible proposals (N_p) to increase as we consider more swaps, more specifically we might imagine that $N_p \rightarrow K!$ as $s \rightarrow \infty$. However, if we suppose that swapping positions $\rho_1 = \rho_2$ is *not* allowed then $N_p = K!/2$ for large s . Moreover the set of possible permutations that can be proposed depends on whether s is chosen to be even or odd. If we let Σ^{odd} and Σ^{even} denote the sets of possible permutations that can be generated when s is odd and even respectively then it can be shown that $\mathcal{S}_K = \{\Sigma^{\text{odd}} \cup \Sigma^{\text{even}}\}$ and $\Sigma^{\text{odd}} \cap \Sigma^{\text{even}} = \emptyset$. It follows that if $s > 1$ is

fixed *a priori* then repeatedly swapping two positions $\rho_1 \neq \rho_2$ only enables us to explore half of the possible permutations – those with either even or odd *parity* (depending on the parity of the current permutation). Although here we have considered the asymptotic result it is clear from our example in Section 6.4.4 that not allowing the null swap results in the proposal mechanism being unable to propose some permutations which are close in distance depending on the choice of s . One strategy to avoid this problem is to consider s to be a random variable as opposed to a fixed constant. Choosing $s \sim \text{Geom}(p_s)$ is perhaps sensible however we opt to instead consider fixed s and allow the null swap.

6.4.6 A Metropolis-within-Gibbs algorithm for the EPL model

A Metropolis-within-Gibbs algorithm can be constructed to target the joint posterior distribution of the skill parameters $\boldsymbol{\lambda}$, the choice order parameter $\boldsymbol{\sigma}$, and the latent variables Z . Posterior samples are generated as follows.

1. Initialise: choose $\boldsymbol{\sigma} \in \mathcal{S}_K$, $\boldsymbol{\lambda} \in \mathbb{R}_{>0}^K$ and $z_{ij} > 0$ for $i = 1, \dots, n$, $j = 1, \dots, K$.
2. Repeatedly perform the following steps:
 - Sample $\lambda_k | \dots \stackrel{\text{indep}}{\sim} \text{Ga}\left(a_k + n, 1 + \sum_{i=1}^n \sum_{j=1}^K \delta_{ij}(k) z_{ij}\right)$ for $k = 1, \dots, K$, where $\delta_{ij}(k)$ is as in Section 6.4.2.
 - Sample $z_{ij} | \dots \stackrel{\text{indep}}{\sim} \text{Exp}\left(\sum_{m=j}^K \lambda_{\sigma_{x_i \sigma_m}^{-1}}\right)$ for $i = 1, \dots, n$, $j = 1, \dots, K$.
 - Sample ℓ from the discrete distribution with probabilities $\Pr(\ell = i) = p_i^{\text{prop}}$ for $i = 1, \dots, 5$
 - propose $\boldsymbol{\sigma}^\dagger$ using proposal mechanism ℓ
 - let $\boldsymbol{\sigma} \rightarrow \boldsymbol{\sigma}^\dagger$ with probability $\min(1, A)$ where $A = \frac{\pi(\mathcal{D}, Z | \boldsymbol{\lambda}, \boldsymbol{\sigma}^\dagger) \Pr(\boldsymbol{\sigma} = \boldsymbol{\sigma}^\dagger)}{\pi(\mathcal{D}, Z | \boldsymbol{\lambda}, \boldsymbol{\sigma}) \Pr(\boldsymbol{\sigma} = \boldsymbol{\sigma})}$
 - Rescale
 - sample $\Lambda^\ddagger \sim \text{Ga}\left(\sum_{k=1}^K a_k, 1\right)$.
 - calculate $\Sigma = \sum_{k=1}^K \lambda_k$.
 - let $\lambda_k \rightarrow \lambda_k / \Sigma$ for $k = 1, \dots, K$.

If K is sufficiently small, or we have the computational power to do so, we could instead sample $\boldsymbol{\sigma}$ from its full conditional distribution as discussed in Section 6.4.3 in which case we would have a straightforward Gibbs sampler.

6.4.7 Simulation study – Metropolis-within-Gibbs

In this study we consider a new dataset (Dataset 6) with $n = 100$ *complete* rankings of $K = 10$ entities. The skill parameters used to simulate these data are $\boldsymbol{\lambda} = (10, 9, \dots, 1)$ and the choice order (*ranking process*) is chosen to be $\boldsymbol{\sigma} = (3, 10, 9, 1, 7, 4, 5, 6, 8, 2)$. The simulated data are given in Tables B.8 and B.9 within the Appendices.

We will adopt the same prior distribution for the skill parameters as used in previous analyses, that is, take $\lambda_k \stackrel{\text{indep}}{\sim} \text{Ga}(1, 1)$ and so $a_k = a = 1$. Further we assume that each choice order is equally likely *a priori* and so $\Pr(\boldsymbol{\sigma} = \boldsymbol{\sigma}_i) = 1/K!$ for all $\boldsymbol{\sigma}_i \in \mathcal{S}_K$. Given this prior distribution we use the Metropolis-within-Gibbs algorithm from Section 6.4.6 in an attempt to obtain posterior realisations from $\pi(\boldsymbol{\lambda}, \boldsymbol{\sigma}, Z | \mathcal{D})$. Of course, we must also choose suitable tuning parameters for the proposal distribution (6.9) of the choice order parameter $\boldsymbol{\sigma}$. From our discussion in Section 6.4.5 we believe it is sensible to let $\mathbf{p}^{\text{prop}} = (0.3, 0.3, 0.3, 0.05, 0.05)$ so that the “local” proposals (swaps and insertion) are favored over the global proposals (prior and reverse) given the former are perhaps more likely to be accepted. We also choose to perform $s = 2$ swaps when using proposals 1 and 2 and take $\tau = 3$ within the latter so the expected distance between positions being swapped is 3.

To investigate the convergence and mixing properties of our Metropolis-within-Gibbs sampler we consider 5 chains; each of which is initialised at a different choice order parameter $\boldsymbol{\sigma}$ as shown in Table 6.5. Naturally we hope that the MCMC scheme has been constructed such that it is capable of (efficiently) exploring the set of all permutations and (quickly) converging to its stationary distribution irrespective of the initial choice order. We report the results from a typical run of our MCMC scheme initialised as described, with a burn-in of 10K iterations and then run for a further 1M iterations and thinned by 100 to obtain 10K (“posterior”) samples. The MCMC scheme runs reasonably quickly, with C code on a single thread of an Intel Core i7-4790S CPU (3.20GHz clock speed) taking around 1 minute 35 seconds (for each chain).

Chain	Initial permutation	
1	$\boldsymbol{\sigma} = (1, 2, \dots, 10)$	Identity permutation
2	$\boldsymbol{\sigma} = (10, 9, \dots, 1)$	Reverse identity permutation
3	$\boldsymbol{\sigma} = (3, 10, 9, 1, 7, 4, 5, 6, 8, 2)$	True permutation
4	$\boldsymbol{\sigma} = (2, 8, 6, 5, 4, 7, 1, 9, 10, 3)$	Reverse true permutation
5	$\boldsymbol{\sigma} = (8, 10, 7, 4, 3, 1, 2, 6, 5, 9)$	Random permutation

Table 6.5: Permutations (choice orders) used for the initialisation of each chain

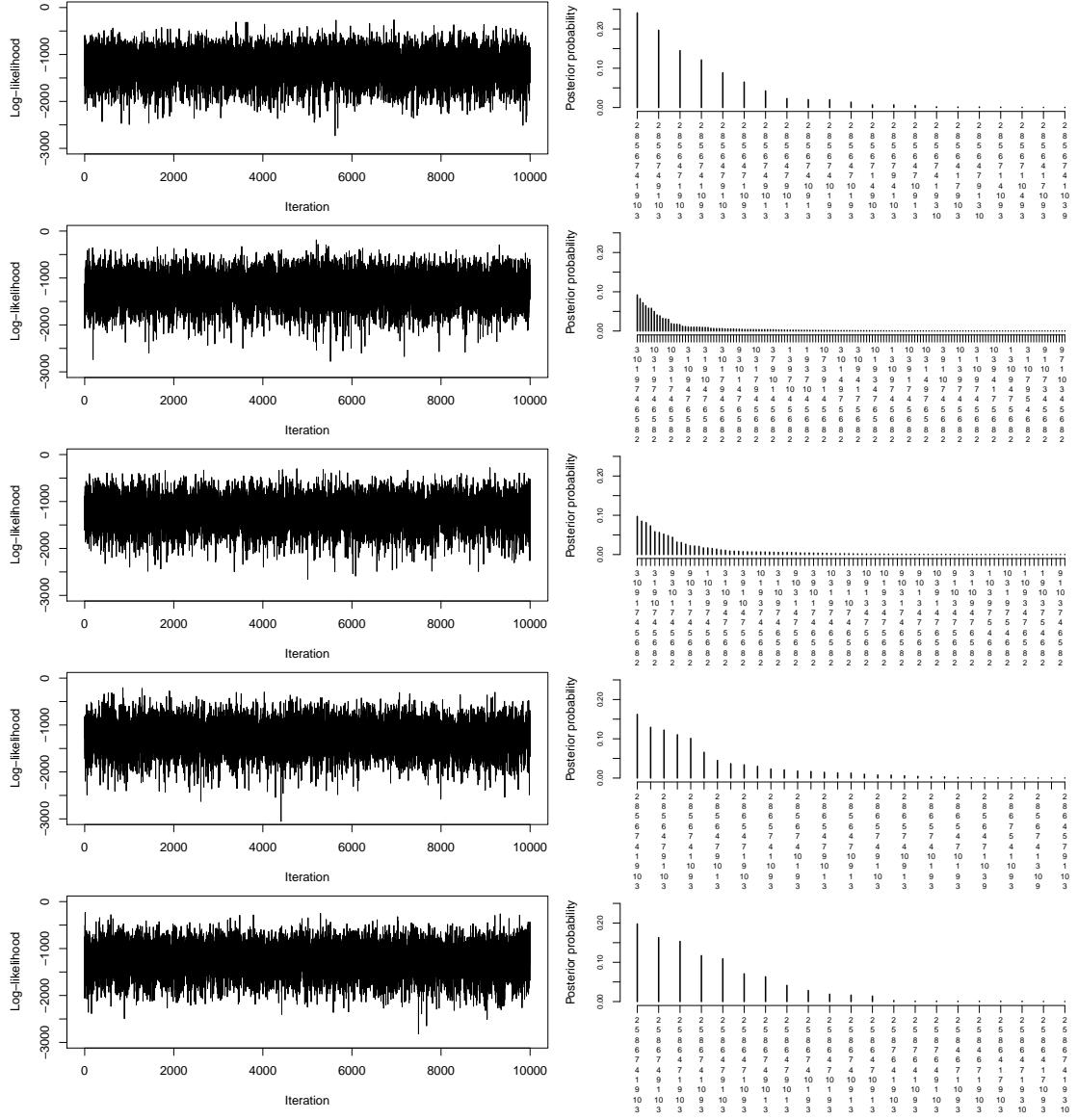


Figure 6.2: Trace plots of the log complete data likelihood (left) and the marginal posterior $\pi(\boldsymbol{\sigma}|\mathcal{D})$ (right) for chains 1 to 5 from top to bottom respectively (Metropolis-within-Gibbs approach).

Inspection of the trace plots showing the log complete data likelihood ($\log \pi(\mathcal{D}, Z|\boldsymbol{\lambda}, \boldsymbol{\sigma})$) for each chain shows that the MCMC scheme appears to be mixing fairly well and indeed shows signs of convergence; see Figure 6.2 left. However, if we look at the marginal posterior distribution of $\boldsymbol{\sigma}$ (defined by the probabilities $\Pr(\boldsymbol{\sigma} = \boldsymbol{\sigma}_i | \dots)$) then it is clear that the chains have not reached their stationary distribution; see Figure 6.2 right. Of course, given the MCMC scheme runs fairly quickly we could perform further iterations in the hope that the chains reach convergence. However, in our experience, this is often difficult to achieve. From Figure 6.2 (right) it is clear that chains 1, 4 and 5 have become stuck in the local mode around the reverse “true” choice order. Although we introduced Proposal 5 (the

	Proposal mechanism				
	1	2	3	4	5
# proposed	302995	302982	303379	50246	50399
# accepted	1867	1921	1383	0	0
Pr(accept)	0.006	0.006	0.005	0	0

Table 6.6: Acceptance rates for each of the 5 proposal mechanisms for the choice order σ .

reverse proposal) in the hope of avoiding this it appears that, in this scenario, this proposal move does not allow the chain to move to the other mode (around the “true” choice order). This observation is further supported if we consider the acceptance rates of the proposal distribution for σ . The overall acceptance rate of σ is less than 1% and Table 6.6 shows the acceptance rates for each of the 5 proposal mechanisms (components of the mixture proposal). Note that the results shown are those obtained from chain 1, however, these results are indicative of the other chains. From Table 6.6 we see that neither the reverse nor the prior proposal (Proposals 4 or 5) are accepted and so our chains struggle to escape local modes. Clearly this is an issue which needs to be resolved.

From further investigation of the posterior distribution (and also the MLEs $\hat{\lambda}_j$ from Section 6.3 which maximise the Extended Plackett–Luce probability for each choice order σ_j) it is clear that there is large correlation between λ and σ , that is, between the skill parameters and the choice order. Moreover throughout this thesis we have seen that, for a fixed choice order, Gibbs sampling via data augmentation leads to rapid convergence of the Markov chain to the stationary distribution $\pi(\lambda, Z | \mathcal{D}, \sigma)$. It follows that, given a fixed choice order σ_i , if the skill parameters λ (and the latent parameters Z) are sufficiently optimised then it is going to be difficult to propose a reasonable alternative choice order (given these λ , Z) and so our chain is going to become stuck in this mode. A typical strategy to improve mixing in such a scenario is to consider a joint update of the highly correlated parameters (Gamerman and Lopes, 2006). Unfortunately performing a joint update of (λ, Z, σ) is not straightforward – the difficulty is how to construct a suitable proposal mechanism. Of course, we could use $q(\sigma^\dagger | \sigma)$ to generate a proposed choice order as before, then, given the proposed choice order σ^\dagger , draw the skill parameters and latent variables from their full conditional distributions. By using this strategy we have the following options

- | | |
|--|--|
| 1. draw $\sigma^\dagger \sigma$ | 2. draw $\sigma^\dagger \sigma$ |
| draw $\lambda^\dagger Z, \sigma^\dagger$ | draw $Z^\dagger \lambda, \sigma^\dagger$ |
| draw $Z^\dagger \lambda^\dagger, \sigma^\dagger$ | draw $\lambda^\dagger Z^\dagger, \sigma^\dagger$ |

however, neither option is likely to generate reasonable proposals $(\boldsymbol{\lambda}^\dagger, Z^\dagger, \boldsymbol{\sigma}^\dagger)$ given that, clearly, $\boldsymbol{\lambda}$ and Z are going to be highly correlated.

Given this we must therefore consider an alternative strategy in an attempt to break the correlations and obtain an MCMC scheme that exhibits good mixing with respect to the choice order $\boldsymbol{\sigma}$. We propose to remove the latent variables Z from the parameter space and instead consider Metropolis-Hastings updates for both the skill parameters λ_k and the choice order $\boldsymbol{\sigma}$; this is the topic of the next section.

