From	**Revision Checklist**	**Reference**
Sir Bong	✓ Right at the start, show a motivating example on the election polls. Explain what quantities will be ranked (i.e., proportion of voters in favor of certain candidates).	See paragraphs 2 to 4 of Section 1.2
	✓ – Mention that the decision on whether a voter votes for a candidate or not is a Bernoulli random variable.	See paragraph 2 of Section 1.2
	✓ – Show the results of Pulse Asia surveys.	See added table in Section 1.2
	✓ – Emphasize that there is now dependence among the estimates.	See paragraphs 3 & 4 of Section 1.2
	✓ – Mention that the estimates may be computed via MLE, least squares, or some other estimation procedure.	See last sentences of paragraph 2 in Section 1.2 and paragraph 1 in Section 2.4
	✓ Add Section 2.4 for discussion on asymptotic variance. Describe the setup assumed by Chen and Dudiewicz. Use notations consistent with the rest of the thesis.	See Section 2.4
	✓ Flesh out Goldstein and Spiegelhalter on use of latent variables.	See Section 2.3.3
	✓ Address the revisions in the PDF file sent by Sir Bong: Add more details to Section 3.4.	See Algorithm 4 in Section 3.4
Ma'am Rose Anne	✓ In the objectives, state explicitly that the proposed methodology will be compared with the Bonferroni approach.	See third bullet in Section 1.1
	✓ In page 4, before Sec. 1.1., explain how your methodology differs from existing approaches.	See paragraph 3 of Section 1
	✓ Explain L_k and U_k (or state that they are real numbers).	See (2.3)
	✓ Page 8, remove the reference to Mogstad et al. (2024).	done
	✓ In Sec. 2.3.1, make a reference to the Pulse Asia example.	See paragraphs 2 to 4 of Section 1.2 and Section 2.3.1
	✓ More details from Bazylik et al. (2025).	See paragraph 3 of Section 2.3.4

Thesis proposal revision checklist.

Joint Confidence Regions for Rankings Based on Correlated Estimates

Shaine Rosewel Paralis Matala

A thesis proposal submitted to UP Diliman School of Statistics

In Partial Fulfillment of the Requirements for the Degree of
Master of Science in Statistics

School of Statistics
University of the Philippines
December 2025

Contents

1 Introduction

Ranked data are commonly of interest because they allow readers to compare populations based on ranked quantities. For example, top universities across the globe may be identified based on their institutional performance indicator; states may receive appropriate intervention according to their relative rank based on average travel times to work; and senatorial candidates who are likely to win seats can be predicted by public opinion polls prior to elections. Whenever data on the entire population are unavailable, the ranks are based on estimates rather than the unknown parameters. Consequently, their overall uncertainty—expressed through joint confidence intervals—should also be quantified. Marginally, these intervals provide information on the possible range of each rank while jointly, they facilitate comparing all ranks simultaneously rather than reporting them in isolation.

Several studies have addressed this concern through different techniques. Some approaches like those by Klein et al. (2020), Mohamad et al. (2019), Mogstad et al. (2024), Andersson et al. (1998) and Lyhagen & Ahlgren (2020), relied solely on the estimates and their standard errors, constructing joint confidence intervals either for the estimated quantities or directly for the ranks themselves. Others fit models in which institutional differences are modeled explicitly via latent effects, thereby accounting for the dependence among student observations induced by the multilevel structure (Spiegelhalter, 1996). Miller (2009) developed a bootstrap algorithm that allows for the assumption of independence in the estimators despite its potential violation while Bazylik et al. (2025) proposed a bootstrap procedure that explicitly accounts for the dependence in estimators that are multinomially distributed.

Assuming independence when constructing joint confidence regions for estimators that are, in fact, correlated may lead to overly conservative and thus wider intervals, implying greater uncertainty—contrary to what is desired. The more general setting, which to our knowledge has not yet been explored, allows for a certain degree of correlation among the estimates to be ranked. This thesis develops a procedure capable of handling such dependencies while maintaining coverage close to the nominal level and producing relatively narrow joint confidence intervals. The proposed methodology uses only the observed estimators and their corresponding standard errors. Although it also employs a parametric bootstrap—commonly used to estimate overall rank uncertainties (e.g., Andersson et al. (1998), Mogstad et al. (2024), Bazylik et al. (2025))—our implementation differs from these existing approaches — Mogstad et al. (2024) and Bazylik et al. (2025) obtained bounds for the simultaneous confidence sets which depend on quantiles from the distribution of the maximum (over k) of the differences (See Section 2.3.2, instead of just the estimators while Andersson et al. (1998) implemented it to obtain a calculated measure as discussed in Section 2.3.1.

1.1 Objective

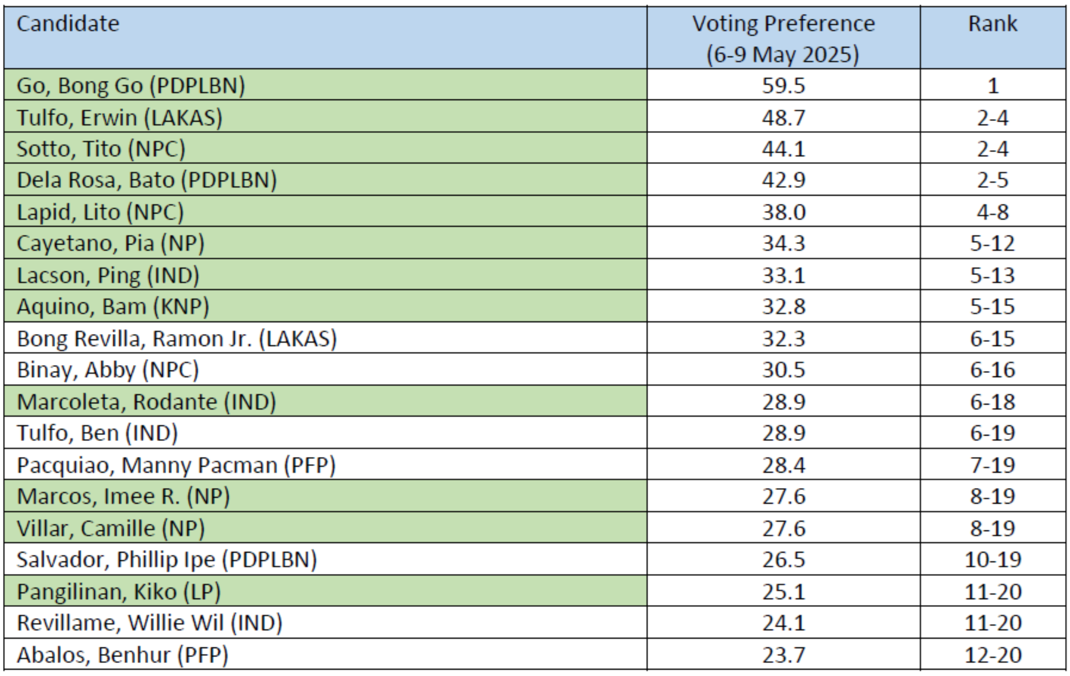
This research builds upon Klein et al. (2020)'s methodology by extending the set of joint confidence intervals used to capture uncertainty in overall rankings. In particular, it intends to:

- Develop a procedure to construct joint confidence intervals for the ranks and the ranked parameters when the estimates to be ranked may be correlated.
- Evaluate the performance of the proposed approaches under various parameter settings.
- Compare the results of Algorithm 1 to Klein's joint confidence regions that uses Bonferroni and independence assumption, in terms of coverage and tightness measures.
- Apply the proposed approaches to a real-life example.

1.2 Significance

In order to obtain joint confidence sets for overall ranks, Klein et al. (2020) estimates confidence intervals for the unknown parameters, with a joint coverage probability of at least $1 - \alpha$. Their goal is to produce confidence intervals that collectively produce a small difference between the upper and the lower bound to yield tighter joint uncertainty for ranks. In the same study, they used individual confidence intervals of the form $\hat{\theta}_k \pm z_{\alpha/2} \times \sigma_k$ for the population parameters $\theta_1, \theta_2, \ldots, \theta_K$, assuming that the estimates of these parameters $\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_K$ are independently distributed. This approach, while simple, is no longer applicable to the case where $\hat{\theta}_k$s may be correlated. In many cases, assuming independence in the presence of correlated quantities leads to conservative confidence intervals resulting in wider intervals which imply a higher uncertainty in overall ranks.

In the Philippines, senators are elected using Multiple Non-transferable Vote system (MNTV)—where candidates are voted for individually regardless of partisan membership and alliances (Ravanilla & Hicken, 2023). This applies to pre-election polls where each voter X_i selects at most 12 from a pool of K candidates. We let the data $\mathcal{X}$ be a set of n vectors of length K, $\mathcal{X} = \{X_1, X_2, \ldots, X_n\}$, where $X_i = (X_{i1}, X_{i2}, \ldots, X_{iK})$ and $X_{ik} \sim$ Bernoulli(p_k) where p_k is the probability that voter i selects candidate k for $k \in \{1, 2, \ldots, K\}$ and $i \in \{1, 2, \ldots, n\}$. $X_{ik} = 1$ if voter i selects candidate k and 0 otherwise. Public opinion polling bodies report their results by ranking the proportion of votes $\hat{\theta}_k$ garnered by the K candidates from the highest to the lowest, stating individual uncertainty to emphasize who can potentially make it to the top 12. Figure 1 is an example of results from Pulse Asia Research, Inc. (2025); this shows that 19 out of the 64 senatorial candidates have statistical chances of winning a seat as of a week prior to the previous 2025 elections. The shaded rows correspond to the ultimate winners. Pulse Asia Inc. notes their surveys are conducted with a 95% confidence level, with confidence intervals calculated using 2.8% margin of error for a sample size of 1,200. To obtain the estimates, each response is weighted to compensate for the unequal selection probabilities in the sample design.



Candidate	Voting Preference (6-9 May 2025)	Rank
Go, Bong Go (PDPLBN)	59.5	1
Tulfo, Erwin (LAKAS)	48.7	2-4
Sotto, Tito (NPC)	44.1	2-4
Dela Rosa, Bato (PDPLBN)	42.9	2-5
Lapid, Lito (NPC)	38.0	4-8
Cayetano, Pia (NP)	34.3	5-12
Lacson, Ping (IND)	33.1	5-13
Aquino, Bam (KNP)	32.8	5-15
Bong Revilla, Ramon Jr. (LAKAS)	32.3	6-15
Binay, Abby (NPC)	30.5	6-16
Marcoleta, Rodante (IND)	28.9	6-18
Tulfo, Ben (IND)	28.9	6-19
Pacquiao, Manny Pacman (PFP)	28.4	7-19
Marcos, Imee R. (NP)	27.6	8-19
Villar, Camille (NP)	27.6	8-19
Salvador, Phillip Ipe (PDPLBN)	26.5	10-19
Pangilinan, Kiko (LP)	25.1	11-20
Revillame, Willie Wil (IND)	24.1	11-20
Abalos, Benhur (PFP)	23.7	12-20

Figure 1: Candidates in Contention in May 2025 Pulso ng Bayan Survey by Pulse Asia Research Inc.

In weak-party systems, candidates who belong to the same political alliance or ticket commonly co-occur in ballots and hence perform with similarity (David & Legara, 2015). As shown in Figure 1, the competitive list is dominated by ten out of 11 (all except Tolentino) candidates under *Alyansa para sa Bagong Pilipinas*, which was actively campaigned by the incumbent president at the time. Four candidates (Go, Dela Rosa, Salvador, and Marcoleta) are associated with the Duterte-backed alliance *DuterTen*, while two (Pangilinan and Aquino) belong to the opposition-backed *KiBam*. The remaining candidates—Marcos, Revillame, and B. Tulfo—do not formally belong to any alliance but benefit from strong name recall. This alliance-based pattern is consistent with dependence in individual voting behavior.

Because each voter is allowed to vote for up to 12 candidates, votes cast by the same individual are not independent. This dependence is reflected at the aggregate level through the proportions of total votes received by candidates, resulting in correlated proportion estimates. Accounting for these dependencies can lead to rank estimates with reduced overall uncertainty and, consequently, shorter simultaneous confidence intervals. In particular, the range of plausible ranks for each candidate may be narrowed, depending on the degree of correlation among candidates' vote shares.

1.3 Scope and Limitations

This study proposes methodologies to construct joint confidence regions to quantify uncertainty in overall ranks, building upon the main results of Klein et al. (2020). Unlike the study of Klein et al. (2020), however, this thesis addresses the case where the parameter estimates may be correlated. Moreover, this study also proposes a methodologies to construct joint confidence regions for the ordered parameters. It applies the parametric bootstrap in implementing the procedures, with a strong emphasis on maintaining the desired coverage. However, several limitations must be acknowledged. First, the framework assumes that the data are generated from a multivariate normal distribution. Second, a limited set of correlation structures is examined to illustrate how different dependence assumptions may affect the resulting joint confidence sets. That it, we do not discuss how to estimate the correlation matrix from the data.

