We deeply thank reviewers for their insightful and constructive comments, which greatly helped us improve the work. As all 3 reviewers pointed out, **the main novelty and contribution of this work are that we found that an adapted interior point algorithm can be much more efficient than traditionally expected both theoretically and computationally in solving the Wasserstein Barycenter problem. The creativity lies in reducing the complexity of solving the normal equations in each iteration by fully exploiting its matrix's block diagonal structure. This finding is potentially useful in further speeding up many state-of-art interior point algorithms that mainly focus on reducing the iteration numbers.** Compared to popular regularization approaches such as Sinkhorn-based methods, this algorithm may gain a good balance between accuracy and speed.

Some common issues from reviewers are addressed here: we conducted more experiments to compare with Solomon et al.'s convolutional Sinkhorn algorithm(will keep trying more experiments!), and the comparison result is similar to that of IBP. For reproducibility, codes are on GitLab now. We will post the link in the revision.

Response to comments of Reviewer 1: Thank you for very insightful inputs!

Q1: Our methods for solving normal equations is parallelizable on the N axis, too. In SLRM/DLRM, the major cost is computing A_{ii}^{-1} for each $i = 1, \ldots, N$ independently(in Algorithm 2 line 1), which can be done in parallel for all i's.

Q2: It's a crucial observation! Thanks for raising it. In Thm. 4.3: the complexity of one iteration in our IPM is linear w.r.t. N and m (or m_i). This is because we directly build up $\bar{A}D\bar{A}^\top$ and the complexity of one IPM iteration is the same as the complexity of SLRM/DLRM. We'll explain it more in the paper. Thus, in figure 4, if we fix 2 parameters among $N,\ m,\ m_i$ and change the other one, it'll show a linear rate. When varying m, the pattern is similar to figure 4.

Q3: We use 2 columns to demonstrate the linear relation between runtime and N(m'). We regard Gurobi's solution accurate and omit it in the middle 2 columns. In the right column, we compare with Gurobi to display MAAIPM's low memory occupancy advantages over traditional second order methods, though it does take more memory than first order methods by nature. It is great advice to display all methods. We'll add it in the paper.

Q4: Thanks for this important advice! We have tested the method and it indeed improved the results from Sinkhorn's side! We will add it in the paper.

Improvement suggestions: Thanks for detailed instructions! We have followed each point carefully and revised the paper accordingly and will include them in the revision.

Response to Reviewer 2: Thanks! Your concerns helped us strengthen some important issues of the paper!

Q1: "However, I am concerned...run time for the previous best interior point algorithm... reported to be $O(N^3m^4)$..."

ANS: Thanks for raising up this issue! It's the most important theoretical part and we should have explained it better(will revise!). The paper focuses on reducing the complexity of solving the normal equation in one iteration. For a standard LP, let $\bar{m},\ \bar{n}$ be the number of constraints and variables. Then computing ADA^T requires $O(\bar{m}^2\bar{n})$ flops, while solving a system with the coefficient matrix ADA^T requires $O(\bar{m}^3)$ flops. So a single IPM iteration takes $O(\bar{m}^2\bar{n})$. If $m = m_i$, the LP has $Nm^2 + m$ variables and $2Nm + 1$ constraints. Thus $O(N^3m^4)$ directly comes from formulating and solving the normal equations in one iteration via direct matrix multiplication and Cholesky decomposition. Possibly $O(N^3m^4)$ can be improved by techniques such as fast matrix multiplication, but not too much. We'll add more related results(especially overall complexity) and look into the valuable references you provide.

Q2: "I am confused why the NeurIPS paper by Cuturi is cited for Iterative Bregman Projection....."

ANS: Thank you for pointing this out! It is a mistake and we'll correct it.

Q3: "I think the evaluation of SLRM and DLRM must be done against more recent solvers for normal equations....."

ANS: We totally agree! Though the computing time of normal equations in a single iteration can't be exacted accurately from commercial solvers(told by solver developers), we'll try our best to figure out.

Q4: " I think it doesn't make much sense to compare an interior point method with first order methods....."

ANS: Thanks for very meaningful comments! We did compare our algorithm with best commercial solvers using interior points methods(Gurobi/MOSEK, and CPLEX later). We also compared with mainstream methods in this area– even though they are totally different types of methods– to show its competitiveness.

The comments on improving the clarity and the fast matrix inversion are greatly appreciated. They can improve the theoretical complexity of our SLRM/DLRM. The revision will be made accordingly!

Response to Reviewer 3: Thank you for sharp observation and kind advice!

Q1: "The paragraphs from 132–148 ..." ANS: Thanks for pointing out the problem! Our writing here isn't clear. MAAIPM cannot optimally solve the free support case. We'll rewrite this more carefully.

Q2: "In lines 237–239 ..." ANS: Thanks for great suggestion! We have finished the experiment and will add it in the paper. The result shows that our memory usage is higher than popular first order methods, but it's usually within constant times of them. Essentially our advantage lies in the balance of accuracy and speed.

Q3: "In addition to Gurobi ..." ANS: We thank the referee for raising an excellent point. Mosek was tested and is much faster than Gurobi, though it is still roughly 3 times slower than MAAIPM and occupies more memory than MAAIPM for large problems. We will include it in the paper and will test CPLEX later. MAAIPM possesses not only computation speed advantage, but also computation memory efficiency.

Q4: "The authors can also compare ..." ANS: Thanks for helpful suggestions! We've been testing newer algorithms and will add those in the revision besides Solomon et al.'s convolutional Sinkhorn algorithm.