6.4.8 Metropolis-Hastings proposals for $\boldsymbol{\lambda}$

From the previous section it is clear that our Metropolis-within-Gibbs sampling scheme is inadequate to generate realisations from the posterior distribution for the Extended Plackett–Luce model. We now instead consider a Metropolis-Hastings update for the skill parameters and so need not introduce the latent variables Z into the sample space. First note that, given we are not considering the latent variables Z , the density of all stochastic quantities is now

$$\begin{aligned}\pi(\boldsymbol{\lambda}, \mathcal{D}, \boldsymbol{\sigma}) &= \pi(\mathcal{D}|\boldsymbol{\lambda}, \boldsymbol{\sigma})\pi(\boldsymbol{\lambda})\pi(\boldsymbol{\sigma}) \\ &= \prod_{i=1}^n \prod_{j=1}^K \frac{\lambda_{\sigma_{x_i \sigma_j}^{-1}}}{\sum_{m=j}^K \lambda_{\sigma_{x_i \sigma_m}^{-1}}} \times \prod_{k=1}^K \frac{\lambda_k^{a_k-1} e^{-\lambda_k}}{\Gamma(a_k)} \times \frac{1}{K!}\end{aligned}\quad (6.11)$$

and so

$$\begin{aligned}\pi(\lambda_k | \dots) &\propto \prod_{i=1}^n \prod_{j=1}^K \frac{\lambda_{\sigma_{x_i \sigma_j}^{-1}}}{\sum_{m=j}^K \lambda_{\sigma_{x_i \sigma_m}^{-1}}} \times \frac{\lambda_k^{a_k-1} e^{-\lambda_k}}{\Gamma(a_k)} \\ &= \pi(\mathcal{D}|\boldsymbol{\lambda}, \boldsymbol{\sigma})\pi(\lambda_k)\end{aligned}\quad (6.12)$$

for $k = 1, \dots, K$.

Clearly, this is a non-standard distribution and so we use a Metropolis-Hastings step to target this distribution. Given $\lambda_k > 0$ it is sensible to consider a Log-normal proposal distribution centered at the current value for the skill parameters, that is, take $\lambda_k^\dagger | \lambda_k \stackrel{\text{indep}}{\sim} \text{LN}(\log \lambda_k, \sigma_{\lambda_k}^2)$. For this choice of proposal distribution the acceptance probability is $\min(1, A)$ where

$$\begin{aligned}A &= \frac{\pi(\lambda_k^\dagger | \dots)q(\lambda_k | \lambda_k^\dagger)}{\pi(\lambda_k | \dots)q(\lambda_k^\dagger | \lambda_k)} \\ &= \frac{\pi(\mathcal{D}|\boldsymbol{\lambda}_{-k}, \lambda_k = \lambda_k^\dagger, \boldsymbol{\sigma})}{\pi(\mathcal{D}|\boldsymbol{\lambda}, \boldsymbol{\sigma})} \times \left(\frac{\lambda_k^\dagger}{\lambda_k} \right)^{a_k} e^{(\lambda_k - \lambda_k^\dagger)}.\end{aligned}$$

Further, because the density of all stochastic quantities has now changed (given we no longer consider the latent variables Z) it is clear that the acceptance probability of $\boldsymbol{\sigma}$ must also change. From the form of the density of all stochastic quantities (6.11) it is clear that $\pi(\boldsymbol{\sigma}|\dots) \propto \pi(\mathcal{D}|\boldsymbol{\lambda}, \boldsymbol{\sigma})\pi(\boldsymbol{\sigma})$ and so, given the proposal mechanism described in Section 6.4.4, the acceptance probability for $\boldsymbol{\sigma}^\dagger$ is $\min(1, A)$ where

$$A = \frac{\pi(\mathcal{D}|\boldsymbol{\lambda}, \boldsymbol{\sigma}^\dagger) \Pr(\boldsymbol{\sigma} = \boldsymbol{\sigma}^\dagger)}{\pi(\mathcal{D}|\boldsymbol{\lambda}, \boldsymbol{\sigma}) \Pr(\boldsymbol{\sigma} = \boldsymbol{\sigma})}.$$

which is simply the ratio of the EPL likelihood given the proposed and the current permutation (assuming each choice order is chosen to be equally likely *a priori*).

6.4.9 A Metropolis-Hastings algorithm for the EPL model

Using the results from Section 6.4.8 we can construct a Metropolis-Hastings algorithm to target the joint posterior distribution of the skill parameters $\boldsymbol{\lambda}$ and the choice order parameter $\boldsymbol{\sigma}$ as follows.

1. Initialise: choose $\boldsymbol{\sigma} \in \mathcal{S}_K$ and $\boldsymbol{\lambda} \in \mathbb{R}_{>0}^K$

2. Repeatedly perform the following steps:

- For $k = 1, \dots, K$
 - draw $\lambda_k^\dagger | \lambda_k \sim \text{LN}(\log \lambda_k, \sigma_{\lambda_k}^2)$
 - let $\lambda_k \rightarrow \lambda_k^\dagger$ with probability $\min(1, A)$ where
- $$A = \frac{\pi(\mathcal{D}|\boldsymbol{\lambda}_{-k}, \lambda_k = \lambda_k^\dagger, \boldsymbol{\sigma})}{\pi(\mathcal{D}|\boldsymbol{\lambda}, \boldsymbol{\sigma})} \times \left(\frac{\lambda_k^\dagger}{\lambda_k} \right)^{a_k} e^{(\lambda_k - \lambda_k^\dagger)}$$
- Sample ℓ from the discrete distribution with probabilities $\Pr(\ell = i) = p_i^{\text{prop}}$ for $i = 1, \dots, 5$
 - propose $\boldsymbol{\sigma}^\dagger$ using proposal mechanism ℓ
 - let $\boldsymbol{\sigma} \rightarrow \boldsymbol{\sigma}^\dagger$ with probability $\min(1, A)$ where $A = \frac{\pi(\mathcal{D}|\boldsymbol{\lambda}, \boldsymbol{\sigma}^\dagger) \Pr(\boldsymbol{\sigma} = \boldsymbol{\sigma}^\dagger)}{\pi(\mathcal{D}|\boldsymbol{\lambda}, \boldsymbol{\sigma}) \Pr(\boldsymbol{\sigma} = \boldsymbol{\sigma})}$.
 - Rescale
 - sample $\Lambda^\ddagger \sim \text{Ga}\left(\sum_{k=1}^K a_k, 1\right)$.
 - calculate $\Sigma = \sum_{k=1}^K \lambda_k$.
 - let $\lambda_k \rightarrow \lambda_k \Lambda^\ddagger / \Sigma$ for $k = 1, \dots, K$.

6.4.10 Simulation study – Metropolis-Hastings

In this study we revisit Dataset 6 which was first introduced in Section 6.4.7. However, here we perform Bayesian inference using the Metropolis-Hastings algorithm from Section 6.4.9 as opposed to the Metropolis-within-Gibbs sampler considered previously. Recall that Dataset 6 comprises $n = 100$ *complete* rankings of $K = 10$ entities. The skill parameters and choice order (*ranking process*) used to simulate these data are $\boldsymbol{\lambda} = (10, 9, \dots, 1)$ and $\boldsymbol{\sigma} = (3, 10, 9, 1, 7, 4, 5, 6, 8, 2)$ respectively.

We choose the same prior distribution as in Section 6.4.7 and so $a_k = a = 1$ and each choice order is equally likely *a priori*, that is, $\Pr(\boldsymbol{\sigma} = \boldsymbol{\sigma}_i) = 1/K!$ for all $\boldsymbol{\sigma}_i \in \mathcal{S}_K$. Further, we take the same tuning parameters for the proposal distribution of the choice order, that is, we choose to perform $s = 2$ swaps when using proposals 1 and 2 and take $\tau = 3$ in the latter. Again the probabilities of each proposal mechanism being used are $\mathbf{p}^{\text{prop}} = (0.3, 0.3, 0.3, 0.05, 0.05)$. As for the previous analysis we consider 5 chains, each of which is initialised at a different choice order parameter $\boldsymbol{\sigma}$ as shown in Table 6.5. Note that for this analysis we must also both initialise and tune the proposal mechanism for $\boldsymbol{\lambda}$. Here we initialise by taking independent draws from the prior ($\lambda_k \stackrel{\text{iid}}{\sim} \text{Ga}(1, 1)$) and let the tuning parameter be $\sigma_{\lambda_k} = \sigma_{\lambda} = 0.75$ which we determined to be sensible after performing numerous pilot runs. The results reported are from a typical run of our MCMC scheme initialised as described, with a burn-in of 20K iterations and then run for a further 2M iterations and thinned by 200 to obtain 10K (“posterior”) samples. The MCMC scheme runs reasonably quickly, with C code on a single thread of an Intel Core i7-4790S CPU (3.20GHz clock speed) taking around 24 minutes 30 seconds (for each chain). Note that this algorithm requires more computational time than the corresponding Metropolis-within-Gibbs algorithm from Section 6.4.6 due to the additional likelihood evaluations required to compute the MH acceptance rates for the skill parameters λ_k .

Figure 6.3 (left) shows the trace plots of the log-likelihood ($\log \pi(\mathcal{D}|\boldsymbol{\lambda}, \boldsymbol{\sigma})$) for each chain from which we see that some of the chains exhibit signs of poor mixing. In spite of the potential mixing issues it will be interesting to see what the posterior distribution looks like under each chain. Recall that when we considered a Metropolis-within-Gibbs sampler it became clear from the marginal posterior over the choice order $\boldsymbol{\sigma}$ that the chains had not reached their stationary distribution. However, in contrast to our previous analysis, our MCMC scheme appears to be mixing fairly well over $\boldsymbol{\sigma}$ with none of the chains becoming stuck in the local mode around the reverse “true” permutation; see Figure 6.3 (right). Indeed, although only a small number of choice orders are labelled on the plots, there is a significant increase in the number of choice orders which have posterior support under this analysis; see Figures 6.2 and 6.3 (right).

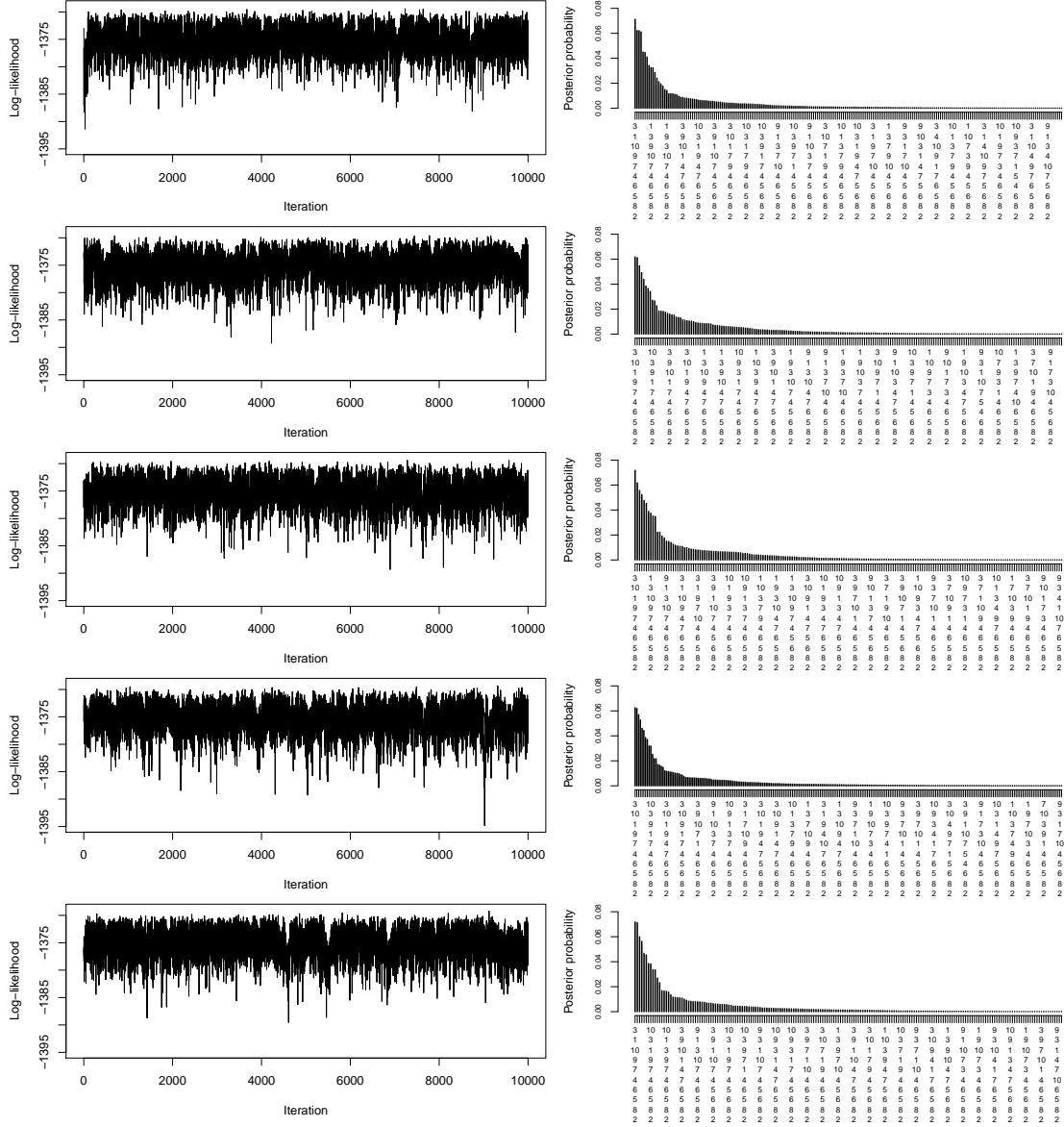


Figure 6.3: Trace plots of the log complete data likelihood (left) and the marginal posterior $\pi(\sigma|\mathcal{D})$ (right) for chains 1 to 5 from top to bottom respectively (Metropolis-Hastings approach).

To further investigate the marginal posterior distribution of σ is it useful to look at which choice orders receive high posterior support. Figure 6.4 shows the 25 choice orders with highest posterior support (and their corresponding posterior probabilities) from chains 1 to 5, that is, a subset of the marginal posterior. Although there are discrepancies between the posteriors produced from each chain it is clear that the Metropolis-Hastings approach is performing substantially better than the Metropolis-within-Gibbs algorithm. It is clear however that mixing over the choice orders must be improved further as, although reasonable here, this issue will only worsen as K increases. To help us better explore \mathcal{S}_K , the set

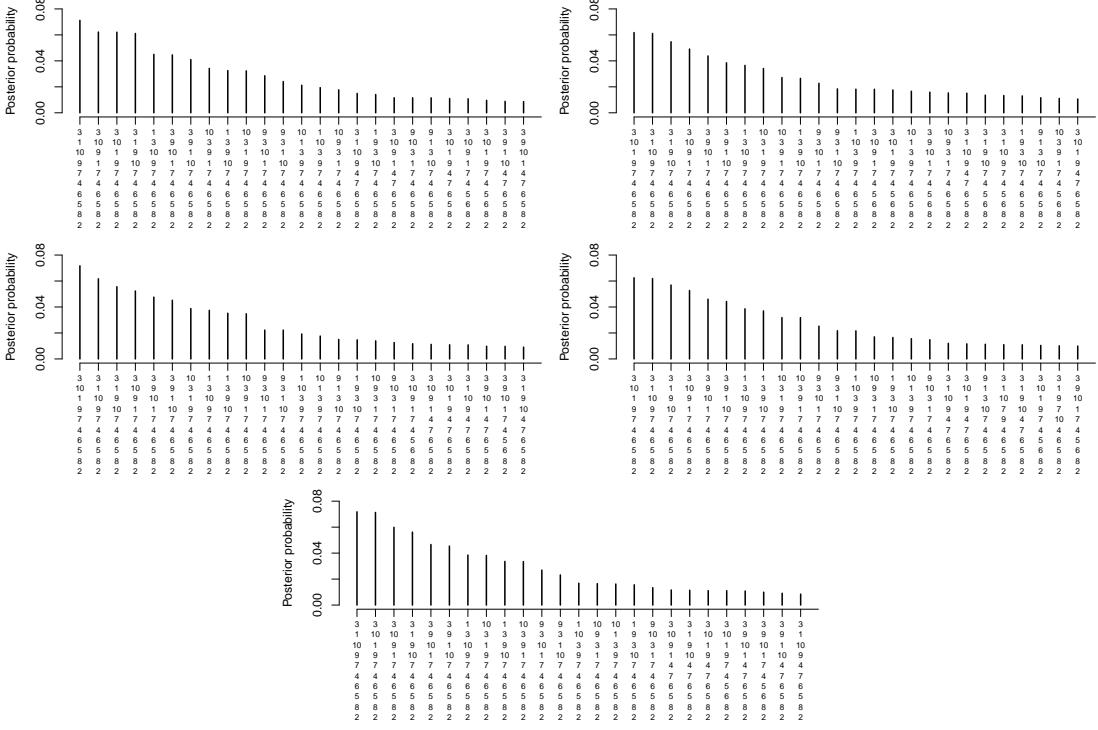


Figure 6.4: Subset of the marginal posterior $\pi(\sigma|\mathcal{D})$ showing the 25 choice orders with highest posterior support from chains 1 to 5 (read from left to right)

of all permutations, we appeal to Metropolis coupled Markov chain Monte Carlo which is the topic of the next section.