2 Related Literature

2.1 Ranking Problem

In the problem of estimating ranks of several unknown real-valued parameters $\theta_1, \theta_2, \ldots, \theta_K$, it is desired to rank these K parameters from smallest to largest, $\theta_{(1)} < \theta_{(2)} < \ldots < \theta_{(K)}$. Let $r_1, r_2, \ldots, r_K$ be the true unknown ranks of $\theta_1, \theta_2, \ldots, \theta_K$. For example, if $\theta_{(2)} = \theta_4$, then $r_4 = 2$. Our first goal is to derive a joint confidence region for $\theta_1, \theta_2, \ldots, \theta_K$ and thereby obtain joint confidence intervals for the $r_1, r_2, \ldots, r_K$ using a result from Klein et al. (2020). In deriving a joint confidence region for $\theta_1, \theta_2, \ldots, \theta_K$, we will make use of the corresponding point estimates $\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_K$. Overall uncertainty highlights that in pre-election polls, reported candidate rankings that seem decisive may in fact be much less stable once uncertainty is considered, and hence it is possible that these rankings do not align with the actual election outcomes.

Our second goal is to derive a joint confidence region for the ordered parameters $\theta_{(1)}, \theta_{(2)}, \ldots, \theta_{(K)}$. In the context of senatorial elections where θ_k denotes the proportion of votes earned by a candidate, confidence intervals on ordered parameters help determine if a candidate's lead is statistically significant. If the confidence intervals for two candidates' support levels do not overlap, it is highly likely (at the specified confidence level) that the candidate with the higher point estimate is the true winner.

A mathematical definition of r_k is as follows:

$$r_k = \sum_{j=1}^{K} I(\theta_j \leq \theta_k) = 1 + \sum_{j:j \neq k} I(\theta_j \leq \theta_k), \qquad \text{for } k = 1, 2, \ldots, K. \quad (2.1)$$

Let $\hat{r}_k$ denote the estimated rank of the kth observation. This can be computed determined based on

$$\hat{r}_k = 1 + \sum_{j:j\neq k} I(\hat{\theta}_j \leq \hat{\theta}_k), \qquad \text{for } k = 1, 2, \ldots, K. \tag{2.2}$$

In this chapter, we review the various approaches to solve this problem of finding a joint confidence region for $r_1, r_2, \ldots, r_K$.

2.2 Klein's Joint Confidence Region for Overall Ranking Uncertainty

The study of Klein et al. (2020) assumes that $\hat{\theta}_k \sim N(\theta_k, \sigma_k^2), k = 1, 2, \ldots, K$ where θ_k is unknown but σ_k^2 is known. We now describe their methodology.

Suppose that for each $k \in \{1, 2, \ldots, K\}$ there exists values L_k and U_k such that

$$\theta_k \in (L_k, U_k), \qquad U_k, L_k \in \mathbb{R} \tag{2.3}$$

Moreover, define the quantities I_k, Λ_{Lk}, Λ_{Rk}, Λ_{Ok} as follows:

$$\left.\begin{array}{c} I_k = \{1, 2, \ldots, K\} - \{k\}, \\ \Lambda_{Lk} = \{j \in I_k : U_j \leq L_k\}, \\ \Lambda_{Rk} = \{j \in I_k : U_k \leq L_j\}, \\ \Lambda_{Ok} = \{j \in I_k : U_j > L_k \text{ and } U_k > L_j\} = I_k - \{\Lambda_{Lk} \cup \Lambda_{Rk}\} \end{array}\right\}$$

which implies the following:

1. $j \in \Lambda_{Lk} \leftrightarrow (L_j, U_j) \cap (L_k, U_k) = \emptyset$ and (L_j, U_j) lies to the left of (L_k, U_k);
2. $j \in \Lambda_{Rk} \leftrightarrow (L_j, U_j) \cap (L_k, U_k) = \emptyset$ and (L_j, U_j) lies to the right of (L_k, U_k);
3. $j \in \Lambda_{Ok} \leftrightarrow (L_j, U_j) \cap (L_k, U_k) \neq \emptyset$
4. $\Lambda_{Lk}, \Lambda_{Rk}$, and Λ_{Ok} are mutually exclusive, and $\Lambda_{Lk} \cup \Lambda_{Rk} \cup \Lambda_{Ok} = I_k$

Let $|A|$ denote the number of elements in a set A. If the condition in (2.3) holds, the main result from Klein et al. (2020) gives a range for the value of r_k for each $k \in \{1, 2, \ldots, K\}$ as follows:

$$r_k \in \{|\Lambda_{Lk}| + 1, |\Lambda_{Lk}| + 2, |\Lambda_{Lk}| + 3, \ldots, |\Lambda_{Lk}| + |\Lambda_{Ok}| + 1\} \tag{2.4}$$

Note that the number of elements in the range given in (2.4) is $|\Lambda_{Ok}| + 1$. Since the smaller difference between U_k and L_k leads to a smaller $|\Lambda_{Ok}|$, narrower confidence intervals for $\theta_1, \theta_2, \ldots, \theta_K$ are desirable.

Suppose that for random quantities L_k and U_k the event defined in (2.3) satisfies the following probability condition:

$$P\left[\bigcap_{k=1}^{K} \{\theta_k \in (L_k, U_k)\}\right] \geq 1 - \alpha, \tag{2.5}$$

then, by the result of Klein et al. (2020), it also follows that

$$P\left[\bigcap_{k=1}^{K} \{r_k \in \{|\Lambda_{Lk}| + 1, |\Lambda_{Lk}| + 2, |\Lambda_{Lk}| + 3, \ldots, |\Lambda_{Lk}| + |\Lambda_{Ok}| + 1\}\}\right] \geq 1-\alpha. \tag{2.6}$$

Due to the assumption of normality on $\hat{\theta}_k$ as well as the fact that σ_k is assumed known, Klein et al. (2020) set the confidence intervals (L_k, U_k) for θ_k to be of the form $\hat{\theta}_k \pm t \times \sigma_k$ for $k \in \{1, 2, \ldots, K\}$. One may use the Bonferroni approach to choose t and call the resulting confidence region $\mathfrak{R}_{\text{bonf}}$. If such an approach is used, the choice of t that would satisfy (2.5) is $t = z_{\alpha/2K}$. Another choice of t is one that exploits the independence assumption on $\hat{\theta}_k$; denote the associated confidence region with $\mathfrak{R}_{\text{ind}}$. Such a choice is given by $z_{\gamma/2}$ where $\gamma = 1 - (1 - \alpha)^{1/K}$. To see why this is the case, note that if we define Y_k as follows:

$$Y_k = \frac{\left(\hat{\theta}_k - \theta_k\right)}{\sigma_k} \sim N(0, 1), \quad k = 1, 2, \ldots, K \tag{2.7}$$

then we get

$$\begin{aligned}\prod_{k=1}^{K} P\left(|Y_k| \leq t\right) &= \prod_{k=1}^{K} P\left(\hat{\theta}_k - t \cdot \sigma_k \leq \theta_k \leq \hat{\theta}_k + t \cdot \sigma_k\right) \\ &= \prod_{k=1}^{K} 1 - [1 - (1-\alpha)^{1/K}] \qquad (2.8) \\ &= 1 - \alpha.\end{aligned}$$

2.3 Alternative Approaches for Ranking Uncertainty

While Klein's approach provides one framework for constructing joint confidence regions for ranks, several other studies have explored related problems using different formulations or assumptions. These alternative methods vary in whether they account for dependence structures, rely on model-based estimation, or use resampling techniques such as the bootstrap.

2.3.1 Calculated Measure

Other studies with similar concern include that of Andersson et al. (1998) who suggest the use of a statistic C which quantifies the number of positions ranked parameters would on average change their order due to random variation. They calculate the measure using a bootstrap approach. Working on risk ratios p_k of K units, Andersson et al. (1998) draw B bootstrap proportions $\hat{p}_k^*$ from (2.9):

$$\hat{p}_{bk}^* \sim N\left(\hat{p}_k, \frac{\hat{p}_k(1-\hat{p}_k)}{n_k}\right), \quad k = 1, 2, \ldots, K;\; b = 1, 2, \ldots, B \qquad (2.9)$$

and $\hat{p}_k$ and n_k are the sample proportion and sample size, respectively, corresponding to the kth unit. For each bootstrap iteration b, they sort the set $\hat{p}_{b1}^*, \ldots, \hat{p}_{bK}^*$ to get the corresponding ranks $r_{bk}^*, k = 1, 2, \ldots, K$. The difference d_{bk} between the original and bootstrap rank is then calculated as $d_{bk} = |\hat{r}_k - \hat{r}^*_{bk}|$, where $\hat{r}_k$ is the rank of $\hat{p}_k$ in the set $\{\hat{p}_1, \hat{p}_2, \ldots, \hat{p}_K\}$. In turn, $\bar{d}_k$ is obtained by taking the average of d_{bk} across the bootstrap samples. Finally, the overall measure C is calculated as the average of $\bar{d}_k$ across all K units.

This can be applied to Pulse Asia senatorial pre-elections survey where we take $\hat{p}_k$ as the estimated proportion of voters who opted for candidate k and $n = 1,200$ as the common sample size. The higher C is, the less we are able to clearly distinguish between the proportion of votes received by different candidates, rendering ranks uninformative.

2.3.2 Pairwise Difference

Mohamad et al. (2019), requiring only the mean estimates and their corresponding standard errors similar to Klein et al. (2020), apply Tukey's Honest Significant Difference (HSD) to test $H_0 : \theta_j - \theta_k = 0$ for all $j \neq k \in 1, ..., K$ at level α. Tukey's HSD is typically used to provide simultaneous confidence statements about the differences between the means while controlling the family-wise error rate (FWER). Mohamad et al. (2019) come up with a joint confidence set for ranks expressed as

$$\left(1 + \#\left\{j : \frac{\hat{\theta}_j - \hat{\theta}_k}{\sqrt{\sigma_j^2 + \sigma_k^2}} > q_{1-\alpha}\right\},\ K - \#\left\{j : \frac{\hat{\theta}_j - \hat{\theta}_k}{\sqrt{\sigma_j^2 + \sigma_k^2}} < -q_{1-\alpha}\right\}\right), \tag{2.10}$$

$$\text{for } k = 1, 2, \ldots, K$$

where $q_{1-\alpha}$ is the $(1 - \alpha)$ quantile of the distribution of the studentized range,

$$\max_{j,k=1,\ldots K} \frac{|\theta_j - \theta_k|}{\sqrt{\sigma_j^2 + \sigma_k^2}}.$$

In (2.10), the notation $\#\{\cdot\}$ counts the number of pairwise hypotheses that are rejected according to Tukey's HSD, which determines the lower and upper bounds of the confidence interval for the rank of $\hat{\theta}_k$. Their method has simultaneous coverage of at least $1-\alpha$ and exactly $1-\alpha$ when $\theta_1 = \theta_2 = \cdots = \theta_K$. However, their approach tends to be overly conservative, showing coverage levels between 0.996 and 1.0 at a 0.90 nominal level in simulations, when the θs differ. They also demonstrated that as the true differences increase from 0 to 0.5, the coverage quickly increases from the nominal level to 1. As a remedy, they proposed a rescaling technique that brings the coverage closer to the nominal level, though it remains conservative (e.g., from 1.0 to 0.978, from 0.998 to 0.961—at 0.90 confidence level).

Mogstad et al. (2024) presents another technique that closely resembles the procedure by Klein et al. (2020) and Mohamad et al. (2019). However, they define ranks in the opposite way (i.e., larger rank value for lower estimate). They construct the rectangular confidence region in (2.11), from the pairwise differences of the estimators $\hat{\theta}_1, \ldots, \hat{\theta}_K$ and an estimator of the variance of $\hat{\theta}_j - \hat{\theta}_k$. The quantities $\hat{\theta}_1, \ldots, \hat{\theta}_K$ need not be independent. Let P_k be the distribution from which $\hat{\theta}_k$ is estimated; and let $\hat{P}_k$ denote the estimate of P_k. The joint confidence intervals for $r_1, \ldots, r_K$ are obtained from the joint confidence intervals of the pairwise difference:

$$C(1-\alpha, S) = \prod_{(j,k)\in S} \left[\hat{\theta}_j - \hat{\theta}_k \pm \sqrt{\sigma_j + \sigma_k}\, L^{-1}(1-\alpha, S, \hat{P})\right], \qquad S \subseteq \{(j,k) \in K \times K : j \neq k\} \tag{2.11}$$

where $L^{-1}(1-\alpha, S, P)$ is the $(1-\alpha)$-quantile corresponding to the CDF:

$$L(x, S, P) = P\left\{\max_{(j,k)\in S} \frac{|\hat{\theta}_j - \hat{\theta}_k - (\theta_j - \theta_k)|}{\sqrt{\sigma_j^2 + \sigma_k^2}} \leq x\right\}. \tag{2.12}$$

They added that if the estimators $\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_K$ are jointly asymptotically normally distributed, then the quantiles $L^{-1}(1-\alpha, S, \hat{P})$, can be computed from the limiting distributions of the max-statistics shown in (2.12), through resampling methods. Their construction of confidence sets for ranks is based on hypothesis testing. Mogstad et al. (2024) show that their approach generally leads to a confidence set that is narrower than that of Klein et al. (2020).

