Thank you for the feedback and constructive suggestions. We will incorporate the proposed changes to improve the exposition, including the extended discussion on the diversity selection strategy, more details on the implementation hyperparameters, clarification of the limitations of our work, and adding the suggested related work.

Technical Novelty: To the best of our knowledge, our method is the first to 1) find a continuous Pareto front representation for the Bayesian optimization problem based on an approach with dual KKT conditions; 2) propose a diversity-guided selection strategy, where the diversity metric takes into consideration both design and performance space. The work of [39] develops a method for computing the Pareto front from a set of known objective functions, which is only one step in the Bayesian optimization problem. On its own, the method of [39] does not have a selection strategy to select a discrete set of samples in each iteration to improve the Pareto front in the most sample-efficient way. Furthermore, it does not define a surrogate model and an acquisition function to be able to handle the black-box or expensive-to-evaluate objective functions. However, we found that it naturally suits one building block of the Bayesian optimization as it requires 1st and 2nd order derivatives of objectives. These derivatives are typically unknown for many real-world problems, yet available to us from the Gaussian process (GP). As opposed to other previous works on the selection strategy, we argue that diversity should not be determined solely by the distance in the design or the performance space since the solutions that are close in the design space may not be close in the performance space, and vice versa. Therefore, we propose to group the solutions into different diversity regions based on their correlation between design parameters and performance, in addition to their locations in both spaces. As a result, the grouping also exports different patterns of the Pareto-optimal solutions, which can provide the user with more insights into the problem and potentially help guide future research (see Appendix E).

Baseline Algorithms Choice: After a thorough literature review, we only found two published algorithms that focus on multi-objective Bayesian optimization (MOBO) with batch evaluations, which are TSEMO and MOEA/D-EGO. Thus we provided direct and more detailed comparisons against them in the paper. As the number of published algorithms that target MOBO in a batch scenario is very limited, we also included comparison against the most popular and state-of-the-art algorithms in MOBO with sequential evaluations that a user would typically utilize if not using TSEMO or MOEA/D-EGO. To use them in the parallel evaluations setup, we implemented relatively simple but reasonable extensions towards batch evaluations on top of these sequential baseline algorithms. However, unlike extending sequential MOBO algorithms to batch scenario, extending single-objective batch BO algorithms to the multi-objective scenario is not straightforward and would require adding technical novelties to those algorithms to make a fair comparison.

Evaluation: We conducted the experiments on standard benchmark problems that involve many different shapes and types of Pareto set and front. For real-world problems, a lot of designs are considered different in not only problem domains but also forms of evaluation functions. We choose these problems with no particular reason except they seem to be the standard in testing the algorithm performance and are easily accessible. We showed the results for batch sizes 4, 5, 10 and 20, and omitted the results for sequential evaluation (when batch size equals 1) because our diversity is defined for the individuals inside a batch. However, we also performed experiments using batches of size 1 and 2. The results show our algorithm still outperforms others, thanks to the multi-objective solver that we use, but with less advantages compared to larger batch sizes. We are happy to add these results in the supplementary material. The main evaluation criterion we used is hypervolume indicator since it is the most commonly used performance metric in the MOBO community. Apart from the quantitative numbers demonstrating the performance, we also directly showed optimization results in the performance space and empirically proved our algorithm converges faster to the true Pareto front (see Figure 1).

Hyperparameters: The hyperparameters of our algorithm only come from two sources: 1) the parameters of mean and covariance functions of GP; 2) the parameters of our multi-objective solver. The choices of all hyperparameters are listed in Appendix B. For parameters of GP, we directly used the values suggested by TSEMO. For parameters of the multi-objective solver, we mainly used the values suggested in [39] and some hand-picked values that are not explicitly specified in [39]. From our experience, the performance of DGEMO is mostly affected by the general parameters like batch size, the number of initial samples, and parameters of GP, while less affected by parameters of the multi-objective solver. Therefore we used the same set of hyperparameters across all 20 benchmark problems. The performance of our algorithm could potentially become higher if the hyperparameters are more specifically tuned. For example, we have tested 2-20x smaller buffer sizes compared to the default buffer capacity shown in the paper, and the results in general only look slightly worse ($< 1\%$ for hypervolume). If we use a larger buffer the results would potentially be slightly better.

Computation time: As shown in Appendix C.2, our algorithm is not the fastest in terms of optimization time compared to the baseline algorithms. However, for many real-world problems that need Bayesian optimization, the slightly longer optimization time is usually negligible compared to the evaluation time or cost. Thus we care more about the sample efficiency of the MOBO algorithm rather than the optimization time.