6.5 Metropolis coupled Markov chain Monte Carlo

Metropolis coupled Markov chain Monte Carlo (Geyer, 1991), or *parallel tempering* as it is also known within the Bayesian literature, is a sampling technique that aims to improve the mixing of Markov chains in comparison to standard MCMC methods (Gilks and Roberts, 1996). Although Metropolis coupled Markov chain Monte Carlo (MC³) methods can be applied in many scenarios this sampling technique is particularly useful when the target distribution is multi-modal (Brooks, 1998). The general idea behind Metropolis coupled Markov chain Monte Carlo is to consider multiple “Markov” chains – one of which targets the density of interest and the remaining chains target “tempered” densities which are constructed so that they are easier to explore. The multiple chains are then *Metropolis coupled* through the proposal of state space swaps between the chains. It follows that samples from the better mixing chains (those targeting the tempered densities) can filter in to the chain targeting the density of interest and thus improve the exploration of this density. We note that, strictly speaking, each of the chains is not Markovian as, due to the

(Metropolis) coupling of the chains, each chain no longer depends only on the previous value within this chain but also on the previous values of all the other chains. In the next section we discuss how it is easier to construct a Markov chain to efficiently target a tempered density as opposed to the non-tempered density. We conclude this section by describing the methodology underlying parallel tempering and give a generic algorithm outline.

6.5.1 The advantage of targeting tempered densities

Here we discuss how the mixing (and therefore the sampling efficiency) of Markov chains whose target densities are multi-modal is improved when they instead target an appropriate *tempered* density. Suppose we are interested in designing a standard Markov chain Monte Carlo scheme to target the density $\pi(\theta)$ where

$$\theta \sim \frac{1}{2}N(-2, 0.5^2) + \frac{1}{2}N(2, 0.5^2),$$

that is, the density of interest is an equally weighted two-component normal mixture with component means of ± 2 and common standard deviations 0.5. It follows that the target density $\pi(\theta)$ is multi-modal by construction; see Figure 6.5. Of course, given that we know the target density it is straightforward to generate realisations of θ , however, for argument's sake, let us suppose that the distribution of θ is unknown. In this scenario a sensible strategy would be to use a Metropolis-Hastings algorithm to target the density. Given that $\theta \in \mathbb{R}$, a Normal random walk is a plausible proposal distribution, that is, $\theta^* | \theta \sim N(\theta, \sigma^2)$. However, it should be clear that if such an algorithm is initialised at one of the modes of the target distribution then the Markov chain will struggle to “bridge” the area of low probability and explore the other mode. Of course, for this example, we could invent a proposal distribution which encouraged the algorithm to jump between the

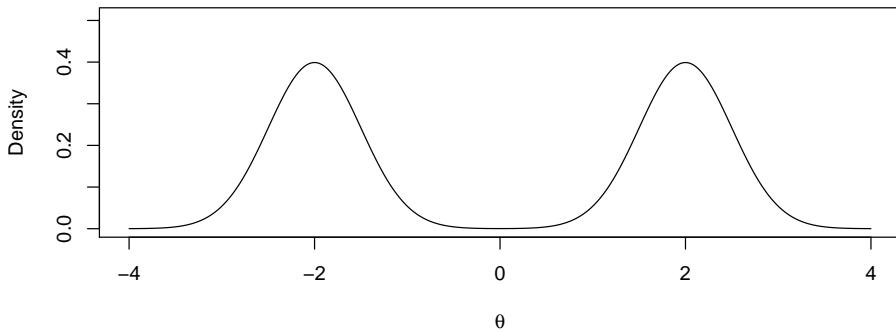


Figure 6.5: $\pi(\theta)$: density plot of an equally weighted two-component normal mixture with component means of ± 2 and standard deviations of 0.05.

modes (e.g. $q(\theta^*|\theta) = \delta_{-\theta}$) as we know the form of the target. However, in general this will not be the case and so constructing a suitable proposal will be non-trivial; especially if the target contains a large number of modes.

Let us now instead consider the *tempered* density $\pi(\theta)^{1/T}$ where $T \geq 1$ is known as the temperature. Clearly the target density of interest is recovered when $T = 1$. Figure 6.6 shows the tempered densities for temperatures $T \in \{1, 2, 4, 8, 16, 32, 64, 128\}$. From this we observe that as we increase the temperature (“heat”) the tempered density becomes “flatter” and it follows that $\pi(\theta)^{1/T}$ is completely flat (uniform) when $T = \infty$. It therefore becomes increasingly more straightforward to construct a Markov chain capable of effectively targeting $\pi(\theta)^{1/T}$ as $T \rightarrow \infty$. Of course, we are only interested in targeting the density $\pi(\theta)^{1/T}$ when $T = 1$. However, as all the tempered densities are inherently related, we can use the accepted samples from a chain targeting a tempered density to improve the

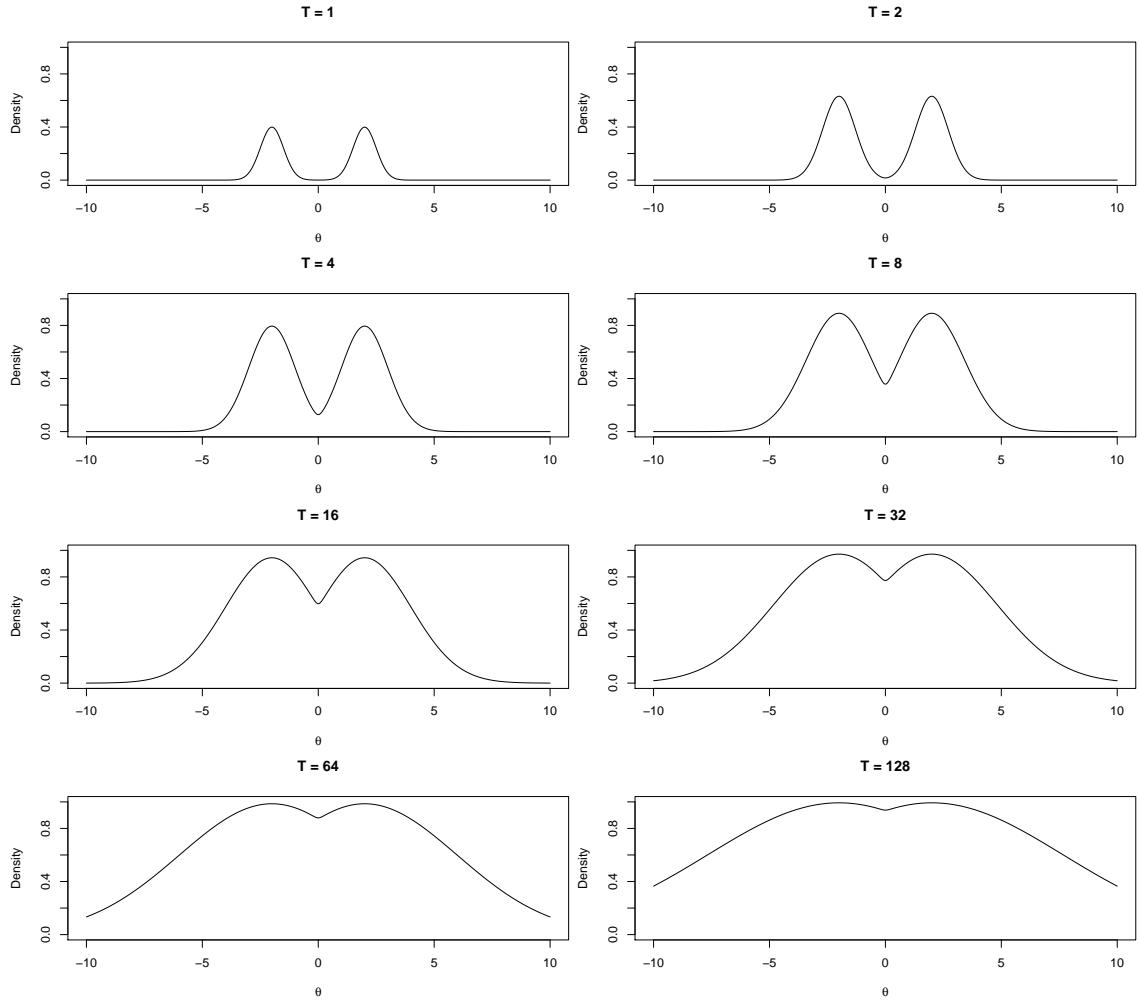


Figure 6.6: Tempered densities $\pi(\theta)^{1/T}$ for $T \in \{1, 2, 4, 8, 16, 32, 64, 128\}$.

mixing within the chain targeting the density of interest. In the next section we discuss how we can take advantage of this technique within a Bayesian inference setting.

6.5.2 Parallel tempering

Parallel tempering (or MC³) is the notion of using tempered densities within a Bayesian inference context. Suppose that we are interested in targeting a (possibly) multi-modal posterior distribution $\pi(\theta|x)$. Now, in contrast to standard MCMC schemes, we construct multiple ‘‘Markov’’ chains. One chain targets the true posterior $\pi(\theta|x)$ and the other chains target tempered posteriors $\tilde{\pi}(\theta|x)$. The multiple chains are then *Metropolis coupled* through a (Metropolis-Hastings) proposal involving state space swaps between the chains and so samples from better mixing chains (targeting the tempered posteriors) can filter in to the chain targeting the true posterior. We now formally define the tempered posteriors and then consider the proposal mechanism for the state space swaps between chains.

Recall from Bayes’ Theorem that the true posterior is

$$\pi(\theta|x) \propto \pi(x|\theta)\pi(\theta)$$

and let the c th tempered posterior be

$$\tilde{\pi}_c(\theta_c|x) \propto \pi(x|\theta_c)^{1/T_c}\pi(\theta)$$

where $T_c > 1$ is the temperature of chain c . We note that the tempered posteriors are formed by only tempering the likelihood component of the true posterior as, given we are working within the Bayesian paradigm, our *a priori* beliefs should be consistent irrespective of the posterior we are targeting.

The issue now is how to swap the states of the chains without affecting their respective target distributions. Suppose we have C chains, that is, a chain to target the true posterior and a further $C - 1$ chains targeting tempered densities. Noting that $\tilde{\pi}_c(\theta_c|x) = \pi(\theta|x)$ when $T_c = 1$ it is useful to let $\mathbf{T} = (1, T_2, T_3, \dots, T_C)$ so that the chains being considered are given by $\tilde{\pi}_c(\theta_c|x)$ for $c = 1, \dots, C$. Now if we consider all C chains to be evolving together then it follows that, as the posteriors $\tilde{\pi}_c(\theta_c|x)$ are conditionally independent given x , together the chains are targeting the joint posterior

$$\pi(\theta_1, \dots, \theta_C|x) = \prod_{c=1}^C \tilde{\pi}_c(\theta_c|x). \quad (6.13)$$

As a brief aside we note that the joint target (6.13) is preserved irrespective of the within-chain updates to the parameters as a consequence of $\tilde{\pi}_i$ and $\tilde{\pi}_j$ being conditionally inde-

pendent. Indeed using different within-chain updates is a sensible strategy given the form of the target density is different for each chain. Returning to the proposal of state space swaps between chains it should be clear that we are simply proposing to swap θ_i and θ_j for some $i \neq j$ within the joint target (6.13). Let $\boldsymbol{\theta} = (\theta_1, \dots, \theta_C)$ denote the current state of the joint chain and so the proposed state of the chain is $\boldsymbol{\theta}^* = (\theta_1^*, \dots, \theta_C^*)$ where $\theta_i^* = \theta_j$, $\theta_j^* = \theta_i$ and $\theta_\ell^* = \theta_\ell$ for $\ell \neq i, j$. Assuming a symmetric proposal mechanism, that is, that the probability of proposing to swap the states of chains (i, j) is the same as the probability of proposing to swap the states of chains (j, i) , the acceptance probability of the state space swap is $\min(1, A)$ where

$$\begin{aligned} A &= \frac{\pi(\boldsymbol{\theta}^*|x)}{\pi(\boldsymbol{\theta}|x)} \\ &= \prod_{c=1}^C \frac{\tilde{\pi}_c(\theta_c^*|x)}{\tilde{\pi}_c(\theta_c|x)} \\ &= \prod_{c=1}^C \frac{\pi(x|\theta_c^*)^{1/T_c} \pi(\theta)}{\pi(x|\theta_c)^{1/T_c} \pi(\theta)} \\ &= \frac{\pi(x|\theta_j)^{1/T_i} \pi(x|\theta_i)^{1/T_j}}{\pi(x|\theta_i)^{1/T_i} \pi(x|\theta_j)^{1/T_j}}. \end{aligned} \tag{6.14}$$

Of course, if the proposal mechanism is not symmetric then the probability A must be multiplied by the proposal ratio $q(\boldsymbol{\theta}|\boldsymbol{\theta}^*)/q(\boldsymbol{\theta}^*|\boldsymbol{\theta})$. Further, it is straightforward to generalise the proposal mechanism above to allow for scenarios where the proposal involves swapping the parameters of more than 2 chains. This should typically be avoided however as such a proposal can have poor acceptance rates; this will be discussed further in Section 6.5.4 where we consider the tuning of proposal distributions within an MC³ sampling scheme. In the next section we consider a general algorithm outline for a MC³ sampling scheme before discussing tuning and a further generalisation of this method.

6.5.3 General algorithm outline

A general algorithm outline of a Metropolis coupled Markov Chain Monte Carlo sampling scheme using C chains is as follows.

- Initialise:
 - let $T_1 = 1$ and choose $T_c > 1$ for $c = 2, \dots, C$
 - let $\boldsymbol{\theta} = (\theta_1, \dots, \theta_C)$
- Repeatedly perform the following steps:

1. For $c = 1, \dots, C$,
 - draw θ_c from $\tilde{\pi}_c(\theta_c|x) \propto \pi(x|\theta_c)^{1/T_c} \pi(\theta)$ using standard MCMC techniques
2. Sample a pair of chain labels (i, j) where $1 \leq i \neq j \leq C$
 - let $\theta_i \rightarrow \theta_j$ and $\theta_j \rightarrow \theta_i$ with probability $\min(1, A)$ where

$$A = \frac{\pi(x|\theta_j)^{1/T_i} \pi(x|\theta_i)^{1/T_j}}{\pi(x|\theta_i)^{1/T_i} \pi(x|\theta_j)^{1/T_j}}$$

The posterior realisations of interest are θ_1 , that is, the accepted draws from chain 1. Given each of the remaining chains target a tempered posterior density the draws from these chains are of no interest and so the collection of samples $(\theta_2, \dots, \theta_C)$ may be discarded.

6.5.4 Tuning a MC³ sampling scheme

Recall that Metropolis coupled Markov chain Monte Carlo schemes consider C chains; each of which targets an alternative (tempered) density. Further, as these chains are conditionally independent given x , each chain can be considered to be an independent standard MCMC scheme targeting its respective density. It follows that each chain should therefore be tuned in a typical fashion, that is, as discussed in Section 1.3.2. Note however that as each chain is targeting an alternative density it is sensible to consider different proposal distributions for each chain. Intuitively we expect larger jumps around the sample space to be more plausible for large T , this follows as the target density becomes “flatter” as the temperature of the chain increases; see Section 6.5.1.