2.3.3 Model-Based Uncertainty

In the context of ranking education and health institutions, Goldstein & Spiegelhalter (1996) treat the estimators as latent variables. Estimators are not directly measured but inferred from observed variables. They describe performance indicator as a summary statistical measurement on an institution (e.g., schools, hospitals, medical practitioners, etc.) which is meant to be related to the quality of its functioning. An example of performance indicator is "outcome" measure such as school exam results which have been used to judge institutional effectiveness. They use multilevel models to address the hierarchical nature of data structures associated with institutional performance. In its basic form,

$$\begin{gathered} y_{ik} = \beta_0 + u_k + e_{ik} \\ \text{var}(\hat{u}_k) = \sigma_u^2 \\ \text{var}(\hat{e}_{ik}) = \sigma_e^2 \end{gathered}$$

where y_{ik} is the exam score for student i from school k, u_k is the residual or the kth school effect, and e_{ik} is the residual effect for student i from school j. This will yield posterior estimates $\hat{u}_j$ and $\text{var}(\hat{u}_j)$ as well as $\text{rank}(\hat{u}_j)$ and $\text{var}(\text{rank}(\hat{u}_j))$ which can be used to compare the institutions. They also visualize rank uncertainty and describe that the non-overlap of confidence intervals convey a significant difference between compared institutions (Goldstein & Healy, 1995). Their example illustrates that while the multilevel model made individual estimates more accurate, it also had the effect of making the ranks even more uncertain.

2.3.4 Accounting for Data Dependencies

Some approaches explicitly account for dependencies in the data. Zhang et al. (2013) analyze U.S. age-adjusted cancer incidence and mortality rates across states and counties by computing individual and overall simultaneous confidence intervals for age-adjusted health index using the Monte Carlo method. Because many health conditions are age-dependent, they use age-adjusted rates to minimize the confounding effect of age differences when comparing different population groups. They also extend their method to handle cases where only the adjusted rates and confidence intervals are available, aligning it more closely with the approach of Klein et al. (2020). Mohamad et al. (2019) show their technique to result in joint confidence sets with very low coverage probabilities and which are only able to reach the nominal level when differences among the means are large enough.

Hall & Miller (2009) mention that in some use cases such as institutions ranking, dependencies can be accommodated through conditioning, similar to the above approaches. However, in genomics where data on expression levels of different genes from the same individual are generally not independent, they suggest using an "independent component" version of the bootstrap on the sample, where m-out-of-n bootstrap (m < n) is applied as though the ranked variables were statistically independent. They show this to perform at its best when a reasonable level of correlation is present among the variables.

Bazylik et al. (2025), in their recent study, tackle the ranking of political candidates or parties using the estimated share of support each one receives in surveys. They use the approach of Mogstad et al. (2024) but account for the dependence in the estimators through their bootstrap implementation. In their setup, they let X_k be the number of votes received by the kth party from the n voters so $\mathbf{X} = (X_1, X_2, \ldots, X_K)'$ is distributed according to the multinomial distribution with parameters n and $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_K)'$. Their simulations show that bootstrap-based confidence sets may have coverage probability below the nominal level despite their being excessively wide. In contrast, the finite-sample confidence sets have coverage probability at least as large as expected and may even be relatively shorter.

2.4 Asymptotic Variance

Let $\hat{\theta}_{k,n}$ be an estimator of θ_k for $k = 1, 2, \ldots, K$ based on a sample of size n, and let $\hat{\theta}_{(k),n}$ be the kth ordered value among $\hat{\theta}_{1,n}, \hat{\theta}_{2,n}, \ldots, \hat{\theta}_{K,n}$. The estimation may be done through standard procedures such as MLE, least squares, and so forth. Moreover, suppose $\hat{\theta}_{k,n}$s are independent for $k = 1, 2, \ldots, K$. Finally, suppose, as in the application of this thesis, that $\hat{\theta}_{k,n}$ is a sample mean (or sample proportion) and θ_k is a population mean (or population proportion) and both $\hat{\theta}_{k,n}$ and θ_k are nonnegative. In this subsection, we show that

$$V[\hat{\theta}_{(k),n}] \longrightarrow k\text{th ordered value in } \{\theta_1^2 + \sigma_1^2, \theta_2^2 + \sigma_2^2, \ldots, \theta_K^2 + \sigma_K^2\} - \theta_{(k)}^2 \tag{2.13}$$

as $n \to \infty$, where $\sigma_k^2 = V(\hat{\theta}_{k,n})$, which in this study is assumed to be known. The result in (2.13) gives us an expression for the asymptotic variance of $\hat{\theta}_{(k),n}$. We shall also give an expression for an estimator of this variance.

By the definition of variance, we have

$$V(\hat{\theta}_{(k),n}) = E(\hat{\theta}_{(k),n}^2) - [E(\hat{\theta}_{(k),n})]^2 \tag{2.14}$$

We make use of Theorem 2.2 of Chen (1976) on asymptotic results for ordered estimators, which we shall now state.

Theorem (Chen, 1976). Suppose that the estimator $\hat{\delta}_{k,n}$ is a strongly consistent estimator of the parameter δ_k for $k = 1, 2, \ldots, K$ and based on a random sample of size n. Moreover, suppose $\hat{\delta}_{1,n}, \hat{\delta}_{2,n}, \ldots \hat{\delta}_{K,n}$ are independent. It follows that $\hat{\delta}_{(k),n}$ is a strongly consistent estimator of $\delta_{(k)}$, where $\hat{\delta}_{(k),n}$ is the kth ordered value of $\hat{\delta}_{1,n}, \hat{\delta}_{2,n}, \ldots, \hat{\delta}_{K,n}$ and $\delta_{(k)}$ is the kth ordered value of the $\delta_1, \delta_2, \ldots, \delta_K$.

Since $\hat{\theta}_{k,n}$, being a sample mean, is a strongly consistent estimator of θ_k, then using the result of Chen (1976), we know that $\hat{\theta}_{(k),n}$ is a strongly consistent estimator of $\theta_{(k)}$. This implies that

$$E[\hat{\theta}_{(k),n}] \longrightarrow \theta_{(k)}. \tag{2.15}$$

Moreover, since $E(\hat{\theta}^2_{k,n}) = \theta_k^2 + \sigma_k^2$ we also have

$$E[\hat{\theta}^2_{(k),n}] \longrightarrow k\text{th ordered value in } \{\theta_1^2 + \sigma_1^2, \theta_2^2 + \sigma_2^2, \ldots, \theta_K^2 + \sigma_K^2\}. \tag{2.16}$$

Using the results in (2.15) and (2.16) and noting the formula in (2.14), we get the result in (2.13). It also follows that we can estimate $V[\hat{\theta}_{(k),n}]$ as follows:

$$\widehat{V[\hat{\theta}_{(k),n}]} = \text{kth ordered value among } \left\{\hat{\theta}^2_{1,n} + \sigma_1^2, \ldots, \hat{\theta}^2_{K,n} + \sigma_K^2\right\} - \hat{\theta}^2_{(k),n}. \tag{2.17}$$

In this subsection, we have included the subscript n in our estimators to indicate that the convergence occurs as n tends to infinity. In the rest of this thesis, we shall drop this subscript.

3 Methodology

This section introduces the proposed methodologies to obtain joint confidence intervals that can be used to quantify uncertainty for the unknown overall true ranking. It addresses the scenario when the estimates of the parameters being ranked are correlated to certain degrees. Section 3.1 develops the procedure used to compute the joint confidence region for $\theta_1, \theta_2, \ldots, \theta_K$. These include a non-rank and rank-based methods. Section 3.2, on the other hand, develops the procedure to compute the joint confidence region for $\theta_{(1)}, \theta_{(2)}, \ldots, \theta_{(K)}$. Section 3.3 discusses some possible correlation structures to be considered for the estimates. Sections 3.4 and 3.5 discuss how to evaluate the proposed methodology and the simulation settings to be used.

3.1 Proposed methodology to compute the joint confidence region for the unordered parameters

In this section, we adopt the notations defined in Section 2.1 and 2.2. Moreover, we define $\hat{\boldsymbol{\theta}} = (\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_K)'$ and assume that $\hat{\boldsymbol{\theta}} \sim N(\boldsymbol{\theta}, \boldsymbol{\Sigma})$ where $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_K)'$ is unknown and $\boldsymbol{\Sigma}$ is a known $K \times K$ positive definite matrix. The diagonal elements of $\boldsymbol{\Sigma}$ are $\sigma_1^2, \ldots, \sigma_K^2$. We note that in the literature on inferences on the ranks, it is customary to assume that the variances are known. As previously mentioned, the setup in this thesis allows the estimates to be correlated, which is in contrast with that of Klein et al. (2020), which assumes independence among the estimates. In other words, in this thesis, the covariance matrix $\boldsymbol{\Sigma}$ need not be a diagonal matrix. Similar to the study of Klein et al. (2020), knowledge of the sampling design and estimation methodology for each population is not required; we shall first derive simultaneous confidence intervals for $\theta_1, \theta_2, \ldots, \theta_K$ of the form

$$\mathfrak{R}_1 = [\hat{\theta}_1 \pm t \times \sigma_1] \times [\hat{\theta}_2 \pm t \times \sigma_2] \times \cdots \times [\hat{\theta}_K \pm t \times \sigma_K]. \tag{3.1}$$

We want the joint confidence region in (3.1) to satisfy the following probability condition:

$$P\left(\hat{\theta}_k - t \cdot \sigma_k \leq \theta_k \leq \hat{\theta}_k + t \cdot \sigma_k,\ \forall\, k = 1, 2, \ldots, K\right) = 1 - \alpha. \tag{3.2}$$

Equivalently, we require

$$P\left(\max_{k=1,2,\ldots,K} \left|\frac{\hat{\theta}_k - \theta_k}{\sigma_k}\right| \leq t\right) = 1 - \alpha. \tag{3.3}$$

We use the parametric bootstrap to estimate the quantile t that will be used to construct the confidence intervals while controlling the coverage to be around the nominal level. In implementing the parametric bootstrap, we sample from the multivariate normal distribution with $\hat{\boldsymbol{\theta}}$ and $\boldsymbol{\Sigma}$ as mean and variance, respectively. The idea is also conceptually similar to that of Mogstad et al. (2024) who use resampling to obtain the quantile in a different context. Algorithm 1 describes the procedure to compute the joint confidence region based on unordered estimates.

Once the confidence intervals in (3.1) have been obtained, we can then use the result of Klein et al. (2020) in (2.6) to get the lower and upper bounds on the ranks $r_k, k = 1, 2, \ldots, K$. That is, we also get a joint confidence region for $r_1, r_2, ..., r_K$.

Algorithm 1 Computing the joint confidence region for the unordered parameters

Let the data be represented by $\hat{\boldsymbol{\theta}} = \left(\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_K\right)'$ and suppose that $\boldsymbol{\Sigma}$ is known

```
for $b = 1, 2, \ldots, B$ do
    Generate $\hat{\boldsymbol{\theta}}_b^* \sim N_K\left(\hat{\boldsymbol{\theta}}, \boldsymbol{\Sigma}\right)$ and write $\hat{\boldsymbol{\theta}}_b^* = \left(\hat{\theta}_{b1}^*, \hat{\theta}_{b2}^*, \ldots, \hat{\theta}_{bK}^*\right)'$
    Compute
        $t_b^* = \max_{1 \le k \le K} \left| \frac{\hat{\theta}_{bk}^* - \hat{\theta}_k}{\sigma_k} \right|$
end for
Compute the $(1-\alpha)$-sample quantile of $t_1^*, t_2^*, \ldots, t_B^*$, call this $\hat{t}$.
The joint confidence region for $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_K)'$ is given by
    $\mathfrak{R}_1 = \left[\hat{\theta}_1 \pm \hat{t} \times \sigma_1\right] \times \left[\hat{\theta}_2 \pm \hat{t} \times \sigma_2\right] \times \cdots \times \left[\hat{\theta}_K \pm \hat{t} \times \sigma_K\right].$
```

3.2 Proposed methodology to compute a joint confidence region for the ordered parameters

We shall also be proposing an approach to compute simultaneous confidence intervals for the ordered parameters. Suppose that for the parameters $\theta_1, \ldots, \theta_K$, the corresponding ordered values are $\theta_{(1)}, \ldots, \theta_{(K)}$. Similarly, let $\hat{\theta}_{(1)}, \ldots, \hat{\theta}_{(K)}$ be the ordered estimates. We now derive a joint confidence region for these ordered values, and set the region to be of the following form:

$$\mathfrak{R}_2 = [\hat{\theta}_{(1)} \pm t \times \sigma_{(1)}] \times [\hat{\theta}_{(2)} \pm t \times \sigma_{(2)}] \times \cdots \times [\hat{\theta}_{(K)} \pm t \times \sigma_{(K)}], \quad (3.4)$$

where $\sigma_{(k)} = \sqrt{V\left(\hat{\theta}_{(k)}\right)}$. The confidence region in (3.4) is computed such that the following condition holds:

$$P\left(\hat{\theta}_{(k)} - t \cdot \sigma_{(k)} \leq \theta_{(k)} \leq \hat{\theta}_{(k)} + t \cdot \sigma_{(k)},\ \forall k = 1, 2, \ldots, K\right) = 1 - \alpha, \tag{3.5}$$

which is equivalent to

$$P\left(\max_{k=1,2,\ldots,K}\left|\frac{\hat{\theta}_{(k)} - \theta_{(k)}}{\sigma_{(k)}}\right| \leq t\right) = 1 - \alpha, \tag{3.6}$$

and we estimate t, which in (3.6) is the $(1-\alpha)$-quantile of the distribution of $\max_k \left|\frac{\hat{\theta}_{(k)} - \theta_{(k)}}{\sigma_{(k)}}\right|$, via a suitable parametric bootstrap. Since $\sigma_{(k)}$ is not known, we estimate its value. We present two approaches to estimate $\sigma_{(k)}$. The first uses results from Chen (1976) and Dudewicz (1972) to obtain an expression of the asymptotic variance of $\hat{\theta}_{(k)}$ as follows:

$$V(\hat{\theta}_{(k)}) = \text{kth ordered value among } \left\{\theta_1^2 + \sigma_1^2, \theta_2^2 + \sigma_2^2, \ldots, \theta_K^2 + \sigma_K^2\right\} - \theta_{(k)}^2. \tag{3.7}$$

Consequently, we get an estimate of the asymptotic variance of $\hat{\theta}_{(k)}$, given by:

$$\widehat{V(\hat{\theta}_{(k)})} = \text{kth ordered value among } \left\{\hat{\theta}_1^2 + \sigma_1^2, \hat{\theta}_2^2 + \sigma_2^2, \ldots, \hat{\theta}_K^2 + \sigma_K^2\right\} - \hat{\theta}_{(k)}^2. \tag{3.8}$$

Algorithm 2 describes the procedure to compute the joint confidence region in (3.4) using this approach.

