**2.a) According to their experimental results, How successful is IPS in addressing bias in click data? In the presence of high degrees of bias, how the performance of their model could be improved?**

The average rank of the IPS ranker is roughly 0.5 lower than the naive ranker from a presentation bias severity of 0.65 and onwards.

When both rankers are given 5 times more training data, the difference in performance as measured by average rank increases as presentation bias severity increases. Therefore, their model could be improved by increasing the amount of training data.

**2.b) One of the implicit biases that are ignored as a result of their IPS formulation is the bias caused by implicitly treating non-clicked items as not relevant. Discuss when this implicit bias is problematic?**

The IPS formulation only accounts for the position bias present in items that are observed and subsequently clicked on. The IPS formulation does not account for the position bias present in items that are not observed. Therefore, this implicit bias becomes problematic when some (relevant) items are not observed at all. Because users won’t click on things you didn’t show them, so the position of the documents matter.

**2.c) Propose a simple method to correct for the implicit bias of non-clicks.**

To correct for the implicit bias of non-clicks, we must make sure that each item must be observed at least once. This can be implemented by only including items that have been ranked above the item that was clicked last in a session. The assumption made here is that every item above the item last clicked, has been observed, which will be true in most cases.

**3.a)** Explain the LTR loss function in Thorsten et al. [1] that can be unbiased using the IPS formula and discuss what is the property of that loss function that allows for IPS correction.

The loss function used by Thorsten is the sum of ranks of relevant results:

\Delta(y|x\_i,r\_i) = \sum\_{y\in Y} rank(y|Y) \cdot r\_i(y)

The property of loss function: irrelevant documents (ri=0) do not contribute to loss. For the loss functions in the assignment, we need to correct them in a way that they do not depend on r\_i(y)=0. There is position bias in Clicks since users won’t click on things you didn’t show them. Documents at higher rank are more likely to be clicked and those at lower rank will never get clicked even though they can be relevant but not observed. That is why a non-clicked document does not mean irrelevant so the idea is that we should not take the irrelevant documents into account, meaning the irrelevant documents (ri=0) should not contribute to the sum of loss function. IPS can also correct position bias due to re-weighting and the intuition is that if a document is very low down in ranking list and hence, has less chance of being observed/examined and if a few times it get observed and got clicked/relevant, we should give more weight to that to correct to correction bias.

3b) Try to provide an IPS corrected formula for each of the three LTR loss functions that you have seen and implemented in the computer assignment. If a loss function cannot be adapted in the IPS formula, discuss the possible reasons.

**Pointwise:**

\Delta\_{pw} = 1/|Y| \sum\_{y\in Y} (s(y) - r\_i(y))^2

With set of documents Y, individual document y, score s(y) and relevance r\_i(y) = {0,1}. When r\_i(y) = 0, we get a contribution to the loss of s(y)^2/|y|, whereas we would like the loss function to have zero contribution. The easiest way is to modify the loss function is:

\Delta = 1/|Y| \sum\_{y\in Y, r\_i(y)=1} (s(y) - r\_i(y))^2,

where the sum runs over all documents that are relevant. This modified loss function is minimized when all relevant documents are predicted to be relevant (i.e. s(y) = 1 when r\_i(y)=1), so that it is still a good loss function. IPS in this case is given by

IPS = 1/|Y| \sum\_{y\in Y, r\_i(y)=1} (s(y) - r\_i(y))^2 / Q(o\_i(y)=1|x\_i,y\_i,r\_i)

**Pairwise loss:**

C = \sum\_{ij\in P} \frac{1}{2}(1- S\_{ij})\sigma(s\_i - s\_j) + \log(1+ e^{-\sigma(s\_i - s\_j)})

where S\_{ij} is 1 when document i is more relevant than document j, equal to 0 for the opposite and equal to 1/2 when they are equally relevant. sigma is a parameter that was used to determine the shape of P\_ij = logistic (\sigma (s(i)-s(j))) in the original RankNet, s\_i are the scores of documents to queries as predicted by the model, and the sum runs over all pairs of documents that are used during training.