Tuning the between chain proposal (Step 2 of the MC³ algorithm) can be tricky in general. The acceptance rate of swaps between chains is not only affected by the choice of temperatures T_c but also by the mechanism which determines which chains to propose a swap between. Atchadé et al. (2011) show that, when targeting a normal density, the *optimal* acceptance rate (that which maximises the expected squared jumping distance) between chains is 0.234. Of course, in any realistic scenario we are unlikely to be targeting a normal density using MC³ as more standard techniques would be sufficient to generate samples from a normal target. It is generally accepted that between chain acceptance rates of around 20% to 60% provide reasonable mixing (with respect to the joint density of $\theta_1, \dots, \theta_C$); see for example Geyer and Thompson (1995) or more recently Altekar et al. (2004). The question is therefore how to choose the temperatures and the swapping mechanism to achieve such acceptance rates. The strategy we suggest, also advocated by Wilkinson (2013), is to choose the temperatures so that $T_i < T_j$ for $i < j$ and consider swaps between adjacent chains. The intuition behind this strategy is that the target densities become increasingly dissimilar as $|T_i - T_j|$ increases and so swaps are less likely to

be accepted. Further, the temperatures should be chosen so that they exhibit *geometric spacing*, that is, $T_{c+1}/T_c = r$ for some $r > 1$. Note that this reduces the problem of choosing $C - 1$ temperatures to that of choosing T_2 (or equivalently r) as given $T_1 = 1$ and T_2 the remaining temperatures are completely determined. Typically it is sensible to perform a few pilot runs of the MC³ scheme to determine a suitable temperature ratio. Of course, this is only a general strategy and each temperature may need adjusting depending on the situation.

6.5.5 Parallel Metropolis coupled Markov chain Monte Carlo

A notable drawback of using Metropolis coupled Markov chain Monte Carlo sampling schemes is the significant increase in the amount of computation required in comparison to standard MCMC samplers. Recall that an MC³ sampling scheme considers C chains; each of which needs to be updated in order to generate a single sample from the desired target density (note that for standard MCMC schemes only a single chain needs to be updated to generate the same number of samples from the target). It is therefore clear that for MC³ schemes the amount of computation required to generate (posterior) realisations increases linearly with C . However, although it is not possible to reduce the amount of computation per se, it is possible to reduce the time required to perform these computations by appealing to Parallel Metropolis coupled Markov chain Monte Carlo (pMC³) – this method is also known as Multi-core Metropolis coupled Markov chain Monte Carlo or MC⁴ within the literature. From Step 1 in the general algorithm given in Section 6.5.3 it should be clear that standard MC³ considers updating of each chain in serial, that is, we sequentially draw θ_c for $c = 1, \dots, C$, from their respective targets $\tilde{\pi}_c$. Typically this step is the most computationally expensive part of the algorithm as it involves multiple (tempered) likelihood evaluations by construction. However, by exploiting the conditional independence of the chains we can reduce the amount of time required to perform this step; this is the idea behind Parallel Metropolis coupled Markov chain Monte Carlo (pMC³). Recall that $\tilde{\pi}_i$ and $\tilde{\pi}_j$ are conditionally independent given x and so it follows that we can perform within-chain updates of the chains targeting these densities independently without affecting the joint target distribution $\pi(\theta_1, \dots, \theta_C | x)$. Clearly these updates can be performed in parallel across multiple cores. It is important to note however that Step 2 of the MC³ algorithm can only be performed once *all* of the individual chains have been updated, that is, we must wait until the last chain is updated before we proceed.

The number of cores to use (n^{core}) is an important factor when using pMC³; especially given that all C chains must be updated before we proceed to Step 2 of the MC³ algorithm. Naturally we might think that increasing the number of cores will reduce the amount of time it takes to update the C chains and, although this is true in general, it is not quite

n^{core}	Core						Time	
	1	2	3	4	5	6	Total	Relative
1	θ_1	—	—	—	—	—	6	1
	θ_2	—	—	—	—	—		
	θ_3	—	—	—	—	—		
	θ_4	—	—	—	—	—		
	θ_5	—	—	—	—	—		
	θ_6	—	—	—	—	—		
2	θ_1	θ_2	—	—	—	—	3	1/2
	θ_3	θ_4	—	—	—	—		
	θ_5	θ_6	—	—	—	—		
3	θ_1	θ_2	θ_3	—	—	—	2	1/3
	θ_4	θ_5	θ_6	—	—	—		
4	θ_1	θ_2	θ_3	θ_4	—	—	2	1/3
	θ_5	θ_6	—	—	—	—		
5	θ_1	θ_2	θ_3	θ_4	θ_5	—	2	1/3
	θ_6	—	—	—	—	—		
6	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	1	1/6

Table 6.7: Theoretical optimal job allocation across cores with total and relative execution time (assuming job takes a unit time).

that straightforward. To see this we consider an example: suppose we choose to use $C = 6$ chains within a pMC³ sampling scheme and so we must update $\theta_1, \dots, \theta_6$ at each iteration. If we assume that updating θ_c takes unit time then the total time taken to update all $C = 6$ Markov chains for different values of n^{core} can be seen in Table 6.7. Of course, this is a theoretically optimal job schedule which is not achievable in practice (due to additional overheads such as memory allocation and initialisation of cores/jobs) however it illustrates the general idea. Note that when $n^{\text{core}} = 1$ pMC³ is the same as standard MC³, that is, each chain is updated in serial. From Table 6.7 it is clear that we can halve the time required to update the C chains simply by considering two cores. Further, for this example, by taking $n^{\text{core}} = 3$ we reduce the updating step to a third of the time in comparison to standard MC³. Note however that we do not improve on this if $n^{\text{core}} = 4, 5$ as although we can update more chains in the first unit time this is of little benefit as the entire update step is only complete when all C chains are updated. Clearly taking $n^{\text{core}} > C$ will provide no additional gains. It follows that, in general, the best strategy is to choose $n^{\text{core}} = C$ however this may not be possible when considering a large number of chains. In this scenario it is sensible to choose the largest possible number of cores n^{core} which is also a divisor of C , that is, choose the largest possible n^{core} so that $C \bmod n^{\text{core}} = 0$. Altekar et al. (2004) provide further computational details and give more realistic relative performance gains of pMC³ (against MC³) as opposed to the theoretical gains considered here.

6.6 Inference – a Bayesian approach (revisited)

We now return to constructing a Bayesian inference scheme which is capable of generating posterior realisations of both the skill and choice order parameters by implementing a parallel Metropolis coupled Markov chain Monte Carlo algorithm to target the joint posterior distribution. This section will then conclude with a simulation study where we show that the mixing of our Markov chains is improved considerably by using MC³ as opposed to standard MCMC techniques developed in Section 6.4.

6.6.1 A pMC³ algorithm for the EPL model

A parallel Metropolis coupled Markov chain Monte Carlo algorithm to target the joint posterior distribution of the skill parameters $\boldsymbol{\lambda}$ and the choice order parameter $\boldsymbol{\sigma}$ is as follows.

1. Tune:

- choose the number of chains (C) and the number of cores (n^{cores})
- let $T_1 = 1$ and choose $T_c > 1$ for $c = 1, \dots, C$

2. Initialise: choose $\boldsymbol{\sigma}_c \in \mathcal{S}_K$ and $\boldsymbol{\lambda}_c \in \mathbb{R}_{>0}^K$ for $c = 1, \dots, C$

3. For $c = 1, \dots, C$ perform, in parallel, the following steps:

- For $k = 1, \dots, K$
 - draw $\lambda_{ck}^\dagger | \lambda_{ck} \sim \text{LN}(\log \lambda_{ck}, \sigma_{\lambda_{ck}}^2)$
 - let $\lambda_{ck} \rightarrow \lambda_{ck}^\dagger$ with probability $\min(1, A)$ where

$$A = \left\{ \frac{\pi(\mathcal{D}|\boldsymbol{\lambda}_{c,-k}, \lambda_{ck} = \lambda_{ck}^\dagger, \boldsymbol{\sigma}_c)}{\pi(\mathcal{D}|\boldsymbol{\lambda}_c, \boldsymbol{\sigma}_c)} \right\}^{1/T_c} \times \left(\frac{\lambda_{ck}^\dagger}{\lambda_{ck}} \right)^{a_k} e^{(\lambda_{ck} - \lambda_{ck}^\dagger)}$$

- Sample ℓ from the discrete distribution with probabilities $\Pr(\ell = i) = p_{i,c}^{\text{prop}}$ for $i = 1, \dots, 5$
 - propose $\boldsymbol{\sigma}_c^\dagger$ using proposal mechanism ℓ
 - let $\boldsymbol{\sigma}_c \rightarrow \boldsymbol{\sigma}_c^\dagger$ with probability $\min(1, A)$ where

$$A = \left\{ \frac{\pi(\mathcal{D}|\boldsymbol{\lambda}_c, \boldsymbol{\sigma}_c^\dagger)}{\pi(\mathcal{D}|\boldsymbol{\lambda}_c, \boldsymbol{\sigma}_c)} \right\}^{1/T_c} \times \frac{\Pr(\boldsymbol{\sigma} = \boldsymbol{\sigma}_c^\dagger)}{\Pr(\boldsymbol{\sigma} = \boldsymbol{\sigma}_c)}.$$

- Rescale
 - sample $\Lambda_c^\dagger \sim \text{Ga} \left(\sum_{k=1}^K a_k, 1 \right)$.
 - calculate $\Sigma_c = \sum_{k=1}^K \lambda_{ck}$.
 - let $\lambda_{ck} \rightarrow \lambda_{ck} \Lambda_c^\dagger / \Sigma_c$ for $k = 1, \dots, K$.
- 4. Sample a pair of chain labels (i, j) where $1 \leq i \neq j \leq C$
 - let $(\boldsymbol{\lambda}_i, \boldsymbol{\sigma}_i) \rightarrow (\boldsymbol{\lambda}_j, \boldsymbol{\sigma}_j)$ and $(\boldsymbol{\lambda}_j, \boldsymbol{\sigma}_j) \rightarrow (\boldsymbol{\lambda}_i, \boldsymbol{\sigma}_i)$ with probability $\min(1, A)$ where
$$A = \frac{\pi(\mathcal{D}|\boldsymbol{\lambda}_j, \boldsymbol{\sigma}_j)^{1/T_i} \pi(\mathcal{D}|\boldsymbol{\lambda}_i, \boldsymbol{\sigma}_i)^{1/T_j}}{\pi(\mathcal{D}|\boldsymbol{\lambda}_i, \boldsymbol{\sigma}_i)^{1/T_i} \pi(\mathcal{D}|\boldsymbol{\lambda}_j, \boldsymbol{\sigma}_j)^{1/T_j}}$$
- 5. Return to Step 3.

6.6.2 Simulation study

For this study we again revisit Dataset 6 which we analysed previously using standard MCMC techniques in Section 6.4. Recall that these data comprise $n = 100$ *complete* rankings of $K = 10$ entities. The skill parameters and choice order (*ranking process*) used to simulate these data are $\boldsymbol{\lambda} = (10, 9, \dots, 1)$ and $\boldsymbol{\sigma} = (3, 10, 9, 1, 7, 4, 5, 6, 8, 2)$ respectively. Here we perform Bayesian inference using the pMC³ algorithm described in Section 6.6.1 with $C = 5$ chains running across $n^{\text{cores}} = C = 5$ cores. We adopt the same prior distribution as for previous analyses of these data, that is, $a_k = a = 1$ and each choice order is chosen to be equally likely *a priori*. The motivation behind this choice of prior distribution is given in Sections 6.4.7 and 6.4.10. Also the proposal distribution for the choice order parameter is chosen to be the same as before; namely $s = 2$, $\tau = 3$ and $\mathbf{p}^{\text{prop}} = (0.3, 0.3, 0.3, 0.05, 0.05)$ within each chain.

Before we can perform Bayesian inference using the pMC³ algorithm from Section 6.6.1 we must also choose appropriate temperatures for each chain along with a suitable mechanism for proposing the between-chain swaps. We choose the temperatures to be $\mathbf{T} = (1, 0.75^{-1}, 0.6^{-1}, 0.5^{-1}, 0.4^{-1})$ which were determined after making some manual adjustments (guided by pilot runs) to the temperatures obtained using geometric spacing with ratio 4/3, that is, $T_{c+1}/T_c = 4/3$. Further, as discussed in Section 6.5.4, we consider it sensible to only consider swaps between two adjacent chains and we sample the chain labels $(i, i + 1)$ uniformly at random. The resulting between-chain acceptance rates are around 50% to 60% which we consider acceptable and should allow us to benefit from using an MC³ scheme. Of course, we must also choose a tuning parameter for the Log-normal random walk proposal which generates the proposed skill parameters. As in the previous

analyses we choose $\sigma_{\lambda_k} = \sigma_\lambda = 0.75$ which gives acceptance rates around 19% to 25% (within each chain).

In this study we initialise each of the C chains (within our *single* inference scheme) from the prior distribution, that is, we draw $\lambda_{ck} \stackrel{\text{indep}}{\sim} \text{Ga}(1, 1)$ and uniformly sample $\boldsymbol{\sigma}_c$ from the set of all permutations \mathcal{S}_K for $c = 1, \dots, C$ and $k = 1, \dots, K$. The results reported are from a typical run of our pMC³ scheme initialised as described, with a burn-in of 20K iterations and then run for a further 2M iterations and thinned by 200 to obtain 10K (almost) unautocorrelated posterior samples. The pMC³ scheme runs reasonably quickly, with C code on $n^{\text{cores}} = C = 5$ cores of an Intel Core i7-4790S CPU (3.20GHz clock speed) taking around 35 minutes 20 seconds. Note the equivalent analysis takes around 123 minutes and 45 seconds under a standard (serial) MC³ implementation ($n^{\text{cores}} = 1$) and so clearly using parallel computing is advantageous. Also, due to the additional likelihood evaluations required to propose the between chain swaps the pMC³ scheme requires more CPU time than the standard Metropolis-Hastings MCMC scheme from Section 6.4.10 where the equivalent analysis took approximately 24 minutes 30 seconds.

Figure 6.7 shows a trace plot of the log-likelihood (left) and also the marginal posterior distribution of the choice order $\boldsymbol{\sigma}$ (right). Here, in contrast to the previous Bayesian analyses, the marginal posterior distribution for $\boldsymbol{\sigma}$ obtained from numerous pMC³ schemes (initialised at different values) are consistent up to stochastic noise (not reported here) and so, for this example, MC³ has enabled us to generate posterior realisations. Figure 6.8 shows the 25 choice orders with highest posterior support and their corresponding posterior probabilities. Note that the choice order used to simulate these data is shown in red and has posterior probability $\Pr(\boldsymbol{\sigma} = (3, 10, 9, 1, 7, 4, 5, 6, 8, 2) | \mathcal{D}) = 0.011$. It is perhaps not surprising that we do not see large posterior support for the choice order used to simulate these data (or indeed any choice order) given we are only considering $n = 100$ observations and there are $10!$ possible choice orderings. That said, it is pleasing to see that the choice orders with largest posterior support are fairly similar to that which was used to simulate these data. Another interesting observation from Figure 6.8 is that it appears we can clearly identify the lower positions within the choice order and much of the uncertainty resides within the first 4 entries of $\boldsymbol{\sigma}$.

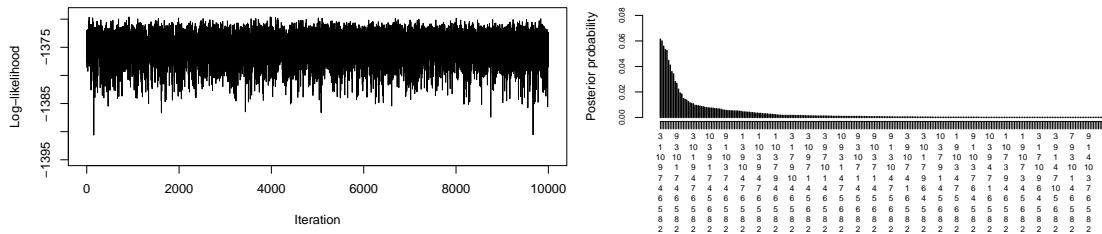


Figure 6.7: Trace plot of the log-likelihood (left) and the marginal posterior $\pi(\boldsymbol{\sigma} | \mathcal{D})$ (right).