Algorithm 2 Computing the joint confidence region for the ordered parameters using the estimated asymptotic variance

```
for $b = 1, 2, \ldots, B$ do
    Generate $\hat{\boldsymbol{\theta}}_b^* = \left(\hat{\theta}_{b1}^*, \hat{\theta}_{b2}^*, \ldots, \hat{\theta}_{bK}^*\right)' \sim N_K\left(\hat{\boldsymbol{\theta}}, \boldsymbol{\Sigma}\right)$ and let
    $\hat{\theta}_{b(1)}^*, \hat{\theta}_{b(2)}^*, \ldots, \hat{\theta}_{b(K)}^*$ be the corresponding ordered values
    Compute
    $\hat{\sigma}_{b(k)}^* = \sqrt{\left[\text{kth ordered value among } \left\{\hat{\theta}_{b1}^{*2} + \sigma_1^2, \hat{\theta}_{b2}^{*2} + \sigma_2^2, \ldots, \hat{\theta}_{bK}^{*2} + \sigma_K^2\right\}\right] - \hat{\theta}_{(k)}^{*2}}$
    Compute $t_b^* = \max_{1 \le k \le K} \left| \frac{\hat{\theta}_{b(k)}^* - \hat{\theta}_{(k)}}{\hat{\sigma}_{b(k)}^*} \right|$
end for
Compute the $(1-\alpha)$-sample quantile of $t_1^*, t_2^*, \ldots, t_B^*$, call this $\hat{t}$.
The joint confidence region of $\theta_{(1)}, \theta_{(2)}, \ldots, \theta_{(K)}$ is given by
    $\mathfrak{R}_2 = \left[\hat{\theta}_{(1)} \pm \hat{t} \times \hat{\sigma}_{(1)}\right] \times \left[\hat{\theta}_{(2)} \pm \hat{t} \times \hat{\sigma}_{(2)}\right] \times \cdots \times \left[\hat{\theta}_{(K)} \pm \hat{t} \times \hat{\sigma}_{(K)}\right]$
where $\hat{\sigma}_{(k)}$ is computed as
    $\hat{\sigma}_{(k)} = \sqrt{\text{kth ordered value among } \left\{\hat{\theta}_1^2 + \sigma_1^2, \hat{\theta}_2^2 + \sigma_2^2, \ldots, \hat{\theta}_K^2 + \sigma_K^2\right\} - \hat{\theta}_{(k)}^2}$
```

The expression for the asymptotic variance of $\hat{\theta}_{(k)}$ has actually been derived in a setting where the $\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_K$ are independent. It is not clear if the expression in (3.7) holds whenever $\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_K$ are correlated. Since this thesis allows for the $\hat{\theta}_k$s to be correlated, it is possible that the expression in (3.8) for the estimate of the variance of $\hat{\theta}_{(k)}$ lacks optimality. As an alternative to using the asymptotic variance, we shall also consider using the bootstrap to estimate the variance of $\hat{\theta}_{(k)}$. To this end, a second-level bootstrap can be employed to estimate the variance. Algorithm 3 illustrates the procedure to compute the joint confidence region for the ordered parameters with the use of the bootstrap to estimate the variance of the ordered estimates. As may be expected, this approach is computationally intensive.

Algorithm 3 Computing the joint confidence region for the ordered parameters using the bootstrap estimate of the variance

```
for $b = 1, 2, \dots, B$ do
    Generate $\hat{\boldsymbol{\theta}}^*_b = \left(\hat{\theta}^*_{b1}, \hat{\theta}^*_{b2}, \dots, \hat{\theta}^*_{bK}\right)' \sim N_K\left(\hat{\boldsymbol{\theta}}, \boldsymbol{\Sigma}\right)$ and let $\hat{\theta}^*_{b(1)}, \hat{\theta}^*_{b(2)}, \dots, \hat{\theta}^*_{b(K)}$ be the corresponding ordered values of $\hat{\theta}^*_{b1}, \hat{\theta}^*_{b2}, \dots, \hat{\theta}^*_{bK}$
    for $c = 1, 2, \dots, C$ do
        Generate $\hat{\boldsymbol{\theta}}^{**}_{bc} = \left(\hat{\theta}^{**}_{bc1}, \hat{\theta}^{**}_{bc2}, \dots, \hat{\theta}^{**}_{bcK}\right) \sim N_K\left(\hat{\boldsymbol{\theta}}^*_b, \boldsymbol{\Sigma}\right)$ and let $\hat{\theta}^{**}_{bc(1)}, \hat{\theta}^{**}_{bc(2)}, \dots, \hat{\theta}^{**}_{bc(K)}$ be the corresponding ordered values of $\hat{\theta}^{**}_{bc1}, \hat{\theta}^{**}_{bc2}, \dots, \hat{\theta}^{**}_{bcK}$
        Compute $\hat{\sigma}^*_{b(k)} = \sqrt{\frac{\sum_{c=1}^{C}\left(\hat{\theta}^{**}_{bc(k)} - \bar{\hat{\theta}}^{**}_{b\cdot(k)}\right)^2}{C-1}}, \quad \bar{\hat{\theta}}^{**}_{b\cdot(k)} = \frac{1}{C}\sum_{c=1}^{C}\hat{\theta}^{**}_{bc(k)}$
    end for
    Compute $t^*_b = \max_{1 \le k < K}\left|\frac{\hat{\theta}^*_{b(k)} - \hat{\theta}_{(k)}}{\hat{\sigma}^*_{b(k)}}\right|$
end for
Compute the $(1-\alpha)$-sample quantile of $t^*_1, t^*_2, \dots, t^*_B$, call this $\hat{t}$.
The joint confidence region of $\theta_{(1)}, \theta_{(2)}, \dots, \theta_{(K)}$ is
    $\mathfrak{R}_2 = \left[\hat{\theta}_{(1)} \pm \hat{t} \times \hat{\sigma}_{(1)}\right] \times \left[\hat{\theta}_{(2)} \pm \hat{t} \times \hat{\sigma}_{(2)}\right] \times \cdots \times \left[\hat{\theta}_{(K)} \pm \hat{t} \times \hat{\sigma}_{(K)}\right]$
where $\hat{\sigma}_{(k)}$ is computed as
    $\hat{\sigma}_{(k)} = \sqrt{\frac{\sum_{b=1}^{B}\left(\hat{\theta}^*_{b(k)} - \bar{\hat{\theta}}^*_{\cdot(k)}\right)^2}{B-1}}, \quad \bar{\hat{\theta}}^*_{\cdot(k)} = \frac{1}{B}\sum_{b=1}^{B}\hat{\theta}^*_{b(k)}$
```

3.3 Correlation structures

The proposed methodologies assume that $V(\hat{\boldsymbol{\theta}}) = \boldsymbol{\Sigma}$ is known. Note that we can express $\boldsymbol{\Sigma}$ as in (3.9), where $\mathbf{R}$ is the population correlation matrix.

$$\boldsymbol{\Sigma} = \boldsymbol{\Delta}^{1/2}\mathbf{R}\boldsymbol{\Delta}^{1/2}; \quad \boldsymbol{\Delta} = \operatorname{diag}\left\{\sigma_1^2, \sigma_2^2, \dots, \sigma_K^2\right\}. \tag{3.9}$$

The diagonal elements of $\mathbf{\Sigma}$, which are $\sigma_k^2 = V(\hat{\theta}_k)$ for $k = 1, 2, \ldots, K$, are treated as known quantities in practice. With large sample sizes, the variance estimates are stable enough that they are treated as the actual values. However, there is limited information to use as basis for the correlations among the estimates. In this thesis, we shall be assuming certain correlation structures among the $\hat{\theta}$s.

One structure that may be used is an equicorrelation matrix shown in (3.10). This assumes that the K variables are equally correlated, i.e., that $\rho_{jk} = \rho$ where $\rho \in [-1, 1]$ for $j \neq k \in \{1, \ldots, K\}$.

$$\mathbf{R}_{\text{eq}} = (1-\rho)\,\mathbf{I}_K + \rho \mathbf{1}_K \mathbf{1}_K' = \begin{bmatrix} 1 & \rho & \cdots & \rho \\ \rho & 1 & \cdots & \rho \\ \vdots & \vdots & \ddots & \vdots \\ \rho & \rho & \cdots & 1 \end{bmatrix}_{K \times K} \tag{3.10}$$

In a block correlation matrix $\mathbf{R}_{block}$ with G blocks, as represented by Archakova & Hansen (2020), the correlation between any two variables is determined by the block to which the two variables belong. Each diagonal block represents an equicorrelation structure within group g, denoted by

$$\mathbf{R}_{\text{eq,g}} = (1-\rho_g)\,\mathbf{I}_{n_g} + \rho_g \mathbf{1}_{n_g} \mathbf{1}_{n_g}'$$

where ρ_g is the within-block correlation and n_g is the number of variables in block g such that $\sum_{g=1}^{G} n_g = K$. The off-diagonal blocks capture between-block correlations, represented by

$$\mathbf{C}_{g'g} = \mathbf{C}_{gg'} = \rho_{gg'} \mathbf{1}_{n_g} \mathbf{1}_{n_g}'$$
$$\text{where } g \neq g' \in \{1, \ldots, G\}$$

Thus, the full block correlation matrix can be expressed as in (3.11).

$$\mathbf{R}_{\text{block}} = \begin{bmatrix} \mathbf{R}_{eq,1} & \mathbf{C}_{12} & \cdots & \mathbf{C}_{1G} \\ \mathbf{C}_{21} & \mathbf{R}_{eq,2} & \cdots & \mathbf{C}_{2G} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{C}_{G1} & \mathbf{C}_{G2} & \cdots & \mathbf{R}_{eq,G} \end{bmatrix}_{K \times K} \tag{3.11}$$

In the context of pre-election surveys, each block may represent correlations induced by party or ticket membership, reflecting stronger associations within parties and weaker associations between them.

Correlation structures that account for spatial proximity can be borrowed from geostatistics. This is particularly relevant in light of Klein's observation that states located within certain regions exhibit similar travel time characteristics. In such cases, spatial dependence can be modeled using a stationary (i.e., no directional dependence) Matérn correlation function, which for two locations $\mathbf{s}_i$ and $\mathbf{s}_j$ is expressed as in (3.12).

$$\rho_{\text{matern}} = \frac{2^{1-\nu}}{\Gamma(\nu)}(\kappa \parallel \mathbf{s}_i - \mathbf{s}_j \parallel)^{\nu} K_{\nu}(\kappa \parallel \mathbf{s}_i - \mathbf{s}_j \parallel) \tag{3.12}$$

where $\parallel \cdot \parallel$ denotes the Euclidean distance and K_ν is the second kind of the modified Bessel function. It has a scale parameter $\kappa > 0$ and a smoothness parameter $\nu > 0$. ρ_{matern} reduces to the exponential correlation when $\nu = 0.5$ and to Gaussian correlation function when $\nu = \infty$. In this paper, the R package "BayesNGSP" (Turek & Risser (2022)), is used to construct the $\mathbf{R}_{\text{matern}}$.

