Dear reviewers, we would first like to thank you for the helpful comments and suggestions of improvements. Remarks concerning typos, bad notations and missing references will be fixed according to your suggestions if the manuscript is accepted. Please see our detailed answer to your major concerns below:

Computational aspects: The implementation described in the paper corresponds to a naive one, that computes distances matrices along the projections, and is not the most efficient. Pykeops was used to avoid memory overflows in the evaluation of the cost in Eq (3) which was computed in $O(n^2)$ (both space and time). In fact this implementation was unnecessary since the final cost can actually be computed in $O(n\log(n))$. Indeed, one can develop the sum (3) to compute it in $O(n)$ operations: the term depending on σ (Eq (2)) can be computed in $O(n)$ operations using $W(x, y, \sigma)$ as shown in the supplementary material and for the remaining constants $\sum_{i,j}(x_i - x_j)^4$ (idem for y) the binomial development gets rid of the $\sum_{i,j}$ and only involves $\sum_i$ terms that can be computed in $O(n)$ operations. Overall, in 1D, GW can be computed as efficiently as Wasserstein. As a consequence, the complexity of SGW is exactly the same as for Sliced Wasserstein and Pykeops is not needed anymore. We believe this discussion can be added without changing the overall message of the paper by updating Fig 3 using a pure pytorch implementation as in Fig 1. With this implementation, one can compute SGW between 1M point distributions in 1s (vs 100s with a naive PyKeops implementation). Note also that entropic-GW is implemented on GPU as well. This way, it is clear that our method is responsible for the computational gain that **is not a consequence of using PyKeops** (#R1).

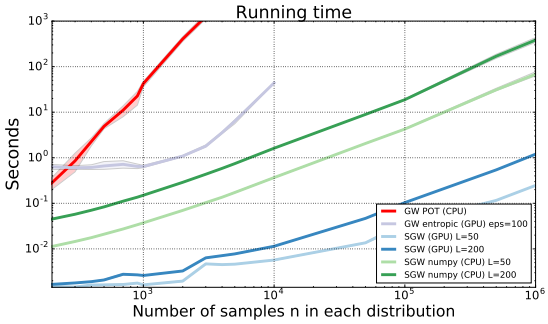

Figure 1: Runtimes comparison between SGW, GW, entropic-GW between 2D random distributions with varying number of points from 0 to 10^6 in log-log scale

Related to the remark of (#R1) we also added the runtimes for two different numbers of projections $L = 50, 200$. The paragraph "Computational aspects" of the paper describes the influence of L on the theoretical complexity. To the best of our knowledge, the effect of L on the quality of estimation of the expectation is a hard question that is still open for the Sliced Wasserstein itself. Runtimes are computed between 2D measures since the dimension does not have an impact on the complexities for computing GW and SGW (they only depend on L and n) (#R1). Moreover, the optimization over the Stiefel manifold does not depend on the number of points but only on the dimension d of the problem so that overall complexity is $niter(Ln\log(n) + d^3)$, which is affordable for small d. On the spirals we observed that computing RISGW is one order of magnitude slower than the non-RI variant on CPU, which is still reasonable (# R1). We propose to add this discussion in the manuscript.

The non Hilbertian case is a very good remark (#R3). One straightforward approach would be to consider an embedding of the distances using multi-dimensional scaling as a prepossessing step or to learn distance-preserving embeddings using Siamese networks. This would come with an additional cost but we believe that this direction is worth investigating and will add it to the discussion.

About the choice of Δ: The map Δ is one of the contributions of the paper. Here we propose a simple method (using a linear map in the Stiefel manifold) to align the spaces, even though one could consider other approaches (e.g a Δ parameterized by a neural network). We believe designing Δ is application dependant and preferred to restrict Δ to the Stiefel manifold in order to ensure rotation invariant guarantees so as to make the connection with an important property of GW. As such, we can use the Δ formulation when $p = q$ to recover this property (#R1). We thank (#R4) for its remark concerning the 4 others discrepancies. We originally did not want to add non obvious extensions of W, SW using the "Δ trick" and only focus our paper on Gromov-Wasserstein as our main result is Theo 3.1. We believe that using Δ with other discrepancies deserves a deeper study since it raises a lot of quantitative and theoretical questions (e.g closed forms for Δ) and we chose not to include such discussions in the scope of the paper. From a purely computational point of view, complexity of all methods are the same (cf remark above). Nevertheless we will add these discrepancies and run corresponding experiments with them in the supplementary as suggested by the reviewer.

Theoretical aspects and proofs: (#R1) the problem (3) is equivalent to Eq (2) in 1D since the quadratic term is constant w.r.t. the permutation σ (as being of the form $\sum_{i,j} c_{\sigma(i),\sigma(j)} = \sum_{i,j} c_{i,j}$) so that the minimization only involves the cross products. We will clarify this point.

Experiments: (# R1) The idea behind the simple spiral example was just to illustrate the different behaviors of GW, SGW and RISGW w.r.t rotations. Indeed, other rotation invariant methods could be applied here and would give similar results.