To correct this loss function, a first guess would be to first fix document D\_i, and correct the partial loss that consists of running over the document index j. Next we repeat with fixing D\_j and running the sum over i. Note that for this approach to yield correct results, this assumes that Thorstens IPS approach naively extends to loss functions involving more than one document.

Next subtracting \frac{1}{2}(1- S\_{ij})\sigma(s\_i - s\_j) when document j is irrelevant. This implies that S\_{ij} is either 1 (doc i relevant) or 1/2 (doc i irrelevant). This term is zero when doc i is relevant.

Next, we subtract \frac{1}{2}(1- S\_{ij})\sigma(s\_i - s\_j) when document i is irrelevant while also correcting the previous correction, i.e. we add \frac{1}{2}(1- S\_{ij})\sigma(s\_i - s\_j) = 1/4(\sigma(s\_i - s\_j)) when both docs are irrelevant. In summary:

i relevant, j relevant: 0

i relevant, j irrelevant: 0

j relevant, i irrelevant: -\frac{1}{2}(\sigma(s\_i - s\_j)

both irrelant: + 1/4(\sigma(s\_i - s\_j))

Note that this sums to zero, so this loss function does not need correction! The corrected IPS function will be: IPS = \sum\_{ij} C\_{ij} / (Q\_i Q\_j).

**Listwise:**

we have the gradients of the loss function: \sum\_{j \in D} \lambda\_{ij} \cdot |\bigtriangleup NDCG (i,j)|

where lambda\_{ij} is given by: \sigma \bigg( \frac{1}{2}(1 - S\_{ij}) - \frac{1}{1 + e^{\sigma(s\_i -s\_j))}} \bigg)

The \bigtriangleup NDCG (i,j)| denotes the absolute difference of NDGC when swapping ranks at i and j. Since we are dealing with derivatives of a loss function with respect to score s(i), it is tempting to go with the previous approach and obtain derivatives of IPS. Since the expectation value is over a finite set and the marginal probabilities Q do not depend on the scores s(i), the expectation value of derivatives of IPS is equal to the derivative of optimized loss and it remains bias-protected.

NDGC is given by DCG / IDCG. The corrected gradients of loss can then be obtained by dC\_corr / ds(i) = \sum\_{j \in D, r\_j = 1} \lambda\_{ij} \cdot |\bigtriangleup NDCG (i,j)|. The gradient of IPS will be:

dIPS/ds(i) = \sum\_{j \in D, r\_j = 1} \lambda\_{ij} . |\bigtriangleup NDCG (i,j)| / Q(o\_i(y)=1|x\_i,y\_i,r\_i)

**4.a**

IPS can be extended to the graded user feedback by in the following ways:

**RandTop-N:** RandTop-N is a way to estimate the position bias of clicks from the user. The way it works is that the user gets presented a randomized order of the top N results multiple times. This way, the position bias of every position compared to the other positions can be observed. The downside to this is that users will get random top results presented when they are looking for the most relevant result to their query. **Recommendation (1)**

**RandPair:** RandPair works in the same way as RandTop-N with the difference that a pivot rank K is chosen and only a swap with another random document is performed at this pivot rank. **Recommendation (3)**

**Interventional Sets:** Exploit inherit “randomness” in data coming from multiple rankers. For example, with A/B testing different ranking methods on the user. This means it does almost random changes. In this case you have to account for the fact that it is not uniformly random. **Recommendation (4)**

**4.b**

Practitioners of counterfactual LTR systems (like IPS) will run into the problem of high variance. High variance can be due to many factors: not enough training data, extreme position bias, and very small propensity or large amounts of noisy clicks on documents with small propensity. However, the usual suspect is one or a few data points with an extremely small propensity that overpowers the rest of the data set. With IPS estimators, we are dividing by examination probability. If this becomes small (close to 0), it will blow up the estimate generating an extremely large number.

A typical solution to high variance in IRS is to apply property clipping. With property clipping, you bound the propensity to prevent any single sample from overpowering the rest of the data sets. This solution trades off bias and variance: It will introduce some amount of bias but can substantially reduce variance.