3.4 Evaluation

Algorithm 4 estimates the coverage probabilities associated with the proposed methodologies. The coverage probabilities correspond to the proportion of replications in which the true parameter values are contained within the confidence intervals for all K simultaneously. Moreover, the tightness of the joint confidence region that results from Algorithm 1 is assessed using three summary measures: the arithmetic mean (T_1), geometric mean (T_2), and the metric T_3 introduced by Wright (2025), as presented in Equations 3.13–3.15.

$$T_1 = \frac{1}{K}\sum_{k=1}^{K}\left|\Lambda_{Ok}\right| \tag{3.13}$$

$$T_2 = \prod_{k=1}^{K} \left|\Lambda_{Ok}\right| \tag{3.14}$$

$$T_3 = 1 - \frac{OP}{K^2} \tag{3.15}$$

In equation 3.15, $OP = K + \Sigma_{k=1}^{K} \left|\Lambda_{Ok}\right|$ denotes the total number of occupied positions in a joint confidence region out of the total number of positions K^2; or the sum of the differences between the upper and lower bound of the simultaneous rank intervals added by 1, for each k. Higher values of T_1 and T_2 indicate wider confidence intervals and are therefore less desirable, whereas higher values of T_3 are preferable. T_3 can range from 0, indicating no tightness, to $\frac{K-1}{K}$, implying the confidence region only contains the estimated ranking which is likely the true ranking.

Algorithm 4 Computing the coverage probability and tightness measures

For given values of $\mathbf{\Sigma}$ and $\theta_1, \theta_2, \ldots, \theta_K$ (with corresponding $\theta_{(1)}, \theta_{(2)}, \ldots, \theta_{(K)}$ for rank-based methods)

```
1: for replications = $1, 2, \ldots, 5000$ do
2:     Generate $\hat{\boldsymbol{\theta}} \sim N_K(\boldsymbol{\theta}, \mathbf{\Sigma})$
3:     Compute the confidence region for the unordered (ordered) parameters
       using Algorithm 1 ($\mathfrak{R}_1$) (Algorithms 2 and 3 ($\mathfrak{R}_2$)), Bonferroni ($\mathfrak{R}_{\text{bonf}}$),
       and independence assumption ($\mathfrak{R}_{\text{ind}}$).
4:     Check if $(\theta_1, \theta_2, \ldots, \theta_K) \in \mathfrak{R}_1, \mathfrak{R}_{\text{bonf}}, \mathfrak{R}_{\text{ind}}$
       $\left(\left(\theta_{(1)}, \theta_{(2)}, \ldots, \theta_{(K)}\right) \in \mathfrak{R}_2\right)$ for the unordered parameters (ordered
       parameters) and compute $T_1, T_2$, and $T_3$.
5: end for
6: Compute the coverage in line 4 is satisfied and the average of $T_1, T_2$, and
   $T_3$. Results of Algorithm 1 ($\mathfrak{R}_1$) is compared to ($\mathfrak{R}_{\text{bonf}}$ and $\mathfrak{R}_{\text{ind}}$).
```

3.5 Simulation settings

For the simulation settings to be used in evaluating the performance of the proposed methodologies, we shall be varying the settings for the population mean $\boldsymbol{\theta}$, population variances $\sigma_1^2, \sigma_2^2, \ldots, \sigma_K^2$, population correlation matrix $\mathbf{R}$, and number of populations being ranked K. In carrying out the simulations, a nominal level of $1 - \alpha = 0.95$ will be used. The table below summarizes the simulation settings.

$\boldsymbol{\theta}$	**R**
K = 10	
(22.5, 24.9, 27.5, 24.3, 24.8, 23.2, 23, 21.6, 25.1, 19)′	$\mathbf{R}_{\text{eq}}, \rho = 0.1, 0.5, 0.9$ $\mathbf{R}_{\text{block}}$ - 2, 3 blocks
(21.5, 25.7, 30.5, 24.7, 25.5, 22.6, 22.3, 19.9, 26.1, 15.2)′	$\mathbf{R}_{\text{eq}}, \rho = 0.1, 0.5, 0.9$ $\mathbf{R}_{\text{block}}$ - 2, 3 blocks
(19.9, 27, 34.9, 25.3, 26.7, 21.9, 21.3, 17.3, 27.7, 9.4)′	$\mathbf{R}_{\text{eq}}, \rho = 0.1, 0.5, 0.9$ $\mathbf{R}_{\text{block}}$ - 2, 3 blocks
K = 20	
(22.5, 24.9, 27.5, 24.3, 24.8, 23.2, 23, 21.6, 25.1, 19, 21.3, 24.1, 26.5, 26.9, 23.2, 25.8, 21.8, 21.6, 25.6, 21.2)′	$\mathbf{R}_{\text{eq}}, \rho = 0.1, 0.5, 0.9$ $\mathbf{R}_{\text{block}}$ - 2, 3 blocks
(21.5, 25.7, 30.5, 24.7, 25.5, 22.6, 22.3, 19.9, 26.1, 15.2, 19.3, 24.3, 28.7, 29.3, 22.7, 27.5, 20.1, 19.9, 27.1, 19.1)′	$\mathbf{R}_{\text{eq}}, \rho = 0.1, 0.5, 0.9$ $\mathbf{R}_{\text{block}}$ - 2, 3 blocks
(19.9, 27, 34.9, 25.3, 26.7, 21.9, 21.3, 17.3, 27.7, 9.4, 16.2, 24.6, 31.9, 33, 21.9, 29.9, 17.7, 17.3, 29.3, 16)′	$\mathbf{R}_{\text{eq}}, \rho = 0.1, 0.5, 0.9$ $\mathbf{R}_{\text{block}}$ - 2, 3 blocks
K = 30	
(22.5, 24.9, 27.5, 24.3, 24.8, 23.2, 23, 21.6, 25.1, 19, 21.3, 24.1, 26.5, 26.9, 23.2, 25.8, 21.8, 21.6, 25.6, 21.2, 25.7, 22.2, 22.7, 22.6, 21.9, 24.2, 23.6, 22, 24.5, 25)′	$\mathbf{R}_{\text{eq}}, \rho = 0.1, 0.5, 0.9$ $\mathbf{R}_{\text{block}}$ - 2, 3 blocks
(21.5, 25.7, 30.5, 24.7, 25.5, 22.6, 22.3, 19.9, 26.1, 15.2, 19.3, 24.3, 28.7, 29.3, 22.7, 27.5, 20.1, 19.9, 27.1, 19.1, 27.2, 20.9, 21.8, 21.7, 20.4, 24.5, 23.5, 20.5, 25.1, 26)′	$\mathbf{R}_{\text{eq}}, \rho = 0.1, 0.5, 0.9$ $\mathbf{R}_{\text{block}}$ - 2, 3 blocks
(19.9, 27, 34.9, 25.3, 26.7, 21.9, 21.3, 17.3, 27.7, 9.4, 16.2, 24.6, 31.9, 33, 21.9, 29.9, 17.7, 17.3, 29.3, 16, 29.4, 19, 20.4, 20.3, 18.2, 25, 23.3, 18.3, 26, 27.4)′	$\mathbf{R}_{\text{eq}}, \rho = 0.1, 0.5, 0.9$ $\mathbf{R}_{\text{block}}$ - 2, 3 blocks

$\boldsymbol{\theta}$	R
K = 40	
(22.5, 24.9, 27.5, 24.3, 24.8, 23.2, 23, 21.6, 25.1, 19, 21.3, 24.1, 26.5, 26.9, 23.2, 25.8, 21.8, 21.6, 25.6, 21.2, 25.7, 22.2, 22.7, 22.6, 21.9, 24.2, 23.6, 22, 24.5, 25, 26.5, 25.1, 25.2, 27.1, 25.3, 22.8, 20.6, 21.1, 26, 26)′	$\mathbf{R}_{\text{eq}}, \rho = 0.1, 0.5, 0.9$ $\mathbf{R}_{\text{block}}$ - 2, 3 blocks
(21.5, 25.7, 30.5, 24.7, 25.5, 22.6, 22.3, 19.9, 26.1, 15.2, 19.3, 24.3, 28.7, 29.3, 22.7, 27.5, 20.1, 19.9, 27.1, 19.1, 27.2, 20.9, 21.8, 21.7, 20.4, 24.5, 23.5, 20.5, 25.1, 26, 28.6, 26.1, 26.4, 29.7, 26.5, 22.1, 18, 18.9, 27.8, 27.8)′	$\mathbf{R}_{\text{eq}}, \rho = 0.1, 0.5, 0.9$ $\mathbf{R}_{\text{block}}$ - 2, 3 blocks
(19.9, 27, 34.9, 25.3, 26.7, 21.9, 21.3, 17.3, 27.7, 9.4, 16.2, 24.6, 31.9, 33, 21.9, 29.9, 17.7, 17.3, 29.3, 16, 29.4, 19, 20.4, 20.3, 18.2, 25, 23.3, 18.3, 26, 27.4, 31.9, 27.7, 28.1, 33.6, 28.2, 20.9, 14.1, 15.6, 30.5, 30.5)′	$\mathbf{R}_{\text{eq}}, \rho = 0.1, 0.5, 0.9$ $\mathbf{R}_{\text{block}}$ - 2, 3 blocks
K = 50	
(22.5, 24.9, 27.5, 24.3, 24.8, 23.2, 23, 21.6, 25.1, 19, 21.3, 24.1, 26.5, 26.9, 23.2, 25.8, 21.8, 21.6, 25.6, 21.2, 25.7, 22.2, 22.7, 22.6, 21.9, 24.2, 23.6, 22, 24.5, 25, 26.5, 25.1, 25.2, 27.1, 25.3, 22.8, 20.6, 21.1, 26, 26, 22.1, 21.4, 23.2, 22.3, 22.4, 23.1, 21.4, 24.8, 27, 23.4)′	$\mathbf{R}_{\text{eq}}, \rho = 0.1, 0.5, 0.9$ $\mathbf{R}_{\text{block}}$ - 2, 3 blocks
(21.5, 25.7, 30.5, 24.7, 25.5, 22.6, 22.3, 19.9, 26.1, 15.2, 19.3, 24.3, 28.7, 29.3, 22.7, 27.5, 20.1, 19.9, 27.1, 19.1, 27.2, 20.9, 21.8, 21.7, 20.4, 24.5, 23.5, 20.5, 25.1, 26, 28.6, 26.1, 26.4, 29.7, 26.5, 22.1, 18, 18.9, 27.8, 27.8, 20.8, 19.5, 22.8, 21.2, 21.3, 22.5, 19.5, 25.5, 29.6, 23)′	$\mathbf{R}_{\text{eq}}, \rho = 0.1, 0.5, 0.9$ $\mathbf{R}_{\text{block}}$ - 2, 3 blocks
(19.9, 27, 34.9, 25.3, 26.7, 21.9, 21.3, 17.3, 27.7, 9.4, 16.2, 24.6, 31.9, 33, 21.9, 29.9, 17.7, 17.3, 29.3, 16, 29.4, 19, 20.4, 20.3, 18.2, 25, 23.3, 18.3, 26, 27.4, 31.9, 27.7, 28.1, 33.6, 28.2, 20.9, 14.1, 15.6, 30.5, 30.5, 18.8, 16.6, 22.1, 19.4, 19.6, 21.6, 16.7, 26.7, 33.5, 22.5)′	$\mathbf{R}_{\text{eq}}, \rho = 0.1, 0.5, 0.9$ $\mathbf{R}_{\text{block}}$ - 2, 3 blocks

K	$\boldsymbol{\sigma}$
10	$(0.14, 0.33, 0.15, 0.23, 0.07, 0.19, 0.19, 0.37, 0.32, 0.11)'$
20	$(0.14, 0.33, 0.15, 0.23, 0.07, 0.19, 0.19, 0.37, 0.32, 0.11, 0.17, 0.27, 0.24, 0.11, 0.11, 0.13, 0.16, 0.15, 0.15, 0.25)'$
30	$(0.14, 0.33, 0.15, 0.23, 0.07, 0.19, 0.19, 0.37, 0.32, 0.11, 0.17, 0.27, 0.24, 0.11, 0.11, 0.13, 0.16, 0.15, 0.15, 0.25, 0.15, 0.13, 0.1, 0.1, 0.24, 0.13, 0.32, 0.19, 0.27, 0.3)'$
40	$(0.14, 0.33, 0.15, 0.23, 0.07, 0.19, 0.19, 0.37, 0.32, 0.11, 0.17, 0.27, 0.24, 0.11, 0.11, 0.13, 0.16, 0.15, 0.15, 0.25, 0.15, 0.13, 0.1, 0.1, 0.24, 0.13, 0.32, 0.19, 0.27, 0.3, 0.12, 0.27, 0.09, 0.12, 0.36, 0.09, 0.15, 0.16, 0.09, 0.29)'$
50	$(0.14, 0.33, 0.15, 0.23, 0.07, 0.19, 0.19, 0.37, 0.32, 0.11, 0.17, 0.27, 0.24, 0.11, 0.11, 0.13, 0.16, 0.15, 0.15, 0.25, 0.15, 0.13, 0.1, 0.1, 0.24, 0.13, 0.32, 0.19, 0.27, 0.3, 0.12, 0.27, 0.09, 0.12, 0.36, 0.09, 0.15, 0.16, 0.09, 0.29, 0.16, 0.28, 0.14, 0.07, 0.2, 0.31, 0.13, 0.14, 0.31, 0.11)'$

4 Methodology

4.1 Results

The results of nonrank-based algorithm are compared to the approaches from Klein et al. (2020): independent and Bonferroni.