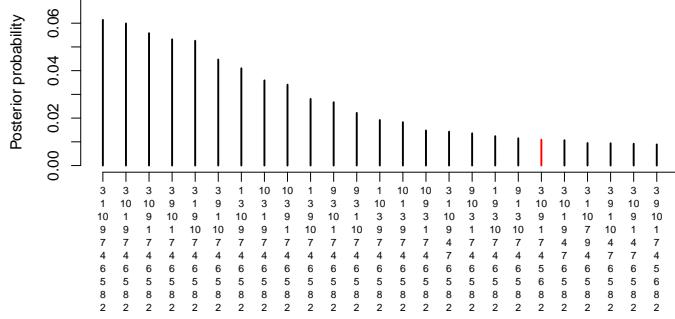


Figure 6.8: Subset of the marginal posterior $\pi(\sigma|\mathcal{D})$ showing the 25 choice orders with highest posterior support (red denotes choice order used to simulate these data)

Figure 6.9 depicts boxplots of the marginal posterior distribution for each $\log \lambda_k$ conditional on both the posterior modal choice order ($\sigma = (3, 1, 10, 9, 7, 4, 6, 5, 8, 2)$) and the “true” choice order ($\sigma = (3, 10, 9, 1, 7, 4, 5, 6, 8, 2)$) in white and red respectively. The blue crosses denote the true values from which these data were simulated and we note that we have rescaled the values so that $\lambda_{10} = 1$ is constant and therefore omitted from the plot along with any outliers. Clearly there is large posterior uncertainty on the values of the skill parameters, however this is perhaps not surprising given the associated uncertainty on the choice order (and the small number of observations). Further, these boxplots are constructed based on a relatively small number of (posterior) realisations due to the conditioning on a particular choice order. However, it is pleasing to see that the marginal posterior for the skill parameters are coherent under both of the selected choice orders and the aggregate rankings (formed by ordering the skill parameters on their posterior mean) are $\mathbf{x}^{\text{agg}} = (4, 3, 1, 2, 5, 6, 8, 7, 9, 10)$ under the posterior modal permutation and $\mathbf{x}^{\text{agg}} = (2, 1, 3, 4, 5, 6, 7, 8, 9, 10)$ under the true permutation – recall the true preference of entities is $(1, \dots, 10)$.

It is also of interest to see if there is any benefit to using the complicated EPL model and whether we can instead draw reasonable inferences about these data by using the standard (forward ranking) Plackett–Luce model. To answer this question we analyse these data using the (Gibbs sampling) algorithm for the standard PL model from Chapter 2. Figure 6.10 depicts boxplots of the marginal posterior distribution for each $\log \lambda_k$ under the top choice order from the EPL model and those obtained under the standard PL in white and green respectively. From this it is clear that the standard Plackett–Luce model is not a good fit for these data and, under such an analysis, we would conclude that the aggregate ranking is $\mathbf{x}^{\text{agg}} = (2, 8, 5, 7, 4, 9, 6, 1, 3, 10)$ and there is little discrepancy between the preference of the entities – which we know not to be true. It follows that the standard Plackett–Luce model is not a good model for these data and we should expect to obtain “better” inferences when using the EPL if the data are not generated from the forward ranking process.

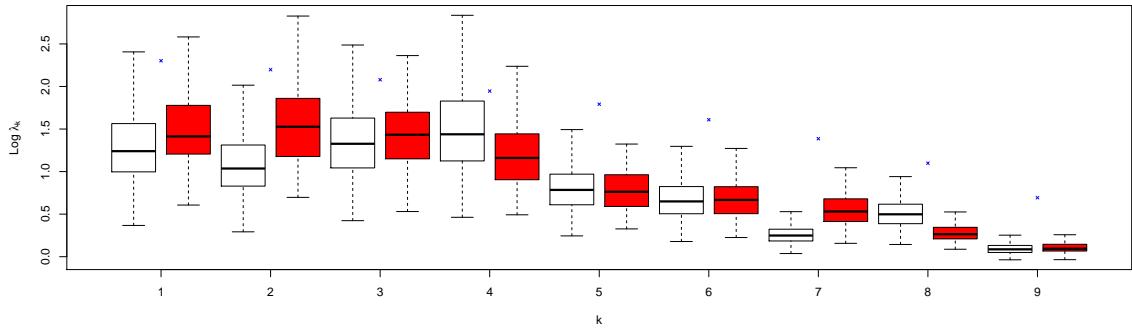


Figure 6.9: Boxplots summarising the marginal posterior densities for each $\log \lambda_k$ given that $\lambda_{10} = 1$. The densities in each case are shown in white and red for those obtained under the choice order with the largest posterior support and the true choice order, respectively. The blue crosses depict the true values from which these data were simulated (log scale).

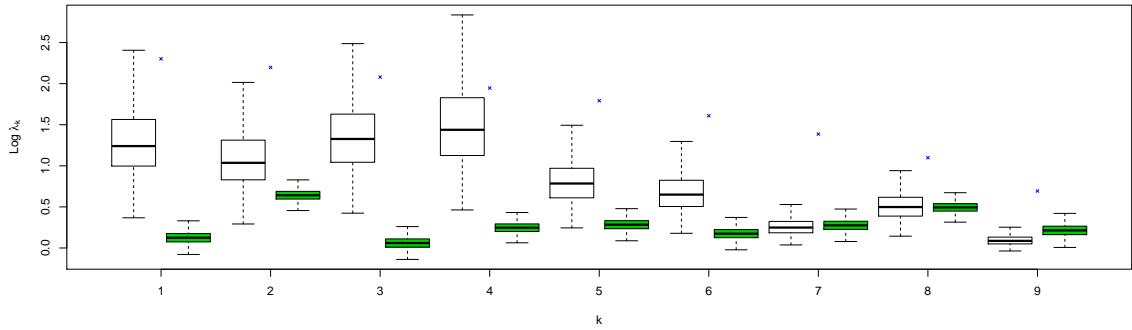


Figure 6.10: Boxplots summarising the marginal posterior densities for each $\log \lambda_k$ given that $\lambda_{10} = 1$. The densities in each case are shown in white and green for those obtained under the EPL model (for the choice order with the largest posterior support) and those obtained under the standard Plackett–Luce model ($\sigma = (1, \dots, K)$) respectively. The blue crosses depict the true values from which these data were simulated (log scale).

6.7 Summary

In this chapter we have described the Extended Plackett–Luce model which allows the *a priori* assumption of an explicit choice order (ranking process) to be relaxed. The choice order, which is an element of the set of all permutations \mathcal{S}_K , is instead represented by an additional free parameter within the model. We described the proposed model in detail and considered whether the choice order was identifiable given a collection of rankings. In Section 6.3 we verified that each alternative choice order results in changes to the (maximum) likelihood by means of a simulation study using maximum likelihood estimation.

The remainder of this chapter focused on the challenging problem of performing a fully Bayesian analysis of the Extended Plackett–Luce model. In Section 6.4.4 we considered

numerous “swap” moves which we hoped would allow us to effectively explore the set of all permutations when the number of entities is large. Unfortunately, the data augmentation approach (to give Gibbs updates of the skill parameters) was ineffective which we believe is caused by the large posterior correlation between the parameters within this model. To overcome this we considered a Metropolis-Hastings sampling approach and we saw through simulation studies in Section 6.4.10 that, although this approach showed promise, the mixing of the Markov chains remained poor. The natural step was to then consider Metropolis coupled Markov chain Monte Carlo (MC³) and in Section 6.5 we spent time exploring this idea and gave a generic algorithm outline and discussed how parallel computing can be used to increase the speed of inference schemes. In Section 6.6.2 we considered a simulation study using a pMC³ sampling scheme which gave promising results and the Markov chain appeared to effectively explore the posterior. That said, future work is needed to verify that this approach will also work for larger datasets – our intuition leads us to believe that the inference problem for the EPL may become too challenging for more than say $K = 20$ entities. It would also be nice to increase modelling flexibility by appealing to Dirichlet process mixtures of EPL models but this too will increase the difficulty of posterior sampling.

In the next chapter we conclude this thesis and provide an overview of the main results along with some potential directions for future work.

Chapter 7

Conclusions

The intention of this thesis was to explore flexible models which allow the identification of (possible) subgroup structure within ranked data. Further, we wanted our modelling framework to be able to capture potential heterogeneity between the abilities of rankers. We have also investigated the effect of the ranking process and developed methods for increasing modelling flexibility further by relaxing the assumption of an explicit ranking process.

We first considered the (standard) Plackett–Luce model (Luce, 1959; Plackett, 1975). Through simulation studies it was shown how inferences from this model can be affected by even a modest amount of spurious rankings. It follows that this model is not well suited to handle a scenario where some of the rankers are not well informed about the entities being ranked. We extended the standard Plackett–Luce model to the novel Weighted Plackett–Luce (WPL) model where the WPL model allows for (potential) differing reliability through a two component mixture model. Bayesian inference for this model is made straightforward by a slight extension of existing data augmentation approaches (Caron and Doucet, 2012) that yields an efficient Gibbs sampling scheme. Simulation studies showed that the Weighted Plackett–Luce model is able to correctly identify spurious rankers (when present) within a collection of data. Further, in contrast to the standard Plackett–Luce model, inferences from the WPL model were shown to be fairly robust to the addition of uninformative rankings.

In Chapter 3 we presented models capable of exploring the (possible) subgroup structure within ranked data. More specifically we aimed to identify homogeneous groups of individuals who share similar beliefs and also looked at how a single group of rankers may struggle to distinguish between certain entities. To implement this structure we appealed to Bayesian non-parametric clustering methods, specifically by using Dirichlet process mixture models (Ferguson, 1973; Antoniak, 1974). We presented two models. The first al-

lowed for the notion that rankers may be heterogeneous in their beliefs about the entities. Both finite and Dirichlet process mixtures of (standard) Plackett–Luce models have been explored within the literature and we extended this approach slightly by building a model that comprises a Dirichlet process mixture of Weighted Plackett–Luce models. The second model we presented allows for the notion that a homogeneous group of rankers may not be able to distinguish between certain groups of entities, that is, they might consider some entities to be indistinguishable (tied in strength). We allowed for this structure by considering a Dirichlet process mixture over the skill parameters in the Weighted Plackett–Luce model. To the best of our knowledge this approach has not previously been considered within the literature. Simulation studies showed that this model proved to be effective at detecting groups of entities and also performed reasonably well when no entity groups were present which was reassuring. Further, this model allowed us to quantify the level of (posterior) similarity between entities in a principled manner; this would require an *ad hoc* approach if using standard (no clustering) techniques. Bayesian inference for each model proceeds via efficient marginal sampling schemes (Neal, 2000).

Chapter 4 presented the Weighted Adapted Nested Dirichlet (WAND) process mixture of Plackett–Luce models. This model combines the aspects of each model presented in the previous chapters. More specifically this model allows for both ranker and entity clustering along with the possibility of (potential) differing ranker reliability. Allowing for both ranker and entity clustering was achieved by appealing to two-way clustering techniques; specifically by adapting the Nested Dirichlet process prior of Rodriguez et al. (2008) so that it could handle ranked data. The (WAND) model was then formed by taking the Adapted Nested Dirichlet process as the prior distribution over the skill parameters and the Weighted Plackett–Luce model as the ranking distribution. Both conditional and marginal approaches to posterior inference were presented for this model. The modelling framework described also allows for inferences to be made using only incomplete rankings, such as top- M or partial rankings. We saw through the simulation studies that reasonable inferences can be made under the WAND model even when only limited (partial) information is available. Although not considered here, in Section 2.2.4 we described how ties within rankings can easily be accounted for within our simulation based inference approach. The richness of information in the posterior distribution allows us to infer many details of the structure both between ranker groups and between entity groups (within ranker groups), in contrast to many other (Bayesian) analyses. The high dimensionality of the posterior distribution can make the production of insightful but simple summaries quite difficult and we explored different approaches, ranging from conditioning on the modal number of groups to adopting a classification based on calculations from a dissimilarity matrix summary. Chapter 5 contained analyses of several real datasets that have been analysed in the literature, and we compared their conclusions with those obtained from fitting the

WAND model. In general we found that the WAND model is well suited to modelling ranked data and provides valuable insight into subgroup structure within ranked data which would not be possible under other models.

We also considered relaxing the assumption of a known ranking process by looking at the recently developed Extended Plackett–Luce model (Mollica and Tardella, 2014). In this model the ranking process (“choice order”) is instead represented by an additional free parameter which is an element of the set of all permutations \mathcal{S}_K . Some insight into the model was provided and we also discussed the identifiability of the ranking process. To motivate the identifiability of the ranking process further we considered a simulation study which showed that alternative choice orders result in changes to the (maximised) likelihood. We then considered the challenging problem of performing Bayesian inference for the Extended Plackett–Luce model. Our aim was to extend the solution of Mollica and Tardella (2018) by considering an unrestricted sample space for the choice order parameter. We presented several “swap” moves which we hoped would allow us to effectively explore the set of all permutations when the number of entities is large. Unfortunately, the data augmentation approach (to give Gibbs updates of the skill parameters) was ineffective and we believe this was caused by the large posterior correlation between the parameters within this model. In an attempt to overcome this problem, we removed the latent parameters from the model and instead considered a Metropolis-Hastings sampling approach and, although this approach showed promise, it was evident (through simulation studies) that the mixing of the Markov chains remained poor. To improve mixing we appealed to Metropolis Coupled Markov chain Monte Carlo (MC³). A simulation study using a (parallel) MC³ sampling scheme gave promising results and it appeared that we are able to explore the posterior distribution effectively. We note that, care must be taken when performing Bayesian inference for the EPL as our solution is unlikely to scale well to scenarios where the number of entities is large.

7.1 Future work

We believe this research offers plenty of opportunity for extension and future work within the Bayesian analysis of ranked data. For example, it may be possible to remove the binary ranker weights w_i from the WAND model and instead introduce a “spam cluster” to house the uninformative rankers. In the finite mixture setting this would be reasonably straightforward to achieve. However, for infinite mixture models, it is somewhat more complicated. To be equivalent to a specification of $w_i = 0$, the spam cluster would need to contain only a single entity cluster and this is unlikely to arise unless the concentration parameter of the mixture distribution is chosen to be small. Of course, we would not want

the (entity clustering) concentration parameter to be small within all ranker groups as we would still want to learn about the entity subgroup structure for groups of rankers that are informative. One strategy could be to choose the prior distribution of the concentration parameters to be a mixture where one of the components is heavily concentrated on small values. However, our intuition leads us to believe that this approach may result in numerous “spam” clusters, making the identification of uninformative rankers somewhat more complicated. It may therefore be more advantageous to handle this aspect with the binary ranker weights. Nevertheless, this merits further research.

Although our work focused mainly on the (Weighted) Plackett–Luce model, the ANDP prior is also well suited to other parametric ranking models. A natural model to consider is the Benter model (Benter, 1994) given that it is a straightforward extension of the (standard) PL model. The Benter model has additional parameters that represent the “importance” of each stage in the ranking process. In theory it would also be possible to introduce our binary ranker indicators and hence consider a Weighted Benter model. However this may also give rise to identifiability issues. Further, given the forward ranking process implicitly implies that there is more uncertainty within the early ranks of an observation, the “position importance” parameter within the Benter model is unlikely to be able to adequately handle a scenario where this is not the case, that is, when there is more certainty about entities in the top ranks than those in the latter ranks. It would however be interesting to see whether the Benter model is capable of mitigating this artefact of the forward ranking process.

Our exploration of the Extended Plackett–Luce model has opened many potential avenues for further research. The associated inference problem is challenging and, although our inference scheme appears to be adequate for a modest number of entities, it is unlikely to perform well in scenarios where the number of entities is large. It may be possible to avoid explicitly considering the set of all permutations and instead consider a continuous (multivariate) parameter that lies on the $(K - 1)$ -dimensional simplex (with the implied permutation being given by ordering the realisation). Constructing a suitable proposal distribution is likely to be more straightforward in this scenario and, although multimodality could still be an issue, it may be possible to overcome this by appealing to Hamiltonian Monte Carlo (Duane et al., 1987; Neal, 2011). Once the inference problem has been explored further it would be interesting to extend modelling flexibility by considering finite or infinite mixtures of Extended Plackett–Luce models. We note however that the ANDP prior is probably not well suited to this scenario as the entity clustering may render the ranking process unidentifiable; further analytical/empirical work would be required to verify this. Finally, a thought provoking topic of potential research is the notion of a “Hierarchical Plackett–Luce” model. Instead of considering the ranking process parameter in the Extended Plackett–Luce model to be a permutation, we could instead consider it

to be a ranking itself. This ranking (the ranking process) could then be modelled using a Plackett–Luce model and so each position in the ranking process would itself have a corresponding skill parameter. If desired the ranking process could also be modelled by another Extended Plackett–Luce model and so the nested layers of Plackett–Luce models could keep going on and on . . .