**5)**

The idea behind TDM is the same as TDI, however TDI takes only 2 rankers (A and B) together and creates an interleave ranking list based on both rankings while TDM allows combining multiple rankers instead of two. Each ranker will have a corresponding team. First, randomly pick up one of the rankers (say, A) among the rankers. Insert the highest rank document in ranker A into the team list of ranker A and the interleave list. Once the document is in the interleave list, it is unavailable for the remaining rankers, so we will go to the next document (position p+1) in the next ranker and place it in the interleave list. Rankers are credited for each click on an item drawn from the corresponding ranker. Multileave rankers address the issue when there are way too many candidate rankers that make it impossible to compare all of them to each other using interleaving.

For TDI there are two vectors in feature space, while for TDM there are multiple vectors in feature space. TDM is a lot more costly, computationally speaking, but it converges to the (optimal) solution much faster., resulting in an improved user experience. By comparing multiple rankers, TDM has an increased chance of finding an improvement.

Another difference between TDI and TDM is that in case when the number of rankers is larger than the number of slots in the interleaved list which can happen in TDM, some teams may not be represented at all.

**6)** How does MGD perform w.r.t ranking performance and convergence w.r.t DBGD and why do these improvements occur?

MGD outperforms DBGD in terms of ranking performance because MGD can handle multiple candidate rankers due to multi-leaving and by comparing multiple candidates at each iteration the probability of finding a better candidate ranker than the current best is expected to increase. Furthermore, adding more rankers to the comparison increases the expected value of the resulting ranker, since the candidate rankers will also compete with each other. Experiments show that system with more candidate rankers can also counter the noise introduced by the click model hence better improve the current best one, for example when number of rankers is 20 ( n = 20), MGD obtain an NDCG value on informational feedback that is close to the converged performance on perfect feedback.

Regarding convergence, MGD does not converge to a better global optimum than DBGD. The difference between the MGD and DBGD are big at the beginning then decreases over time, until they converge to a similar level of performance. However, MGD reached the convergence level at a much faster time.

**7)**

The two major advantages of PDGD over MGD are:

1. MGD is strongly affected by noise and position bias while PDGD is robust to these effects.

2. MGD is unable to reach the optimal performance in an ideal setting while PDGD is capable of reaching optimal performance in an optimal setting.

**8)**

The ranking performance of online approaches at convergence is far worse than the performance of offline approaches. This remains true when bias and noise are completely removed from the click logs. The worse performance obviously leads to a worse user experience.

Using a counterfactual approach, to estimate the position bias, a form of randomization is necessary. This negatively impacts the user experience. To keep this negative effect to a minimum while still estimating the position bias, either the RandPair or Interventional Sets algorithm can be used. Both algorithms result in the user sometimes being presented with a random item, which is more likely to be irrelevant than an item presented by the model that is being improved.

Using an online approach, a user is negatively impacted when providing a query which has been seen only very little, or never before, because the bandit gradient descent methods do not generalize to other queries. This is because no ranking model is learned, rather an (optimal) ranking is learned for each query.

9.a) Counterfactual LTR assumes a relation between a click on a document and the document's relevance. What is this assumption?

A click is assumed to only occur with probability P(c=1 | y(d) = 1) if and only if a document is examined and relevant documents are more likely to be clicked. Using Bayes theorem, the probability that a clicked document is relevant is given by: P(y=1|c=1) = P(c=1|y=1) \* P(y=1) / P(c=1) where P(y=1) is the probability that a document is relevant and P(c=1) is the probability that a document is clicked. The ratio P(y=1) / P(c=1) can be approximated to be the ratio # relevant documents / # clicked documents. Note that unclicked documents can still be relevant due to either being unobserved or due to P(c=0 | y=1) != 0

9. b

1. In a situation where there is a lot of noise. Users randomly click some of the documents that are retrieved without considering the relevance.

2. In the situation of position bias, for example if a non-relevant document is ranked on the first position on a query retrieval or a relevant document is ranked at the bottom of the result list. The relation between the relevance and the number of clicks now changes because the document will receive a lot of clicks even though it is not relevant.