Table 1: Simulation results for coverage probabilities

K	r	Independent	Bonferroni	Nonrank-based
10	0.1	0.9474	0.9484	0.9464
	0.5	0.9614	0.9620	0.9496
	0.9	0.9840	0.9840	0.9482
20	0.1	0.9450	0.9472	0.9438
	0.5	0.9610	0.9614	0.9462
	0.9	0.9864	0.9864	0.9522
30	0.1	0.9510	0.9522	0.9488
	0.5	0.9646	0.9652	0.9504
	0.9	0.9904	0.9908	0.9526
40	0.1	0.9496	0.9506	0.9462
	0.5	0.9612	0.9626	0.9442
	0.9	0.9880	0.9880	0.9462
50	0.1	0.9508	0.9520	0.9472
	0.5	0.9686	0.9688	0.9472
	0.9	0.9918	0.9918	0.9488

Table 2: Simulation results for tightness measure t1

		Low			Med			High		
K	r	Independent	Bonferroni	Nonrank-based	Independent	Bonferroni	Nonrank-based	Independent	Bonferroni	Nonrank-based
10	0.1	3.44444	3.45052	3.435920	2.445560	2.450680	2.440200	1.802560	1.805040	1.801920
10	0.5	3.43376	3.43964	3.363240	2.459800	2.463880	2.405360	1.799520	1.801760	1.775320
10	0.9	3.41648	3.42316	3.094360	2.503840	2.508320	2.164800	1.784600	1.787160	1.676640
20	0.1	5.67294	5.68438	5.661240	3.645120	3.651960	3.638040	2.572500	2.575400	2.568080
20	0.5	5.66384	5.67506	5.488720	3.658580	3.665540	3.556200	2.562460	2.566240	2.508340
20	0.9	5.67722	5.68764	4.837580	3.679540	3.685680	3.163200	2.549400	2.553100	2.318540
30	0.1	10.14211	10.16159	10.115880	6.180720	6.193707	6.162293	3.999533	4.007147	3.988707
30	0.5	10.14804	10.16737	9.796840	6.194267	6.207347	5.958587	4.001600	4.008733	3.868147
30	0.9	10.18795	10.20716	8.519027	6.216440	6.228240	5.091240	4.005987	4.013373	3.402453
40	0.1	12.82436	12.84825	12.789230	7.699570	7.716190	7.676280	4.760430	4.769720	4.746380
40	0.5	12.84452	12.86891	12.377840	7.716470	7.732850	7.369470	4.764730	4.773450	4.579730
40	0.9	12.88154	12.90236	10.697260	7.766780	7.783970	6.075630	4.774930	4.783880	3.901920
50	0.1	16.64882	16.68181	16.601704	9.780272	9.799840	9.753936	6.188472	6.200008	6.171592
50	0.5	16.68502	16.71798	15.980872	9.787704	9.807208	9.358392	6.206400	6.218376	5.943096
50	0.9	16.73784	16.77114	13.436976	9.823712	9.843216	7.813736	6.232280	6.244280	4.976224

Table 3: Merged Rows with 3 Value States

Combined	Value
A - X	High
	High
A - Y	Low
B - Z	Mid
	Mid
	Mid

Bibliography

Andersson, J., Carling, K., & Mattson, S. (1998). Random ranking of hospitals is unsound. *CHANCE*, *11*(3), 33–39. https://doi.org/doi:10.1080/09332480.1998.10542106

Archakova, I., & Hansen, P. R. (2020). *A canonical representation of block matrices with applications to covariance and correlation matrices.*

Bazylik, S., Mogstad, M., Romano, J. P., Shaikh, A. M., & Wilhelm, D. (2025). Simultaneous confidence regions for ranks. *Journal of Econometrics*. https://doi.org/https://doi.org/10.1016/j.jeconom.2025.106010

Chen, H. J. (1976). Strong consistency and asymptotic unbiasedness of a natural estimator for a ranked parameter. *Sankhyā: The Indian Journal of Statistics, B*, *38*(1), 92–94.

Dudewicz, E. J. (1972). Point estimation of ordered parameters: The general location parameter case. *Tamkang Journal of Mathematics*, *3*(2), 101–114.

Dunn, O. J. (1958). *Estimation of the means of dependent variables.*

Goldstein, H., & Healy, M. J. R. (1995). The graphical presentation of a collection of means. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, *158*(1), 175–177.

Goldstein, H., & Spiegelhalter, D. J. (1996). League tables and their limitations: Statistical issues in comparisons of institutional performance. *Journal of the Royal Statistical Society: Series A (Statistics in Society), 159*(3), 385–443.

Hall, P., & Miller, H. (2009). Using the bootstrap to quantify the authority of an empirical ranking. *The Annals of Statistics*, *37*(6B), 3929–3959.

Klein, M., Wright, T., & Wieczorek, J. (2020). A joint confidence region for an overall ranking of populations. *Journal of the Royal Statistical Society*, 589–606.

Krainski, E. T., Gómez-Rubio, V., Bakka, H., Lenzi, A., Castro-Camilo, D., Simpson, D., Lindgren, F., & Rue, H. (2019). *Advanced spatial modeling with stochastic partial differential equations using r and INLA*. Chapman & Hall/CRC Press.

Lyhagen, J., & Ahlgren, P. (2020). Uncertainty and the ranking of economics journals. *Scientometrics*, (125), 2545–2560. https://doi.org/https://doi.org/10.1007/s11192-020-03681-5

Mogstad, M., Romano, J. P., Shaikh, A. M., & Wilhelm, D. (2024). Inference for ranks with applications to mobility across neighbourhoods and academic achievement across countries. *The Review of Economic Studies*, *91*(1), 476–518.

Mohamad, D. A., Zwet, E. W. van, & Goeman, J. J. (2019). Simultaneous confidence intervals for ranks with application to ranking institutions. *Journal of the International Biometric Society.*

Pulse Asia Research, Inc. (2025). Pulso ng bayan pre-electoral survey: May 6 – 9, 2025 / philippines general report. *Pulse Asia Research, Inc. Official Reports*. https://pulseasia.ph/wp-content/uploads/2025/05/PB-May-2025-Media-Release-and-General-Report.pdf

Ravanilla, N., & Hicken, A. (2023). *When legislators don't bring home the pork: The case of philippine senators.*

Šidák, Z. (1967). *Rectangular confidence regions for the means of multivariate normal distributions.*

Turek, D., & Risser, M. (2022). *bayesNSGP: Bayesian analysis of non-stationary gaussian process models*. https://cran.stat.auckland.ac.nz/web/packages/BayesNSGP/BayesNSGP.pdf

Wright, T. (2025). *Optimal tightening of the KWW joint confidence region for a ranking.*

Zhang, S., Luo, J., Zhu, L., Stinchcomb, D. G., Campbell, D., Carter, G., Gilkesone, S., & Feuerc, E. J. (2013). *Confidence intervals for ranks of age-adjusted rates across states or counties.*

5 Appendices

5.1 True θ Values Used in the Simulation

The actual θ values used in the simulation are shown in Table 4. Figure 2 shows how the spread of the θ varies in the cases considered.

Table 4: True θ values used in the simulation have different levels of variability.

K	Variability	$\boldsymbol{\theta}$
10	low	22.5, 24.9, 27.5, 24.3, 24.8, 23.2, 23, 21.6, 25.1, 19
10	med	21.5, 25.7, 30.5, 24.7, 25.5, 22.6, 22.3, 19.9, 26.1, 15.2
10	high	19.9, 27, 34.9, 25.3, 26.7, 21.9, 21.3, 17.3, 27.7, 9.4
20	low	22.5, 24.9, 27.5, 24.3, 24.8, 23.2, 23, 21.6, 25.1, 19, 21.3, 24.1, 26.5, 26.9, 23.2, 25.8, 21.8, 21.6, 25.6, 21.2
20	med	21.5, 25.7, 30.5, 24.7, 25.5, 22.6, 22.3, 19.9, 26.1, 15.2, 19.3, 24.3, 28.7, 29.3, 22.7, 27.5, 20.1, 19.9, 27.1, 19.1
20	high	19.9, 27, 34.9, 25.3, 26.7, 21.9, 21.3, 17.3, 27.7, 9.4, 16.2, 24.6, 31.9, 33, 21.9, 29.9, 17.7, 17.3, 29.3, 16
30	low	22.5, 24.9, 27.5, 24.3, 24.8, 23.2, 23, 21.6, 25.1, 19, 21.3, 24.1, 26.5, 26.9, 23.2, 25.8, 21.8, 21.6, 25.6, 21.2, 25.7, 22.2, 22.7, 22.6, 21.9, 24.2, 23.6, 22, 24.5, 25

Table 4: True θ values used in the simulation have different levels of variability. *(continued)*

K	Variability	$\boldsymbol{\theta}$
30	med	21.5, 25.7, 30.5, 24.7, 25.5, 22.6, 22.3, 19.9, 26.1, 15.2, 19.3, 24.3, 28.7, 29.3, 22.7, 27.5, 20.1, 19.9, 27.1, 19.1, 27.2, 20.9, 21.8, 21.7, 20.4, 24.5, 23.5, 20.5, 25.1, 26
30	high	19.9, 27, 34.9, 25.3, 26.7, 21.9, 21.3, 17.3, 27.7, 9.4, 16.2, 24.6, 31.9, 33, 21.9, 29.9, 17.7, 17.3, 29.3, 16, 29.4, 19, 20.4, 20.3, 18.2, 25, 23.3, 18.3, 26, 27.4
40	low	22.5, 24.9, 27.5, 24.3, 24.8, 23.2, 23, 21.6, 25.1, 19, 21.3, 24.1, 26.5, 26.9, 23.2, 25.8, 21.8, 21.6, 25.6, 21.2, 25.7, 22.2, 22.7, 22.6, 21.9, 24.2, 23.6, 22, 24.5, 25, 26.5, 25.1, 25.2, 27.1, 25.3, 22.8, 20.6, 21.1, 26, 26
40	med	21.5, 25.7, 30.5, 24.7, 25.5, 22.6, 22.3, 19.9, 26.1, 15.2, 19.3, 24.3, 28.7, 29.3, 22.7, 27.5, 20.1, 19.9, 27.1, 19.1, 27.2, 20.9, 21.8, 21.7, 20.4, 24.5, 23.5, 20.5, 25.1, 26, 28.6, 26.1, 26.4, 29.7, 26.5, 22.1, 18, 18.9, 27.8, 27.8
40	high	19.9, 27, 34.9, 25.3, 26.7, 21.9, 21.3, 17.3, 27.7, 9.4, 16.2, 24.6, 31.9, 33, 21.9, 29.9, 17.7, 17.3, 29.3, 16, 29.4, 19, 20.4, 20.3, 18.2, 25, 23.3, 18.3, 26, 27.4, 31.9, 27.7, 28.1, 33.6, 28.2, 20.9, 14.1, 15.6, 30.5, 30.5
50	low	22.5, 24.9, 27.5, 24.3, 24.8, 23.2, 23, 21.6, 25.1, 19, 21.3, 24.1, 26.5, 26.9, 23.2, 25.8, 21.8, 21.6, 25.6, 21.2, 25.7, 22.2, 22.7, 22.6, 21.9, 24.2, 23.6, 22, 24.5, 25, 26.5, 25.1, 25.2, 27.1, 25.3, 22.8, 20.6, 21.1, 26, 26, 22.1, 21.4, 23.2, 22.3, 22.4, 23.1, 21.4, 24.8, 27, 23.4
50	med	21.5, 25.7, 30.5, 24.7, 25.5, 22.6, 22.3, 19.9, 26.1, 15.2, 19.3, 24.3, 28.7, 29.3, 22.7, 27.5, 20.1, 19.9, 27.1, 19.1, 27.2, 20.9, 21.8, 21.7, 20.4, 24.5, 23.5, 20.5, 25.1, 26, 28.6, 26.1, 26.4, 29.7, 26.5, 22.1, 18, 18.9, 27.8, 27.8, 20.8, 19.5, 22.8, 21.2, 21.3, 22.5, 19.5, 25.5, 29.6, 23

Table 4: True θ values used in the simulation have different levels of variability. *(continued)*

K	Variability	$\boldsymbol{\theta}$
50	high	19.9, 27, 34.9, 25.3, 26.7, 21.9, 21.3, 17.3, 27.7, 9.4, 16.2, 24.6, 31.9, 33, 21.9, 29.9, 17.7, 17.3, 29.3, 16, 29.4, 19, 20.4, 20.3, 18.2, 25, 23.3, 18.3, 26, 27.4, 31.9, 27.7, 28.1, 33.6, 28.2, 20.9, 14.1, 15.6, 30.5, 30.5, 18.8, 16.6, 22.1, 19.4, 19.6, 21.6, 16.7, 26.7, 33.5, 22.5