Appendix A

Miscellaneous

A.1 Derivation of the FCD for DP concentration parameter α

Central to the implementation of a Dirichlet process mixture model is the choice of concentration parameter α . It is often the case that we wish to infer this parameter from the data which is possible if we incorporate α into our analysis. Assuming we have n samples and a continuous density $\pi(\alpha)$ it has been shown by Antoniak (1974) that the implied prior on the number of clusters N^c is

$$\pi(N^c|\alpha, n) = c_n(N^c)n!\alpha^{N^c} \frac{\Gamma(\alpha)}{\Gamma(\alpha + n)} \quad (\text{A.1})$$

for $N^c = 1, \dots, n$.

The function $c_n(N^c)$ is defined as the density of the number of clusters conditional on $\alpha = 1$, that is, $c_n(N^c) = \pi(N^c|\alpha = 1, n)$ and hence does not involve α . If we suppose we have sampled our parameters Λ then it should be clear that we have also obtained a sample of N^c (the number of clusters) which is given by the number of unique entries within Λ . Furthermore we also assume we have a sample of our latent cluster indicators, \mathbf{c} , and thus the configuration of these data into the N^c groups is also known. Now, given that N^c , Λ and \mathbf{c} are known it can be shown that the data \mathcal{D} are conditionally independent of α . It follows that, for $\alpha > 0$

$$\pi(\alpha|N^c, \Lambda, \mathcal{D}) = \pi(\alpha|N^c) \propto \pi(N^c|\alpha)\pi(\alpha),$$

and, substituting in (A.1) gives

$$\begin{aligned}
 \pi(\alpha|N^c, \Lambda, \mathcal{D}) &\propto c_n(N^c) n! \alpha^{N^c} \frac{\Gamma(\alpha)}{\Gamma(\alpha+n)} \pi(\alpha) \\
 &\propto \alpha^{N^c} \frac{\Gamma(\alpha)}{\Gamma(\alpha+n)} \pi(\alpha) \\
 &= \alpha^{N^c} \frac{(\alpha+n)B(\alpha+1, n)}{\alpha\Gamma(n)} \pi(\alpha) \\
 &\propto \alpha^{N^c-1} (\alpha+n) B(\alpha+1, n) \pi(\alpha) \\
 &= \alpha^{N^c-1} (\alpha+n) \int_0^1 x^\alpha (1-x)^{n-1} dx \pi(\alpha).
 \end{aligned} \tag{A.2}$$

Fortunately we can construct a joint distribution of α and a quantity $\eta \in (0, 1)$ which has (A.2) as its marginal density. This joint density takes the form, for $\alpha > 0$ and $\eta \in (0, 1)$

$$\pi(\alpha, \eta|N^c) \propto \alpha^{N^c-1} (\alpha+n) \eta^\alpha (1-\eta)^{n-1} \pi(\alpha). \tag{A.3}$$

Single component prior

In order to maintain conjugacy we assign α a mixture of Gamma distributions *a priori*. If we first consider the simple case where there is only a single mixture component, that is, $\alpha \sim \text{Ga}(a_0, b_0)$, it follows that the joint density of α and η is, for $\alpha > 0$ and $\eta \in (0, 1)$

$$\begin{aligned}
 \pi(\alpha, \eta|N^c) &\propto \alpha^{N^c-1} (\alpha+n) \eta^\alpha (1-\eta)^{n-1} \frac{b_0^{a_0}}{\Gamma(a_0)} \alpha^{a_0-1} e^{-\alpha b_0} \\
 &= \frac{b_0^{a_0}}{\Gamma(a_0)} \alpha^{a_0+N^c-2} e^{-\alpha b_0} (\alpha+n) \eta^\alpha (1-\eta)^{n-1}.
 \end{aligned} \tag{A.4}$$

From (A.4) the marginal distribution for α is, for $\alpha > 0$,

$$\begin{aligned}
 \pi(\alpha|\eta, N^c) &\propto \frac{b_0^{a_0}}{\Gamma(a_0)} \alpha^{a_0+N^c-2} e^{-\alpha b_0} (\alpha+n) \eta^\alpha \\
 &\propto \frac{b_0^{a_0}}{\Gamma(a_0)} \alpha^{a_0+N^c-2} e^{-\alpha(b_0-\log\eta)} (\alpha+n) \\
 &= \frac{b_0^{a_0}}{\Gamma(a_0)} \alpha^{a_0+N^c-1} e^{-\alpha(b_0-\log\eta)} + n \frac{b_0^{a_0}}{\Gamma(a_0)} \alpha^{a_0+N^c-2} e^{-\alpha(b_0-\log\eta)},
 \end{aligned}$$

and so we have

$$\alpha|\dots \sim \pi_0 \text{Ga}(a_0 + N^c, b_0 - \log\eta) + (1 - \pi_0) \text{Ga}(a_0 + N^c - 1, b_0 - \log\eta).$$

The mixture weighting π_0 is given by

$$\begin{aligned}\pi_0 &= \frac{b_0^{a_0}}{\Gamma(a_0)} \int_0^\infty \alpha^{a_0+N^c-1} e^{-\alpha(b_0-\log \eta)} \\ &\implies \pi_0 = \frac{b_0^{a_0}}{\Gamma(a_0)} \frac{\Gamma(a_0 + N^c)}{(b_0 - \log \eta)^{a_0 + N^c}},\end{aligned}\tag{A.5}$$

and

$$\begin{aligned}(1 - \pi_0) &= \frac{nb_0^{a_0}}{\Gamma(a_0)} \int_0^\infty \alpha^{a_0+N^c-2} e^{-\alpha(b_0-\log \eta)} \\ &\implies (1 - \pi_0) = \frac{nb_0^{a_0}}{\Gamma(a_0)} \frac{\Gamma(a_0 + N^c - 1)}{(b_0 - \log \eta)^{a_0 + N^c - 1}},\end{aligned}\tag{A.6}$$

whence

$$\frac{\pi_0}{(1 - \pi_0)} = \frac{a_0 + N^c - 1}{n(b_0 - \log \eta)}.\tag{A.7}$$

From (A.4) we also deduce the conditional distribution for η to be, for $\eta \in (0, 1)$

$$\pi(\eta|\cdot) \propto \eta^\alpha (1 - \eta)^{n-1},$$

that is

$$\eta|\cdot \sim \text{Beta}(\alpha + 1, n).\tag{A.8}$$

Appendix B

Datasets

Rank	Ranker																																							
	1	...	10	...	20	...	30	...	40																															
218	3	12	4	6	3	9	19	3	2	11	14	6	3	16	6	10	7	5	11	3	3	16	2	3	6	10	2	5	2	18	1	6	6	5	12	13	19	3		
	4	19	1	3	6	1	15	13	5	4	1	11	18	18	5	7	9	16	1	3	5	4	5	1	11	1	9	17	3	19	11	6	3	4	12	11	10	3	9	1
	1	6	8	5	7	17	4	5	10	14	3	7	6	9	12	1	1	13	3	2	8	2	2	9	10	4	8	5	5	10	3	4	8	1	1	2	9	5	1	7
	13	11	15	2	4	2	2	11	3	1	13	3	4	4	10	8	2	4	5	18	7	9	19	3	3	5	4	2	4	7	1	14	7	9	7	10	5	9	10	13
	2	13	5	14	8	12	1	10	16	13	12	4	5	2	3	5	4	6	8	13	2	8	1	14	1	2	12	7	14	1	7	8	4	2	2	1	1	12	17	6
	7	10	9	8	10	3	8	12	9	3	9	5	2	5	14	13	11	7	6	11	14	5	6	6	8	13	1	1	10	6	8	7	10	8	14	19	2	8	18	9
	8	3	13	9	17	8	10	1	15	7	2	10	1	8	20	3	19	2	17	17	12	12	7	10	14	16	7	8	1	16	13	1	11	7	13	4	15	11	7	5
	9	4	19	11	9	10	6	6	1	6	17	1	12	1	8	2	18	3	4	4	17	15	10	17	16	6	3	9	18	2	9	5	9	11	5	7	13	2	8	2
	11	2	12	19	12	4	5	19	4	5	8	8	8	3	11	15	6	1	7	8	3	10	11	7	5	7	16	13	16	11	6	3	5	13	3	18	14	6	13	12
	12	16	2	15	1	11	3	14	18	2	7	15	11	7	2	4	10	17	2	9	6	17	9	4	9	9	5	4	7	4	5	12	12	10	19	9	8	7	16	15
	6	9	14	17	11	6	17	15	12	9	4	2	9	6	4	9	12	11	9	12	1	6	4	5	13	8	13	15	6	3	16	15	6	14	8	14	7	1	3	8
	20	8	6	16	5	5	9	16	6	10	15	13	14	13	16	16	8	8	15	10	15	11	8	15	6	10	14	11	12	18	4	2	15	3	16	3	19	10	6	10
	17	5	7	10	16	7	11	9	14	8	5	12	13	17	9	11	14	10	14	1	13	13	14	2	4	12	2	6	13	12	12	9	19	5	4	17	11	16	2	11
	10	7	3	4	13	20	7	7	11	12	6	16	16	12	7	6	15	20	13	7	10	18	15	8	12	11	11	3	11	9	18	10	13	12	10	6	4	4	4	
	16	1	10	7	2	14	12	8	7	15	20	14	10	10	1	12	5	9	10	14	4	7	17	13	18	15	10	14	8	8	10	19	16	16	17	15	3	17	5	16
	5	15	16	12	19	16	14	2	8	17	10	9	17	19	17	14	16	14	12	16	9	1	16	11	17	14	18	12	15	15	14	16	2	15	9	12	20	20	11	18
	14	14	17	1	20	13	13	18	17	16	16	18	7	11	18	17	3	15	16	15	16	14	20	20	7	18	15	16	9	13	17	11	14	17	11	8	6	14	12	14
	15	20	18	13	18	15	18	17	19	18	11	17	20	14	13	20	13	12	18	6	19	16	13	18	20	19	19	20	17	14	15	17	18	19	18	13	17	18	14	19
	19	17	11	18	15	19	16	4	13	19	19	19	15	15	19	19	17	18	19	19	18	19	12	12	15	17	20	19	19	17	19	13	17	18	15	16	16	15	15	17
	18	18	20	20	14	18	20	20	20	18	20	19	20	15	18	20	19	20	20	20	20	18	19	19	20	20	17	18	20	20	20	20	20	20	18	19	20	20		

Table B.1: Dataset 1 used in standard PL analysis.

Rank	Ranker									
	41	...				50				
1	16	5	8	10	20	9	19	8	2	15
2	2	3	11	13	9	10	14	16	20	9
3	18	8	12	15	19	2	5	2	14	3
4	1	12	6	11	8	11	9	9	10	2
5	6	10	1	3	10	20	13	6	4	18
6	4	2	19	4	13	7	8	20	13	20
7	8	4	9	14	2	19	1	11	19	7
8	9	15	18	6	4	16	15	17	7	14
9	19	20	13	9	5	13	12	19	6	1
10	17	7	4	1	15	4	20	4	3	16
11	13	9	20	20	18	18	11	12	15	8
12	15	19	16	18	6	1	17	15	5	17
13	3	1	7	17	3	8	18	18	17	19
14	11	6	3	12	17	12	10	7	12	11
15	10	14	17	5	7	3	4	14	9	4
16	7	11	5	16	12	17	7	10	8	13
17	5	16	15	7	1	15	6	1	1	12
18	14	18	2	2	11	5	2	3	16	6
19	20	17	14	19	14	14	3	5	11	10
20	12	13	10	8	16	6	16	13	18	5

Table B.2: Additional 10 uninformative rankings used to form Dataset 2.

Rank	Ranker																																							
	1	...	10	...	20	...	30	...	40																															
1	13	4	12	16	3	4	2	15	1	1	4	7	1	7	3	12	2	1	7	16	13	2	19	10	1	13	13	1	11	7	6	4	16	10	3	4	7	10	3	3
2	5	14	2	12	2	3	1	2	3	4	1	9	4	1	1	3	16	11	4	11	2	5	3	3	8	14	6	4	9	2	3	6	1	1	8	13	16	3	10	4
3	2	13	3	1	12	2	5	1	5	3	2	2	3	4	9	2	3	2	2	4	10	13	8	9	5	15	4	12	13	1	1	1	6	11	4	2	13	14	16	2
4	8	5	15	9	4	1	15	6	4	6	8	5	16	6	10	1	5	3	1	13	3	7	4	14	3	6	1	7	6	8	15	5	9	4	15	1	11	11	1	14
5	9	7	1	2	1	8	12	3	14	8	10	1	6	12	2	8	13	6	3	3	14	1	12	1	9	8	2	10	4	3	8	15	3	2	6	5	1	12	4	7
6	6	2	7	7	5	10	4	4	7	10	6	11	9	10	4	7	1	10	15	9	1	15	6	12	2	4	3	3	7	6	10	3	10	3	1	8	10	1	11	1
7	1	6	6	4	7	13	3	7	2	2	14	10	2	8	11	4	14	9	8	10	11	4	1	6	12	1	7	5	2	4	2	2	8	16	3	4	16	2	10	
8	7	1	14	14	15	9	9	9	6	5	12	3	12	3	6	9	15	16	12	1	4	11	5	11	6	5	11	14	1	5	5	7	4	6	2	16	15	8	8	13
9	16	3	9	3	6	15	13	19	9	12	3	6	7	2	19	10	4	15	5	2	5	10	2	13	4	3	5	13	5	14	4	8	5	12	5	9	3	2	9	16
10	10	8	16	5	11	11	7	13	10	9	13	8	13	11	13	6	8	7	10	12	6	6	13	4	11	10	12	8	12	9	12	14	8	13	11	10	14	5	13	8
11	11	12	5	6	16	5	16	16	13	7	5	4	15	5	8	16	10	14	6	5	8	3	10	2	13	11	14	2	14	12	9	16	14	5	7	7	5	4	6	6
12	12	11	4	11	14	12	6	11	12	13	9	15	11	9	12	5	6	4	14	6	9	12	14	5	7	2	20	6	8	10	14	11	12	16	12	14	8	7	15	11
13	4	15	8	8	13	16	11	14	11	14	7	13	18	13	14	11	7	13	11	14	16	14	20	16	10	16	16	15	3	11	13	9	11	9	10	6	2	13	19	12
14	15	19	13	10	18	14	14	5	8	15	11	16	5	16	5	18	11	8	17	7	7	9	16	7	17	9	10	11	10	15	7	10	7	15	9	15	6	17	14	5
15	14	10	10	13	8	17	8	12	16	11	17	14	8	15	16	14	20	12	16	8	12	16	15	15	18	12	8	20	16	13	11	13	13	14	13	12	9	6	5	15
16	13	9	11	15	9	6	10	10	15	20	15	12	14	14	20	15	9	5	9	15	15	8	7	19	15	7	15	9	15	16	16	12	15	7	17	11	20	9	12	20
17	18	16	20	19	10	7	18	8	19	17	16	18	10	18	18	13	12	20	19	19	17	20	11	8	16	20	9	16	17	20	19	18	14	18	12	20	7	17		
18	20	17	18	18	17	19	20	20	18	16	19	17	17	20	7	17	18	17	13	18	20	17	9	17	14	19	19	17	20	19	19	18	19	19	17	19	18	9		
19	17	20	19	20	20	20	17	17	17	18	20	20	19	15	19	17	18	20	17	19	19	18	20	19	17	17	19	19	18	18	17	20	20	20	18	15	17	19		
20	19	18	17	17	19	18	19	18	20	19	18	19	19	17	17	20	19	19	18	20	18	18	17	18	18	17	20	18	17	17	18	17	19	18	20	18				

Table B.3: Dataset 3 used in simulation study 1 under WAND.

