We thank the reviewers for their insightful comments. In the following we only address the major issues. The manuscript will be updated accordingly to reflect the clarifications made here.

Reviewer #1 (I) ["Comparison to naive post processing"]: We recall that our function evaluations are expensive, and hence, throwing away evaluations during post-processing is undesirable. Our approach, in contrast, samples such that most of the function evaluations would have desirable characteristics, and hence, would be efficient. Consider the plots in Fig 1, given a preference-order constraint as "stability of f_0 being more important than f_1" in Schaffer function N. 1, i.e. $||\frac{\partial f_0}{\partial x}|| \leq ||\frac{\partial f_1}{\partial x}||$, Fig 1(left) illustrates the Pareto front obtained by a plain multi-objective optimisation (with no constraints). After the Pareto solutions are found (in 20 iterations), using the derivatives of the trained Gaussian Processes (actual objective functions are black-box), we can post process the obtained Pareto front based on the stability of solutions (lines $46-62$ of the paper). Fig 1(left) shows that only $\frac{6}{18}$ of these solutions have actually met the preference-order constraints. Whereas Fig 1 (right) shows that $\frac{16}{16}$ of the obtained Pareto front solutions by MOBO-PC (in the same 20 iterations) have met the preference-order constraints.

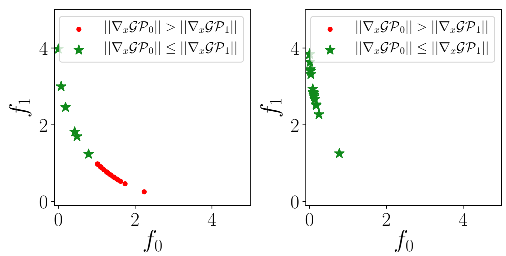

Figure 1: Comparison between a naive post processing approach (left) and MOBO-PC (right).

Reviewer #1 (II), Reviewer #2 (I), Reviewer #3 (I) ["Background and related works"]: Including preferences over objectives in MOO problems for expensive functions dates back to Hakanen et al. [1]. The authors proposed an interactive version of the ParEGO algorithm for identifying "most preferred solutions". At each interaction, the decision maker is shown a subset of non-dominated solutions and she is assumed to provide her preferences in the form of preferred ranges for each objective. Internally, the algorithm samples reference points within the **hyperbox** defined those preferred ranges. This study required both **interaction with user** at each iteration and also **prior knowledge about these hyperboxes**. Recently, Paria et al. [14] (line 330 of paper) introduced a new method to handle such constraints. However this method still requires prior knowledge about the hyperboxes of the form $[[y_1, ..., y_m], [y'_1, ..., y'_m]]$ as exact location of the hyperbox in the objective function space ($\mathbb{R}^m$). We were motivated to remove the need for such complex prior information. Our proposed method achieves this as it only needs information of kind "objective A is more important than objective B", and nothing else. We also note that **evolutionary methods** are not discussed in this paper as they require many evaluations, and hence are not suitable for inexpensive functions.

Reviewer #2 (II) ["Measurement of performance"]: We appreciate this question and agree that our current method of comparison through plots is subjective. However, we can define a measurement by checking how many of the Pareto front solutions satisfy the preference-order constraints. Based on **Algorithm 3** (line 202 of paper), we can calculate **the percentage of solutions that satisfy the preference-order constraints** by using the gradients of the actual functions at iteration t. For example, in the case of Fig 1, all of the obtained solutions are complying with stability preference-order constraints. Our experimental results show 98.8% of solutions found for Schaffer function N. 1 after 20 iterations comply with constraints. As for Poloni's two objective function, 86.3% of the solutions follow the constraints after 200 iterations and finally for Viennet 3D function, this number is 82.5%. Given that the prior knowledge is not provided in [14] (line 330 of paper), the obtained results for their method with same experimental design and same number of iterations are 47.2% for Schaffer function N. 1, 29.6% for Poloni's two objective function and 19.3% for Viennet 3D function respectively. This gap explains **the importance of the prior knowledge** about hyperboxes for their method. The reported numbers are averaged over 10 independent runs. We will include the comprehensive results based on the iteration number in the final version of the paper.

Reviewer #3 (II) ["Usefulness and real-world example"]: We will use two real-world examples on **stability and diversity** to better illustrate the usefulness of MOBO-PC. **(a) Stability:** According to Chow et al. [2] a drug must be tested for **stability** before it can be released for human use. Testing the drugs on humans is a costly and potentially dangerous procedure. There are some vital signs routinely monitored (e.g. heart rate) in the testing procedure and the dosage of the drugs to be tested must be selected in a way that the practitioner can confidently confirm the **positive effects of the drug (objective 1)**, yet make sure the **vital signs such as heart rate (objective 2)** remain stable. Considering these two objectives, finding stable solutions with respect to heart rate is essential. **(b) Diversity:** There are scenarios when diversity is crucial, e.g. the investment strategists generally looking for Pareto optimal investment strategies that prefer diversity in **risk (objective 1)** over **return (objective 2)** as they can later decide their appetite for risk. **(c) Neural networks:** As in neural network example (line 277 of paper), the goal is to illustrate that one can simply ask for more stable solutions with respect to training time of a neural network while optimising the hyperparameters. As all the solutions found with MOBO-PC are in range of $(0, 5)$ training time (unlike the other methods).

[1] On using decision maker preferences with ParEGO, *International Conference on Evolutionary Multi-Criterion Optimization, 282–297, 2017*

[2] Statistical Designs for Pharmaceutical/Clinical Development *Drug Designing, 2169-0138, 2014*