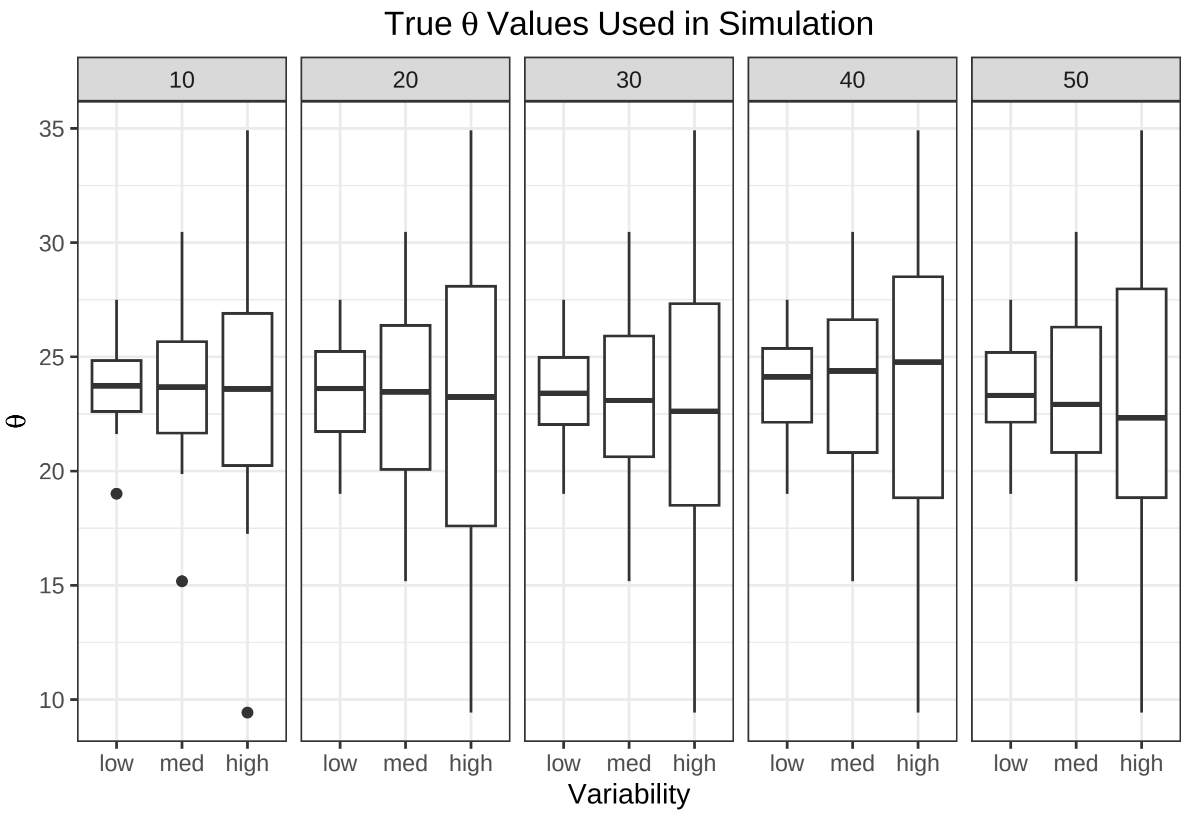


Figure 2: True θ values used in the simulation have different levels of variability.

Codes for algorithm 1

```
get_ci_independent <- function(theta_hat,
                               S,
                               alpha){
  K <- length(theta_hat)
  gamma = 1-(1-alpha)^(1/K)
  z = qnorm(1-gamma/2)
  ci_lower <- theta_hat - z*S
  ci_upper <- theta_hat + z*S
  return(list(
    ci_lower = ci_lower,
    ci_upper = ci_upper
  ))
}

get_ci_bonferroni <- function(theta_hat,
                              S,
                              alpha){
  K <- length(theta_hat)
  z = qnorm(1-(alpha/K)/2)
  ci_lower <- theta_hat - z*S
  ci_upper <- theta_hat + z*S
  return(list(
    ci_lower = ci_lower,
    ci_upper = ci_upper
  ))
}

get_ci_rankbased_asymptotic <- function(B,
                                        theta_hat,
                                        varcovar_matrix,
                                        alpha) {

  K <- length(theta_hat)

  # generate_data <- function(){MASS::mvrnorm(n = 1,
  #                                                   mu = theta_hat,
  #                                                   Sigma = varcovar_matrix)}
  # # line 2 ~~~
  # thetahat_star <- t(replicate(B, generate_data())) # B x K

  # line 2 ~~~
  thetahat_star <- MASS::mvrnorm(n = B,
                                 mu = theta_hat,
                                 Sigma = varcovar_matrix) # B x K
  print("theta_hat")
  print(theta_hat)
  print("thetahat_star")
  print(cat("shape: ", dim(thetahat_star)))
  print(thetahat_star[c(1:3),])

  # sorted counterpart line 2
  sorted_thetahat_star <- t(apply(thetahat_star, 1, sort))
  print("sorted_thetahat_star")
  print(cat("shape: ", dim(thetahat_star)))
  print(sorted_thetahat_star[c(1:3),])

  variance_vector <- diag(varcovar_matrix)
  print("Variance vec:")
  print(variance_vector)
  print("Variance mat:")
  print(matrix(variance_vector, B, K, byrow = TRUE)[c(1:3),])
  print("thetahat_star:")
  print(thetahat_star[c(1:3),])
  print("thetahat_star squared:")
  print((thetahat_star^2)[c(1:3),])
  print("minuend:")
```

```
# line 3 ~~~
minuend <- thetahat_star^2 + matrix(variance_vector, B, K, byrow = TRUE)
print(minuend[c(1:3),])
print("sorted minuend:")
print(t(apply(minuend, 1, sort))[c(1:3),])
print("sorted_thetahat_star^2")
print((sorted_thetahat_star^2)[c(1:3),])
  #(matrix(rep(variance_vector, each = B),nrow = B, byrow = FALSE))
print("their difference")
print((t(apply(minuend, 1, sort)) - sorted_thetahat_star^2)[c(1:3),])

sigma_hat_star <- sqrt(
  t(apply(minuend, 1, sort)) - sorted_thetahat_star^2
  )
# step 1c ====================================
print("sigma_hat_star")
print(sigma_hat_star)

sorted_theta_hat <- sort(theta_hat)
print("sorted theta hat")
print(sorted_theta_hat)
# print((matrix(rep(sorted_theta_hat, each = B),
#                 nrow = B, byrow = FALSE)))
# t_star <- apply(
#   abs(
#     (
#       sorted_thetahat_star - matrix(sorted_theta_hat, B, K, byrow = TRUE)
#       # (matrix(rep(sorted_theta_hat, each = B),
#       #                                   nrow = B, byrow = FALSE))
#       )/sigma_hat_star
```

```r
#     ),
#   1,
#   max)

# line 4 ~~~
compute_max <- function(b) {
  t_b <- max(abs(
    (sorted_thetahat_star[b, ] - sorted_theta_hat) /
      sigma_hat_star[b,]
  ))
  return(t_b)
}
print(sorted_thetahat_star[1, ])
print(sorted_theta_hat)
print(sigma_hat_star[1,])
print("compute max for b = 1")
print(sigma_hat_star[1,])
print(sorted_thetahat_star[1, ] - sorted_theta_hat)
print((sorted_thetahat_star[1, ] - sorted_theta_hat)/sigma_hat_star[1,])
print(max(abs(
  (sorted_thetahat_star[1, ] - sorted_theta_hat) /
    sigma_hat_star[1,]
)))
print(compute_max(1))

t_star <- sapply(1:B, compute_max)

# line 6 ~~~
t_hat <- quantile(t_star, probs = 1 - alpha)

# line 7 ~~~
```

```
  sigma_hat <- sqrt(
    sort(theta_hat^2 + variance_vector) - sorted_theta_hat^2
    )
  ci_lower <- sorted_theta_hat - t_hat*sigma_hat
  ci_upper <- sorted_theta_hat + t_hat*sigma_hat

  return(list(
    ci_lower = ci_lower,
    ci_upper = ci_upper
  ))
}

get_ci_rankbased_level2bs <- function(B,
                                      C,
                                      theta_hat,
                                      varcovar_matrix,
                                      alpha) {
  print("ci_rankbased_level2bs ==================================
  K <- length(theta_hat)
  sorted_theta_hat <- sort(theta_hat)

  # generate_data <- function(){MASS::mvrnorm(n = 1,
  #                                           mu = theta_hat,
  #                                           Sigma = varcovar_matrix)}
  # # line 2 ~~~
  # thetahat_star <- t(replicate(B, generate_data())) # B x K

  # line 2 ~~~
  thetahat_star <- MASS::mvrnorm(n = B,
                                 mu = theta_hat,
                                 Sigma = varcovar_matrix) # B x K
```

```r
# print("theta_hat")
# print(theta_hat)
# print("thetahat_star")
# print(cat("shape: ", dim(thetahat_star)))
# print(thetahat_star)

# sorted counterpart line 2
sorted_thetahat_star <- t(apply(thetahat_star, 1, sort)) # B x K
print("sorted_thetahat_star")
print(cat("sorted_shape: ", dim(thetahat_star)))
print(sorted_thetahat_star)

# for each b row in level 1, generate a K x C matrix,
generate_level2_data <- function(mu){MASS::mvrnorm(n = 1,
                                                   mu,
                                                   Sigma = varcovar_matrix)}

# line 4 ~~~
thetahat_double_star <- # list of length B, each a K x C matrix
  apply(thetahat_star,
        1,
        function(thetahat_b) {replicate(C, # expected is a K x C matrix
                                        generate_level2_data(mu=thetahat_b))},
        simplify = FALSE
        )

# print("Bootstrap 1")
# print(thetahat_star[5,])
# print("thetahat_double_star")
# # print(length(thetahat_double_star))
# print(dim(thetahat_double_star[[5]]))
```

```
# print(thetahat_double_star[[5]])

# sorted counterpart line 4
sorted_thetahat_double_star <- lapply(thetahat_double_star,
                                      function(x)apply(x, 2, sort))
# print(dim(sorted_thetahat_double_star[[5]]))
# print(sorted_thetahat_double_star[[5]])

# line 5 ~~~
# for each matrix b and for each row k
# K x C matrix arg
compute_sigma_hat <- function(mat, n){
  # outputs a vector of length K
  apply(mat,
        1, # done for each k
        function(row){
          sqrt(
            sum((row - mean(row))^2)/(n-1))
          })}

# output must be B lists, each a vector of length K
sigma_hat_star <- lapply(sorted_thetahat_double_star, # K by C
                         function(x) compute_sigma_hat(x, C)
                         )
# print(length(sigma_hat_star)) # = B
print(length(sorted_thetahat_star[5, ])) # = K
print(length(sorted_theta_hat)) # = K
print(length(sigma_hat_star[[5]])) # = K

compute_max <- function(b) {
  t_b <- abs(
```

```
      (sorted_thetahat_star[b, ] - sorted_theta_hat) /
        sigma_hat_star[[b]]
    )
    max(t_b)
  }

  # line 7 ~~~
  t_star <- sapply(1:B, compute_max)
  # print(cat("t_star: ",t_star))
  # print(cat("length t_star: ",length(t_star))) # = B

  # line 9 ~~~
  t_hat <- quantile(t_star, probs = 1 - alpha)

  # line 10 ~~~
  # print("transposed")
  # print(dim(t(sorted_thetahat_star)))
  sigma_hat <- compute_sigma_hat(t(sorted_thetahat_star),B)
  # print('sigma_hat')
  # print(sigma_hat)
  # print(length(sigma_hat))

  ci_lower <- sorted_theta_hat - t_hat*sigma_hat
  ci_upper <- sorted_theta_hat + t_hat*sigma_hat

  return(list(
    ci_lower = ci_lower,
    ci_upper = ci_upper
  ))
}
```

```r
get_ci_nonrankbased <- function(B,
                                theta_hat,
                                alpha,
                                varcovar_matrix) {
  print("ci_nonrankbased ================================
  K <- length(theta_hat)
  # step 1a ===================================
  # generate_data <- function(){MASS::mvrnorm(n = 1,
  #                                           mu = theta_hat,
  #                                           Sigma = varcovar_matrix)}
  # # line 2 ~~~
  # thetahat_star <- t(replicate(B, generate_data())) # B x K



  # line 2 ~~~
  thetahat_star <- MASS::mvrnorm(n = B,
                                 mu = theta_hat,
                                 Sigma = varcovar_matrix) # B x K


  # line 3 ~~~
  t_star <- apply(thetahat_star,
                  1,
                  function(x) max(
                    abs(
                    (x - theta_hat) / sqrt(diag(varcovar_matrix))
                    )
                    ))
  # line 5 ~~~
  t_hat <- quantile(t_star, probs = 1 - alpha)
  # step 3 ====================================
  ci_lower <- theta_hat - t_hat*sqrt(diag(varcovar_matrix))
```

```
  ci_upper <- theta_hat + t_hat*sqrt(diag(varcovar_matrix))
  return(list(
    ci_lower = ci_lower,
    ci_upper = ci_upper
  ))
}
```