Rank	Ranker											
	41	...				50						
1	11	7	20	13	17	8	15	19	18	18		
2	1	11	15	12	13	7	17	10	20	6		
3	18	20	16	2	10	1	5	4	1	19		
4	15	19	17	17	6	11	7	3	11	1		
5	12	15	1	19	18	19	6	12	7	16		
6	13	14	14	20	9	6	18	9	2	10		
7	20	12	7	6	15	9	14	7	10	14		
8	10	2	9	10	1	16	9	15	6	20		
9	6	18	13	4	14	14	1	18	15	8		
10	7	3	10	16	20	17	4	8	5	13		
11	19	9	5	18	16	5	13	6	3	2		
12	14	8	19	3	5	13	19	17	4	3		
13	4	5	6	9	19	12	2	13	13	15		
14	16	4	3	8	2	4	12	14	9	11		
15	8	6	2	7	3	20	16	20	19	17		
16	2	10	18	5	8	18	3	5	12	5		
17	17	1	4	14	11	10	10	11	16	7		
18	3	16	12	11	12	2	8	2	14	9		
19	9	17	8	15	4	3	11	16	17	12		
20	5	13	11	1	7	15	20	1	8	4		

Table B.4: Additional 10 uninformative rankings used to form Dataset 4.

Rank	Ranker																								37	38	39	40															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36							
222	1	2	1	13	13	18	1	14	7	9	10	19	1	1	1	13	1	10	7	5	9	1	20	18	1	2	8	9	12	3	16	14	12	5	16	8	7	10	10	7			
	20	1	6	1	6	13	13	4	2	4	1	14	10	11	20	6	10	1	14	15	11	12	14	7	3	8	1	16	8	11	18	12	3	9	13	7	10	19	9	17			
	3	7	15	10	10	1	1	15	10	1	7	15	3	18	13	3	17	20	18	12	1	1	5	10	13	5	1	5	4	2	8	2	13	8	14	2	4	3	18	6	16		
	4	2	3	12	15	15	3	5	1	5	14	12	1	7	3	2	20	13	3	11	10	5	13	1	5	11	7	3	7	16	4	5	16	13	4	5	2	6	2	2	18		
	5	17	18	14	7	16	17	2	15	18	18	14	17	15	7	7	5	9	13	6	12	10	18	13	4	2	4	17	17	20	2	8	5	17	2	20	3	12	7	3	6		
	6	10	19	13	4	9	20	7	6	6	13	6	10	9	18	17	14	2	20	13	9	19	4	18	12	12	9	4	11	3	1	1	2	2	7	1	9	1	17	5	8		
	7	4	20	4	9	7	4	10	7	12	1	17	4	14	14	13	12	6	12	2	7	14	7	3	10	10	5	2	5	5	16	3	7	9	13	4	11	17	9	19	2		
	8	19	7	18	11	11	15	6	13	3	15	13	20	20	6	14	1	15	5	1	14	6	9	15	6	6	3	16	2	1	5	7	11	4	3	18	10	15	4	20	12		
	9	12	14	3	17	4	6	9	17	14	16	9	9	3	9	4	11	7	9	17	2	4	17	6	14	7	14	11	1	6	9	20	4	10	16	11	5	13	3	16	3		
	10	14	10	19	20	17	19	4	20	10	6	7	13	13	15	6	19	12	19	4	3	7	15	4	9	9	12	7	12	10	14	10	8	1	12	3	16	2	13	1	19		
	11	9	12	15	12	20	16	14	9	13	12	4	7	5	12	19	10	18	7	19	6	13	6	7	11	15	16	20	8	11	12	11	9	16	8	10	1	18	8	15	13		
	12	15	4	5	6	3	7	8	12	17	5	19	12	17	4	12	4	19	14	15	20	12	14	17	1	8	11	12	3	13	7	4	15	5	15	19	14	8	6	12	10		
	13	6	13	2	2	10	10	20	19	16	20	3	5	2	2	9	18	11	2	3	13	15	3	12	2	4	17	15	19	4	6	14	3	18	18	14	19	11	5	7	14		
	14	5	6	7	3	12	12	11	3	20	2	18	2	6	5	18	9	4	6	9	11	2	10	2	20	14	15	9	14	14	10	13	6	14	6	8	20	19	14	18	5		
	15	18	11	20	14	14	11	12	2	4	3	2	6	12	10	11	16	5	11	10	4	3	11	9	15	17	13	14	20	9	19	6	1	7	1	9	12	9	15	11	1		
	16	13	5	17	18	2	14	18	5	9	10	20	15	4	19	10	15	14	15	18	17	18	2	5	3	20	19	10	18	7	15	15	18	11	19	12	13	5	20	17	20		
	17	3	9	9	5	18	5	17	11	11	19	11	11	19	20	15	2	17	17	5	18	17	19	11	17	18	18	6	10	17	20	9	10	15	11	7	17	20	16	14	9		
	18	11	17	11	19	19	2	19	16	15	17	5	18	11	17	5	7	3	4	20	19	16	16	19	19	16	10	18	6	19	17	12	19	20	10	17	15	16	12	4	11		
	19	16	16	16	5	9	3	18	19	11	8	16	16	8	16	3	16	8	16	8	8	16	16	20	20	8	8	13	6	19	13	18	18	19	17	6	20	6	18	14	11	13	4
	20	8	8	8	8	8	16	8	8	16	8	8	16	8	8	8	16	8	8	16	20	20	8	8	13	6	19	13	18	18	19	17	6	20	6	18	4	1	8	15			

Table B.5: Dataset 5 used in simulation study 2 under WAND. Vertical lines separate the rankings within each different ranker group.

Ranker	Rank								
	1								9
1	1	5	4	6	2	7	3	9	8
2	1	3	2	6	7	5	4	8	9
3	1	6	4	7	3	2	8	5	9
4	1	6	7	3	2	5	4	8	9
5	2	7	4	3	9	5	1	6	8
6	2	3	5	8	4	1	6	7	9
7	2	1	5	3	4	7	8	6	9
8	2	3	5	1	6	8	9	4	7
9	2	8	3	1	4	6	5	7	9
10	2	3	1	4	9	5	8	6	7
11	2	7	4	1	8	6	5	3	9
12	3	2	1	8	4	5	7	6	9
13	3	1	6	7	8	4	9	5	2
14	3	1	5	4	7	8	2	9	6
15	3	2	4	8	5	9	1	6	7
16	4	8	5	7	1	9	2	3	6
17	4	5	8	9	1	7	6	3	2
18	4	8	5	9	3	2	6	7	1
19	5	4	7	8	9	2	3	6	1
20	5	4	2	7	8	9	3	1	6
21	5	2	9	8	4	7	1	3	6
22	5	7	4	9	8	3	2	1	6
23	5	4	9	7	3	2	8	1	6
24	5	4	9	8	3	7	2	1	6
25	5	4	2	8	9	3	1	7	6
26	5	7	8	1	4	9	2	6	3
27	5	4	9	7	1	2	8	3	6
28	1	4	5	8	9	7	6	2	3
29	6	1	8	4	2	3	7	5	9
30	6	1	4	3	2	5	8	7	9
31	6	3	2	5	1	7	8	9	4
32	7	4	8	1	2	3	6	5	9
33	7	5	4	8	1	9	2	6	3
34	7	4	2	8	6	1	3	5	9
35	7	5	8	4	2	9	3	1	6
36	7	1	5	4	3	2	6	8	9
37	7	4	5	8	1	9	6	2	3
38	9	5	3	4	7	8	2	1	6
39	9	4	8	5	1	2	3	7	6

Entity No	1	2	3	4	5	6	7	8	9
	SOC	EDU	CLI	MAT	EXP	CUL	IND	TST	PHY
Area									

Table B.6: Roskam's psychology data

Entity No	Team	Professionals			Avid Fans			Fans			Infrequent Watchers			Not-interested Individuals			34	
		1	...	6	7	...	12	13	...	18	19	...	25	26	...	34		
1	Heat	1	9	1	1	1	1	4	6	1	6	6	1	4	1	6	1	10
2	Thunder	9	1	2	9	2	10	2	2	1	2	4	2	4	6	1	2	1
3	Spurs	2	2	9	2	10	2	6	11	2	1	6	1	10	4	2	4	1
4	Celtics	10	10	10	10	9	9	4	10	11	4	10	10	26	8	2	3	4
5	Clippers	4	5	13	3	5	4	10	3	3	3	11	4	18	26	3	15	21
6	Lakers	6	11	5	6	6	5	3	5	6	10	22	3	14	3	14	26	26
7	Pacers	3	6	6	12	11	3	15	6	9	9	27	5	24	18	6	8	14
8	76ers	5	12	12	11	3	6	7	4	10	5	13	9	16	20	9	3	20
9	Mavericks	11	13	11	4	4	13	-	-	-	-	-	-	-	-	-	-	-
10	Bulls	12	3	4	5	13	12	-	-	-	-	-	-	-	-	-	-	-
11	Knicks	14	4	3	14	12	14	-	-	-	-	-	-	-	-	-	-	-
12	Grizzlies	15	14	15	17	15	7	-	-	-	-	-	-	-	-	-	-	-
13	Nuggets	17	7	8	13	7	11	-	-	-	-	-	-	-	-	-	-	-
14	Magic	7	17	7	7	14	17	-	-	-	-	-	-	-	-	-	-	-
15	Hawks	8	18	17	8	8	8	-	-	-	-	-	-	-	-	-	-	-
16	Jazz	19	8	18	18	17	19	-	-	-	-	-	-	-	-	-	-	-
17	TrailBlazers	21	19	14	19	18	18	-	-	-	-	-	-	-	-	-	-	-
18	Rockets	16	15	24	15	24	15	-	-	-	-	-	-	-	-	-	-	-
19	Bucks	13	21	20	22	20	16	-	-	-	-	-	-	-	-	-	-	-
20	Suns	20	23	19	21	19	22	-	-	-	-	-	-	-	-	-	-	-
21	Nets	18	22	26	20	26	20	-	-	-	-	-	-	-	-	-	-	-
22	Warriors	22	20	23	23	22	25	-	-	-	-	-	-	-	-	-	-	-
23	Timberwolves	23	16	22	24	23	21	-	-	-	-	-	-	-	-	-	-	-
24	Hornets	28	26	21	25	21	23	-	-	-	-	-	-	-	-	-	-	-
25	Pistons	25	25	25	27	25	24	-	-	-	-	-	-	-	-	-	-	-
26	Kings	29	28	16	26	29	26	-	-	-	-	-	-	-	-	-	-	-
27	Wizards	24	27	29	16	27	27	-	-	-	-	-	-	-	-	-	-	-
28	Raptors	27	24	27	28	16	28	-	-	-	-	-	-	-	-	-	-	-
29	Cavaliers	26	29	28	29	30	29	-	-	-	-	-	-	-	-	-	-	-
30	Bobcats	30	30	30	30	28	30	-	-	-	-	-	-	-	-	-	-	-

Table B.7: NBA data

Rank	Ranker																																							
	1	...	10	...	20	...	30	...	40																															
1	7	10	5	9	2	3	1	6	10	4	7	9	1	4	3	5	7	1	9	9	1	4	8	3	7	10	9	10	8	1	2	10	1	7	9	9	4	9	6	10
2	6	8	2	8	5	8	8	2	8	2	6	5	8	2	5	8	6	8	2	2	8	6	4	2	5	8	2	2	2	1	8	8	5	2	8	2	2	7	8	
3	10	1	3	7	9	10	10	9	3	10	5	4	5	10	9	9	1	9	10	1	9	7	3	9	10	7	3	1	1	7	3	9	5	10	5	6	9	3	3	1
4	3	6	1	3	8	4	2	1	5	6	4	1	9	1	2	4	4	4	7	5	5	9	7	1	1	6	6	7	7	9	8	3	4	3	3	3	5	5	1	5
5	1	5	6	2	1	1	6	10	6	7	1	6	6	8	6	1	8	6	8	3	3	5	10	4	2	3	8	4	4	4	6	1	2	4	6	5	10	8	8	4
6	5	9	8	6	4	7	9	4	4	5	2	8	7	5	4	10	2	2	5	6	4	3	6	10	3	2	10	5	3	5	10	4	9	8	7	10	7	4	5	2
7	4	7	10	1	10	9	7	5	1	8	8	10	3	7	8	6	9	10	3	4	7	1	5	5	6	4	4	3	10	8	7	7	10	1	1	7	3	7	9	9
8	2	2	4	5	6	2	5	8	2	1	9	2	2	6	7	2	3	7	4	8	2	2	8	8	5	7	8	9	6	4	2	6	2	8	2	8	6	4	6	
9	8	3	7	10	7	5	4	3	9	9	3	3	10	9	10	7	5	3	1	10	6	8	1	6	4	1	1	9	5	10	9	6	3	9	10	4	1	1	10	3
10	9	4	9	4	3	6	3	7	7	3	10	7	4	3	1	3	10	5	6	7	10	10	9	7	9	9	5	6	6	3	5	5	7	6	4	1	6	10	2	7

Rank	Ranker																																							
	41	...	50	...	60	...	70	...	80																															
1	7	6	1	9	1	4	7	10	5	7	1	3	1	9	7	2	1	3	4	9	7	3	4	9	9	4	6	3	7	9	10	9	10	8	3	10	9	2	1	7
2	2	2	5	8	6	2	6	8	2	2	8	2	2	8	8	2	2	8	10	8	6	6	7	2	6	2	2	4	8	4	2	8	2	2	6	1	6	6	6	
3	5	3	6	10	5	10	3	3	9	3	3	7	9	5	3	4	10	7	9	5	4	10	10	10	7	1	8	9	1	1	9	7	3	10	5	7	7	4	10	3
4	3	5	3	4	4	3	4	9	7	1	9	10	8	4	2	9	3	8	7	3	6	7	1	4	8	9	1	5	3	7	1	4	9	6	4	5	2	1	8	10
5	1	8	8	7	2	5	1	6	8	10	6	4	5	3	5	5	1	10	2	9	9	9	3	4	7	10	7	2	5	6	5	2	9	8	2	10	5	5	2	
6	6	4	7	6	7	7	9	2	4	4	7	9	7	6	6	7	8	5	5	8	2	4	8	8	6	5	9	4	5	4	3	6	7	5	10	1	3	7	7	8
7	4	9	4	3	9	6	2	5	1	5	5	5	3	7	9	3	7	10	6	1	3	1	2	2	1	8	3	1	8	3	8	1	4	7	7	4	8	3	2	4
8	8	1	2	2	8	8	5	7	6	6	2	6	6	10	10	6	6	6	2	4	5	2	7	6	5	2	4	8	9	6	5	8	6	4	6	8	6	8	4	5
9	9	7	10	1	3	1	10	1	3	8	4	8	4	8	1	10	4	4	3	7	10	5	3	5	10	3	5	6	10	10	2	3	5	1	9	3	4	10	3	9
10	10	10	9	5	10	9	8	4	10	9	10	1	10	1	4	1	9	9	1	6	1	8	5	1	3	10	7	10	6	2	7	10	1	3	1	9	5	9	9	1

Table B.8: Dataset 6 used in the simulation studies for the Bayesian analysis of the Extended Plackett–Luce model.