Codes for algorithm 2

```
 source("R/compute_ci.R")
library("doRNG")

get_ranks <- function(k, tuple_list){
  Lambda_lk <- which(
    tuple_list[,2]<=tuple_list[k,1])
  Lambda_lk <- Lambda_lk[Lambda_lk != k]
  Lambda_Ok <- which(
    tuple_list[,2]>tuple_list[k,1] & tuple_list[k,2] > tuple_list[,1])
  Lambda_Ok <- Lambda_Ok[Lambda_Ok != k]
  ranks <- seq(
    length(unique(Lambda_lk)) + 1,
    length(unique(Lambda_lk)) + length(unique(Lambda_Ok)) + 1,
    1
  )
  return(list(
    ranks = ranks,
    Lambda_Ok = Lambda_Ok
  ))
}
```

```
get_t1 <- function(v) mean(v)

get_t2 <- function(v) prod(v)^(1/length(v))

get_t3 <- function(v) {
  1 - ((length(v)+sum(v))/(length(v)^2))
}

get_coverage <- function(ci_lower,
                         ci_upper,
                         true_theta,
                         rank_theta=TRUE) {

  if (rank_theta) {
    return(all(ci_lower<=sort(true_theta)) & all(sort(true_theta)<=ci_upper))
  } else {
    return(all(ci_lower<=true_theta) & all(true_theta<=ci_upper))
  }

}

implement_algorithm2 <- function(
    true_theta,
    K,
    reps=300, # step 4
    B=200,
    alpha= 0.10,
    C=100,
    varcovar_matrix){
  foreach(iter = 1:reps,
          .combine = rbind,
          .packages = c("foreach", "arrow", "MASS"),
          .export = c("get_ci_nonrankbased",
                      "get_ci_rankbased_asymptotic",
                      "get_ci_rankbased_level2bs",
                      "get_ci_independent",
                      "get_ci_bonferroni",
                      "get_ranks", "get_coverage",
                      "get_t1", "get_t2", "get_t3")
  ) %do% {
    print(true_theta)
    # step 1 =======
    theta_hat <- mvrnorm(n = 1,
                         mu = true_theta,
                         Sigma = varcovar_matrix)

    # print("true theta")
    # print(true_theta)
    #
    # print("sample theta")
    # print(theta_hat)



    # print(varcovar_matrix)

    # step 2 =======
    S <- sqrt(diag(varcovar_matrix))

    ci_methods <- list(
      nonrankbased = function() get_ci_nonrankbased(B,
                                                    theta_hat,
                                                    alpha,
                                                   varcovar_matrix),
    rankbased_asymptotic = function() get_ci_rankbased_asymptotic(B,
                                                                    theta_hat,
                                                                    varcovar_matrix,
                                                                    alpha),
    rankbased_level2bs = function() get_ci_rankbased_level2bs(B,
                                                                C,
                                                                theta_hat,
                                                                varcovar_matrix,
                                                                alpha),
    independent  = function() get_ci_independent(theta_hat, S, alpha),
    bonferroni   = function() get_ci_bonferroni(theta_hat, S, alpha)
  )
  # print("REACHED 0 ======================================
  ci_results <- lapply(ci_methods, function(f) f())
  # print("REACHED 1 ======================================

  coverages <- list(
      nonrankbased = get_coverage(
        ci_lower   = ci_results[[1]]$ci_lower,
        ci_upper   = ci_results[[1]]$ci_upper,
        true_theta = true_theta,
        rank_theta = FALSE
      ),
      rankbased_asymptotic = get_coverage(
        ci_lower   = ci_results[[2]]$ci_lower,
        ci_upper   = ci_results[[2]]$ci_upper,
        true_theta = true_theta,
        rank_theta = TRUE
      ),
      rankbased_level2bs = get_coverage(
```

```r
      ci_lower   = ci_results[[3]]$ci_lower,
      ci_upper   = ci_results[[3]]$ci_upper,
      true_theta = true_theta,
      rank_theta = TRUE
    ),
    independent = get_coverage(
      ci_lower   = ci_results[[4]]$ci_lower,
      ci_upper   = ci_results[[4]]$ci_upper,
      true_theta = true_theta,
      rank_theta = FALSE
    ),
    bonferroni = get_coverage(
      ci_lower   = ci_results[[5]]$ci_lower,
      ci_upper   = ci_results[[5]]$ci_upper,
      true_theta = true_theta,
      rank_theta = FALSE
    )
)

# print(coverages)

# coverages <- lapply(ci_results, function(res) {
#   get_coverage(
#     ci_lower   = res$ci_lower,
#     ci_upper   = res$ci_upper,
#     true_theta = true_theta
#   )
# })

# print("REACHED 2 ================================
process_ci_result <- function(result, K) {
  tuple_list <- t(apply(
    data.frame(
      ci_lower = result$ci_lower,
      ci_upper = result$ci_upper
    ),
    1,
    function(row) as.numeric(row)
  ))

  rank_range_length <- sapply(1:K, function(x)
    length(get_ranks(x, tuple_list)$ranks)
  )



  list(
    t1 = get_t1(rank_range_length),
    t2 = get_t2(rank_range_length),
    t3 = get_t3(rank_range_length)
  )
}

# print("REACHED 3 ====================================
processed <- lapply(ci_results, process_ci_result, K = K)
# print("REACHED 4 ====================================
data.frame(
  t1_nonrankbased = processed$nonrankbased$t1,
  t2_nonrankbased = processed$nonrankbased$t2,
  t3_nonrankbased = processed$nonrankbased$t3,
  coverage_nonrankbased = coverages$nonrankbased,
  t1_rankbased_asymptotic = processed$rankbased_asymptotic$t1,
   t2_rankbased_asymptotic = processed$rankbased_asymptotic$t2,
   t3_rankbased_asymptotic = processed$rankbased_asymptotic$t3,
   coverage_rankbased_asymptotic = coverages$rankbased_asymptotic,
   t1_rankbased_level2bs = processed$rankbased_level2bs$t1,
   t2_rankbased_level2bs = processed$rankbased_level2bs$t2,
   t3_rankbased_level2bs = processed$rankbased_level2bs$t3,
   coverage_rankbased_level2bs = coverages$rankbased_level2bs,
   t1_independent = processed$independent$t1,
   t2_independent = processed$independent$t2,
   t3_independent = processed$independent$t3,
   coverage_independent = coverages$independent,
   t1_bonferroni = processed$bonferroni$t1,
   t2_bonferroni = processed$bonferroni$t2,
   t3_bonferroni = processed$bonferroni$t3,
   coverage_bonferroni = coverages$bonferroni
   )
  # print("REACHED 5 ======================================
 }
}



# algo2_parametric <- function(
#    true_theta,
#    K,
#    reps = 5, # step 4
#    B=100,
#    alpha= 0.10,
#    S){
#  foreach(iter = 1:reps,
#       .combine = rbind,
#       .packages = c("foreach", "arrow", "MASS"),
#           .export = c("get_ci_parametric","get_ranks", "get_coverage",
#                       "get_t1", "get_t2", "get_t3")
#   ) %dorng% {
#
#     # step 1 ========
#     theta_hat <- rnorm(
#       n    = K,
#       mean = true_theta,
#       sd   = S
#     )
#
#     # step 2 ========
#     result <- get_parametric_ci(B,
#                                 theta_hat,
#                                 S,
#                                 alpha)
#
#     # step 3 ========
#     sorted_true_theta <- sort(true_theta)
#     coverage <- get_coverage(ci_lower = result$ci_lower,
#                              ci_upper = result$ci_upper,
#                              true_theta = sorted_true_theta)
#
#     tuple_list <- t(apply(
#       data.frame(ci_lower = result$ci_lower,
#                  ci_upper = result$ci_upper), 1, function(row) as.numeric(row)))
#     rank_range_length <- sapply(1:K, function(x) length(
#       get_ranks(x, tuple_list)$ranks))
#     t1 <- get_t1(rank_range_length)
#     t2 <- get_t2(rank_range_length)
#     t3 <- get_t3(rank_range_length)
```

```
#
#     data.frame(
#       t1_parametric = t1,
#       t2_parametric = t2,
#       t3_parametric = t3,
#       coverage_parametric = coverage
#     )
#   }
# }
#
```

Codes for simulation

```
 #3:37PM
print(getwd())
source("R/compute_metrics.R")
library(doParallel)
library(foreach)
library(tictoc)
mean <- 23.8
df <- readRDS("data/mean_travel_time_ranking_2011.rds")
cl=parallel::makeCluster(15)
registerDoParallel(cl)

sds <- c(2, 3.6, 6)
Ks <- c(10)
#corrs <- c(0.1,0.5,0.9)
alphas <- c(0.05)#c(0.05, 0.1, 0.15, 0.2)
```

```
blocks <- list(1:3, 4:6, 7:10)          # 3 blocks of sizes 3, 3, 4
within_corrs <- c(0.1, 0.5, 0.9)        # one correlation per block
between_corr <- 0.1                     # correlation between blocks
corr_matrix <- matrix(between_corr, Ks, Ks)
for (i in seq_along(blocks)) {
  block <- blocks[[i]]
  corr_matrix[block, block] <- within_corrs[i]
}
diag(corr_matrix) <- 1



for (sd in sds) {
  for (K in Ks) {
    set.seed(123974)
    true_theta <- rnorm(K, mean, sd)
    SE <- df$S[1:K]

    for (alpha in alphas) {

      tic()
      # for (corr in corrs) {
        #corr_matrix <- (1 - corr) * diag(K) + corr * matrix(1, K, K)
        variance_vector <- SE^2
        delta <- diag(variance_vector)
        varcovar_matrix <- delta^(1/2) %*% corr_matrix %*% delta^(1/2)

        cat("=============================================\
        cat("SIMULATION SETTINGS ~ K:", K,"corr:", "block","sd:", sd, "corr:", "block"
        cat("\n")
        cat("SORTED TRUE THETA", sort(true_theta))
        cat("\n")
        case_start <- Sys.time()

        tic()
        coverage_output_df <- implement_algorithm2(
          true_theta,
          K,
          reps = 5000,
          B = 600,
          alpha=alpha,
          C = 300,
          varcovar_matrix = varcovar_matrix)
        toc()

        # saveRDS(coverage_output_df,  paste0("output/final_coverage_",
        saveRDS(coverage_output_df,  paste0("output/blockcorr_coverage_",
                                            K,"_", sd, "_", "block", "_",
                                            alpha, ".rds"))
        case_end <- Sys.time()
        case_runtime <- as.numeric(difftime(case_end, case_start, units = "mins"))
        cat("Finished at:", format(case_end, "%I:%M %p"), "\n")
        cat("Runtime for this case:", round(case_runtime, 2), "minutes\n")
        cat("==========================================\
      # }
    }
  }
}

stopCluster(cl)
#
# param_grid <- expand.grid(K = Ks, sd = sds, corr = corrs, alpha = alphas)
#
```

```
# results <- do.call(rbind, lapply(seq_len(nrow(param_grid)), function(i) {
#   K <- param_grid$K[i]
#   sd <- param_grid$sd[i]
#   corr <- param_grid$corr[i]
#   alpha <- param_grid$alpha[i]
#
#   a <- readRDS(paste0("output/TEST_coverage_",
#                       K, "_", sd, "_", corr, "_", alpha, ".rds"))
#
#   data.frame(
#     K = K,sd = sd,corr = corr,alpha = alpha,
#
#     Cov_nonrankbased = mean(a$coverage_nonrankbased),
#     Cov_rankbased_asymptotic = mean(a$coverage_rankbased_asymptotic),
#     Cov_rankbased_level2bs = mean(a$coverage_rankbased_level2bs),
#     Cov_independent = mean(a$coverage_independent),
#     Cov_bonferroni = mean(a$coverage_bonferroni),
#
#     T1_nonrankbased = mean(a$t1_nonrankbased),
#     T1_rankbased_asymptotic = mean(a$t1_rankbased_asymptotic),
#     T1_rankbased_level2bs = mean(a$t1_rankbased_level2bs),
#     T1_independent = mean(a$t1_independent),
#     T1_bonferroni = mean(a$t1_bonferroni),
#
#     T2_nonrankbased = mean(a$t2_nonrankbased),
#     T2_rankbased_asymptotic = mean(a$t2_rankbased_asymptotic),
#     T2_rankbased_level2bs = mean(a$t2_rankbased_level2bs),
#     T2_independent = mean(a$t2_independent),
#     T2_bonferroni = mean(a$t2_bonferroni),
#
#     T3_nonrankbased = mean(a$t3_nonrankbased),
#      T3_rankbased_asymptotic = mean(a$t3_rankbased_asymptotic),
#      T3_rankbased_level2bs = mean(a$t3_rankbased_level2bs),
#      T3_independent = mean(a$t3_independent),
#      T3_bonferroni = mean(a$t3_bonferroni)
#   )
# }))
#
# # param_grid <- expand.grid(K = Ks, sd = sds, alpha = alphas)
# #
# # results1 <- do.call(rbind, lapply(seq_len(nrow(param_grid)), function(i) {
# #   K <- param_grid$K[i]
# #   sd <- param_grid$sd[i]
# #   alpha <- param_grid$alpha[i]
# #
# #   a <- readRDS(paste0("output/coverage_parametric_",
# #                       K, "_", sd, "_", alpha, ".rds"))
# #
# #   data.frame(
# #     K = K,
# #     sd = sd,
# #     alpha = alpha,
# #     Cov_parametric = mean(a$coverage_parametric),
# #     T1_parametric = mean(a$t1_parametric),
# #     T2_parametric = mean(a$t2_parametric),
# #     T3_parametric = mean(a$t3_parametric)
# #   )
# # }))
# #
# # save(results, results1, file = "simulation_results.RData")
```