Rank	Ranker																							
	81					...					90					...					100			
1	9	6	9	9	10	4	9	3	7	10	10	5	9	5	7	3	5	10	7	8				
2	8	8	8	2	2	2	2	2	2	5	8	6	2	8	9	2	2	2	2	10				
3	7	5	1	4	9	6	1	6	4	8	9	1	7	7	1	10	4	5	8	9				
4	10	1	5	6	5	3	5	4	6	4	6	9	6	4	3	5	1	4	4	6				
5	5	3	2	3	8	7	6	9	3	2	1	8	10	10	8	8	8	7	1	7				
6	2	10	3	1	4	8	7	1	1	9	4	7	1	6	2	7	7	8	6	4				
7	3	2	10	5	1	10	3	10	5	1	2	3	5	9	10	6	9	9	10	2				
8	1	4	7	7	7	5	8	8	8	7	3	2	8	2	5	4	6	6	5	5				
9	4	7	4	8	6	9	4	7	9	6	5	4	4	3	4	9	10	3	9	1				
10	6	9	6	10	3	1	10	5	10	3	7	10	3	1	6	1	3	1	3	3				

Table B.9: Additional 20 rankings used to form Dataset 6

Bibliography

- Aldous, D. J. (1985), Exchangeability and related topics, in ‘École d’Été de Probabilités de Saint-Flour XIII 1983’, Springer, pp. 1–198.
- Altekar, G., Dwarkadas, S., Huelsenbeck, J. P. and Ronquist, F. (2004), ‘Parallel Metropolis coupled Markov chain Monte Carlo for Bayesian phylogenetic inference’, *Bioinformatics* **20**(3), 407–415.
- Antoniak, C. E. (1974), ‘Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems’, *The Annals of Statistics* **2**, 1152–1174.
- Atchadé, Y. F., Roberts, G. O. and Rosenthal, J. S. (2011), ‘Towards optimal scaling of Metropolis-coupled Markov chain Monte Carlo’, *Statistics and Computing* **21**(4), 555–568.
- Baker, R. D. and McHale, I. G. (2015), ‘Deterministic evolution of strength in multiple comparisons models: Who is the greatest golfer?’, *Scandinavian Journal of Statistics* **42**(1), 180–196.
- Benter, W. (1994), Computer based horse race handicapping and wagering systems: A report, in ‘Efficiency of racetrack betting markets’, Academic Press, pp. 183–198.
- Bernardo, J. M. and Smith, A. F. M. (1994), *Bayesian Theory*, Wiley, Chichester, U.K.
- Bezáková, I., Kalai, A. and Santhanam, R. (2006), Graph model selection using maximum likelihood, in ‘Proceedings of the 23rd international conference on Machine learning’, ACM, pp. 105–112.
- Blackwell, D. and MacQueen, J. B. (1973), ‘Ferguson distributions via Pólya urn schemes’, *The Annals of Statistics* **1**, 353–355.
- Bradley, R. A. and Terry, M. E. (1952), ‘Rank analysis of incomplete block designs: I. The method of paired comparisons’, *Biometrika* **39**(3–4), 324–345.
- Breslow, N., Crowley, J. et al. (1974), ‘A large sample study of the life table and product limit estimates under random censorship’, *The Annals of Statistics* **2**(3), 437–453.

- Brooks, S. (1998), ‘Markov chain Monte Carlo method and its application’, *Journal of the Royal Statistical Society: Series D (the Statistician)* **47**(1), 69–100.
- Caron, F. and Doucet, A. (2012), ‘Efficient Bayesian inference for generalized Bradley–Terry models’, *Journal of Computational and Graphical Statistics* **21**(1), 174–196.
- Caron, F., Teh, Y. W. and Murphy, T. B. (2014), ‘Bayesian nonparametric Plackett–Luce models for the analysis of preferences for college degree programmes’, *The Annals of Applied Statistics* **8**(2), 1145–1181.
- Casella, G. and Robert, C. P. (1996), ‘Rao-Blackwellisation of sampling schemes’, *Biometrika* **83**(1), 81–94.
- Chib, S. and Greenberg, E. (1995), ‘Understanding the Metropolis-Hastings algorithm’, *The American Statistician* **49**(4), 327–335.
- Choulakian, V. (2016), ‘Globally homogenous mixture components and local heterogeneity of rank data’, *arXiv preprint arXiv:1608.05058*.
- Dahl, D. B. (2006), Model-based clustering for expression data via a Dirichlet process mixture model, in K.-A. Do, P. Müller and M. Vannucci, eds, ‘Bayesian Inference for Gene Expression and Proteomics’, Cambridge University Press, pp. 201–218.
- de Leeuw, J. (2006), Nonlinear principal component analysis, in M. Greenacre and J. Blasius, eds, ‘Multiple correspondence analysis and related methods’, CRC Press, chapter 4, pp. 107–134.
- de Leeuw, J. and Mair, P. (2009), ‘Gifi methods for optimal scaling in R: The package homals’, *Journal of Statistical Software* **31**(4), 1–20.
URL: <http://www.jstatsoft.org/v31/i04/>
- Deng, K., Han, S., Li, K. J. and Liu, J. S. (2014), ‘Bayesian aggregation of order-based rank data’, *Journal of the American Statistical Association* **109**(507), 1023–1039.
- Diaconis, P. (1988), ‘Group representations in probability and statistics’, *Lecture Notes-Monograph Series* **11**, 1–192.
- Duane, S., Kennedy, A. D., Pendleton, B. J. and Roweth, D. (1987), ‘Hybrid Monte Carlo’, *Physics Letters B* **195**(2), 216–222.
- Escobar, M. D. and West, M. (1995), ‘Bayesian density estimation and inference using mixtures’, *Journal of the American Statistical Association* **90**(430), 577–588.
- Everitt, B. and Hand, D. (1981), ‘Finite mixture distributions’, *Monographs on Applied Probability and Statistics, London: Chapman and Hall, 1981*.

- Everitt, B. S., Landau, S., Leese, M. and Stahl, D. (2011), ‘Hierarchical clustering’, *Cluster Analysis, 5th Edition* pp. 71–110.
- Ferguson, T. S. (1973), ‘A Bayesian analysis of some nonparametric problems’, *The Annals of Statistics* **1**, 209–230.
- Gamerman, D. and Lopes, H. F. (2006), *Markov chain Monte Carlo: stochastic simulation for Bayesian inference*, Chapman and Hall/CRC.
- Gelfand, A. E. and Smith, A. F. (1990), ‘Sampling-based approaches to calculating marginal densities’, *Journal of the American Statistical Association* **85**(410), 398–409.
- Gelman, A. (1996), Inference and monitoring convergence, in W. Gilks, S. Richardson and D. Spiegelhalter, eds, ‘Markov Chain Monte Carlo in Practice’, Chapman & Hall/CRC Interdisciplinary Statistics, Taylor & Francis.
- Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A. and Rubin, D. B. (2014), *Bayesian Data Analysis*, 3rd edn, CRC press Boca Raton, FL.
- Gelman, A., Roberts, G. O., Gilks, W. R. et al. (1996), Efficient Metropolis jumping rules, in J. M. Bernardo, J. Berger, A. P. Dawid and J. F. M. Smith, eds, ‘Bayesian Statistics 5’, Oxford University Press, pp. 599–608.
- Geman, S. and Geman, D. (1984), ‘Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images’, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **6**, 721–741.
- Geweke, J. (1992), Evaluating the accuracy of sampling-based approaches to calculating posterior moments, in J. M. Bernardo, J. Berger, A. P. Dawid and J. F. M. Smith, eds, ‘Bayesian Statistics 4’, Oxford University Press, pp. 169–193.
- Geyer, C. J. (1991), Markov chain Monte Carlo maximum likelihood, in ‘Proceedings of the 23rd Symposium on the Interface’, Computing Science and Statistics, Interface Foundation of North America, pp. 156–163.
- Geyer, C. J. and Thompson, E. A. (1995), ‘Annealing Markov chain Monte Carlo with applications to ancestral inference’, *Journal of the American Statistical Association* **90**(431), 909–920.
- Gilks, W. R. and Roberts, G. O. (1996), Strategies for improving MCMC, in W. Gilks, S. Richardson and D. Spiegelhalter, eds, ‘Markov chain Monte Carlo in Practice’, Chapman & Hall/CRC Interdisciplinary Statistics, Taylor & Francis.

- Glickman, M. E. and Hennessy, J. (2015), ‘A stochastic rank ordered logit model for rating multi-competitor games and sports’, *Journal of Quantitative Analysis in Sports* **11**(3), 131–144.
- Gormley, I. C. and Murphy, T. B. (2006), ‘Analysis of Irish third-level college applications data’, *Journal of the Royal Statistical Society: Series A (Statistics in Society)* **169**(2), 361–379.
- Gormley, I. C. and Murphy, T. B. (2008a), ‘Exploring voting blocs within the Irish electorate: a mixture modeling approach’, *Journal of the American Statistical Association* **103**(483), 1014–1027.
- Gormley, I. C. and Murphy, T. B. (2008b), ‘A mixture of experts model for rank data with applications in election studies’, *Annals of Applied Statistics* **2**(4), 1452–1477.
- Gormley, I. C. and Murphy, T. B. (2009), ‘A grade of membership model for rank data’, *Bayesian Analysis* **4**(2), 265–295.
- Graves, T., Reese, C. S. and Fitzgerald, M. (2003), ‘Hierarchical models for permutations: Analysis of auto racing results’, *Journal of the American Statistical Association* **98**(462), 282–291.
- Hastie, D. I., Liverani, S. and Richardson, S. (2015), ‘Sampling from Dirichlet process mixture models with unknown concentration parameter: mixing issues in large data implementations’, *Statistics and Computing* **25**(5), 1023–1037.
- Hastings, W. K. (1970), ‘Monte Carlo sampling methods using Markov chains and their Applications’, *Biometrika* **57**(1), 97–109.
- Henderson, D. A. and Kirrane, L. J. (2018), ‘A comparison of truncated and time-weighted Plackett–Luce models for probabilistic forecasting of Formula One results’, *Bayesian Analysis* **13**(2), 335–358.
- Henery, R. (1981), ‘Permutation probabilities as models for horse races’, *Journal of the Royal Statistical Society. Series B (Statistical Methodology)* pp. 86–91.
- Hjort, N. L., Holmes, C. C., Müller, P. and Walker, S. G., eds (2010), *Bayesian Nonparametrics*, Cambridge University Press, Cambridge UK.
- Hunter, D. R. (2004), ‘MM algorithms for generalized Bradley-Terry models’, *The Annals of Statistics* **32**(1), 384–406.
- Ishwaran, H. and James, L. F. (2001), ‘Gibbs sampling methods for stick-breaking priors’, *Journal of the American Statistical Association* **96**(453), 161–173.

- Ishwaran, H. and Zarepour, M. (2002), ‘Exact and approximate sum representations for the Dirichlet process’, *Canadian Journal of Statistics* **30**(2), 269–283.
- Lau, J. W. and Green, P. J. (2007), ‘Bayesian model-based clustering procedures’, *Journal of Computational and Graphical Statistics* **16**(3), 526–558.
- Lindsay, B. G. (1995), Mixture models: theory, geometry and applications, in ‘NSF-CBMS regional conference series in probability and statistics’, JSTOR, pp. 1–163.
- Luce, R. (1959), *Individual Choice Behavior: A Theoretical Analysis*, Wiley.
- MacEachern, S. N. and Müller, P. (1998), ‘Estimating mixture of Dirichlet process models’, *Journal of Computational and Graphical Statistics* **7**(2), 223–238.
- Mallows, C. L. (1957), ‘Non-null ranking models. I’, *Biometrika* **44**(1–2), 114–130.
- Marden, J. I. (1995), *Analyzing and Modeling Rank Data*, Chapman and Hall, London.
- Medvedovic, M. and Sivaganesan, S. (2002), ‘Bayesian infinite mixture model based clustering of gene expression profiles’, *Bioinformatics* **18**(9), 1194–1206.
- Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. and Teller, E. (1953), ‘Equation of state calculations by fast computing machines’, *The Journal of Chemical Physics* **21**(6), 1087–1092.
- Mollica, C. and Tardella, L. (2014), ‘Epitope profiling via mixture modelling of ranked data’, *Statistics in Medicine* **33**(21), 3738–3758.
- Mollica, C. and Tardella, L. (2016), ‘Bayesian Plackett–Luce mixture models for partially ranked data’, *Psychometrika* **82**(2), 1–17.
- Mollica, C. and Tardella, L. (2018), ‘Algorithms and diagnostics for the analysis of preference rankings with the Extended Plackett–Luce model’, *arXiv preprint arXiv:1803.02881*.
- Neal, R. M. (2000), ‘Markov chain sampling methods for Dirichlet process mixture models’, *Journal of Computational and Graphical Statistics* **9**(2), 249–265.
- Neal, R. M. (2011), MCMC using Hamiltonian dynamics, in S. Brooks, A. Gelman, G. Jones and X.-L. Meng, eds, ‘Handbook of Markov Chain Monte Carlo’, CRC Press, pp. 93–162.
- Papaspiliopoulos, O. and Roberts, G. O. (2008), ‘Retrospective Markov chain Monte Carlo methods for Dirichlet process hierarchical models’, *Biometrika* **95**(1), 169–186.

- Pitman, J. and Yor, M. (1997), ‘The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator’, *The Annals of Probability* **25**(2), 855–900.
- Plackett, R. L. (1975), ‘The analysis of permutations’, *Applied Statistics* **24**, 193–202.
- Plummer, M., Best, N., Cowles, K. and Vines, K. (2006), ‘CODA: Convergence Diagnosis and Output Analysis for MCMC’, *R News* **6**(1), 7–11.
URL: <https://journal.r-project.org/archive/>
- Raftery, A. E. and Lewis, S. (1992), How many iterations in the Gibbs sampler?, in J. M. Bernardo, J. Berger, A. P. Dawid and J. F. M. Smith, eds, ‘Bayesian Statistics 4’, Oxford University Press, pp. 763–773.
- Raftery, A. E. and Lewis, S. (1996), Implementing MCMC, in ‘Markov Chain Monte Carlo in Practice’, W. R. Gilks, S. Richardson and D. J. Spiegelhalter, eds, Chapman & Hall, London, pp. 115–130.
- Rastelli, R. and Friel, N. (2017), ‘Optimal Bayesian estimators for latent variable cluster models’, *Statistics and Computing* pp. 1–18.
- Richardson, S. and Green, P. J. (1997), ‘On Bayesian analysis of mixtures with an unknown number of components (with discussion)’, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **59**(4), 731–792.
- Roberts, G. O. and Rosenthal (2001), ‘Optimal scaling for various Metropolis-Hastings algorithms’, *Statistical Science* **16**(4), 351–367.
- Rodriguez, A. (2007), Some advances in Bayesian nonparametric modeling, PhD thesis, Duke University.
- Rodriguez, A., Dunson, D. B. and Gelfand, A. E. (2008), ‘The Nested Dirichlet process’, *Journal of the American Statistical Association* **103**(483), 1131–1154.
- Sethuraman, J. (1994), ‘A constructive definition of Dirichlet priors’, *Statistica Sinica* **4**, 639–650.
- Sherlock, C. (2013), ‘Optimal scaling of the random walk Metropolis: general criteria for the 0.234 acceptance rule’, *Journal of Applied Probability* **50**(1), 1–15.
- Sherlock, C. and Roberts, G. (2009), ‘Optimal scaling of the random walk Metropolis on elliptically symmetric unimodal targets’, *Bernoulli* **15**(3), 774–798.
- Stephens, M. (2000), ‘Dealing with label switching in mixture models’, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **62**(4), 795–809.

- Stern, H. (1990), ‘Models for distributions on permutations’, *Journal of the American Statistical Association* **85**(410), 558–564.
- Tanner, M. A. and Wong, W. H. (1987), ‘The calculation of posterior distributions by data augmentation’, *Journal of the American Statistical Association* **82**(398), 528–540.
- Teh, Y. W., Jordan, M. I., Beal, M. J. and Blei, D. M. (2006), ‘Hierarchical Dirichlet processes’, *Journal of the American Statistical Association* **101**(476), 1566–1581.
- Vigneau, E., Courcoux, P. and Semenou, M. (1999), ‘Analysis of ranked preference data using latent class models’, *Food Quality and Preference* **10**(3), 201–207.
- Vitelli, V., Sørensen, Ø., Crispino, M., Frigessi, A. and Arjas, E. (2018), ‘Probabilistic preference learning with the Mallows rank model’, *Journal of Machine Learning Research* **18**(158), 1–49.
- Walker, S. G. (2007), ‘Sampling the Dirichlet mixture model with slices’, *Communications in Statistics – Simulation and Computation* **36**(1), 45–54.
- West, M. (1992), *Hyperparameter estimation in Dirichlet process mixture models*, Duke University ISDS Discussion Paper# 92-A03.
- Wilkinson, D. (2013), ‘Parallel tempering and Metropolis coupled MCMC’, <https://darrenjw.wordpress.com/tag/mc3/>. [Online; accessed 4-May-2018].
- Yu, P. L., Lam, K. and Lo, S. (2005), ‘Factor analysis for ranked data with application to a job selection attitude survey’, *Journal of the Royal Statistical Society: Series A (Statistics in Society)* **168**(3), 583–597.