Reviewer #1 (R1): ♦*[...]tree-sliced Wasserstein distance outperforms the original optimal transport distance[...]* We used the tree-sliced Wasserstein (TSW) and OT distances within an RBF kernel, $k = \exp(-td)$. The TSW kernel is p.d. while the OT kernel $k_{\mathrm{OT}}$ is not. The indefiniteness of $k_{\mathrm{OT}}$ may affect its performances in some applications: $k_{\mathrm{OT}}$ performs worse in TDA applications in §6.1, but works well for documents with word embedding applications in §6.2. ♦*[...]include (in the supplement, at least) the full proof [of Proposition 1]?* Agreed, we will follow your suggestion.
Reviewer #2: ♦*[...]Prop. 1 and 2 are not original[...]* ♦R1: *proof of negative definiteness of the proposed OT[...]* We do state that the proof of Prop. 1 is not original in $\ell$.39–41. Although Prop. 2 follows from Prop. 1, it follows the idea underlying sliced W kernels (or Gaussian processes as you mention), and this result remains new to our knowledge.
♦*[...]bound on the Wasserstein distance[...]interesting result[...]numeric analysis comparing the limit of the hypercube tree-sliced metric to the Wasserstein metric and the sliced Wasserstein metric.*♦R1: *An upper bound on the Euclidean OT[...]* The hypercube tree-sliced metric is our suggestion to build practical tree metrics for TSW when used on low-dimensional data spaces (e.g. TDA in §6.1). We insist that we do not try to mimic the Euclidean OT ($W_2$) or the sliced-Wasserstein (SW), but rather propose a variant of OT distance. As stated in $\ell$.173–179, and $\ell$.158–161 in §5, SW is a special case of TSW. Following your point, we have carried out the following experiment: for a query point $q$, let $p$ be its nearest neighbor w.r.t. TSW. Figure 1 illustrates that $p$ is very likely among the top 5 (on MPEG7), and top 10 (on Orbit) near neighbors on the $W_2$ space (results are averaged over 1000 runs of random split $90\%/10\%$ for training and test). When the number of tree-slices in TSW increases, the $W_2$ near-neighbor rank of $p$ is improved. These empirical results suggest that TSW may agree with some aspects of $W_2$.

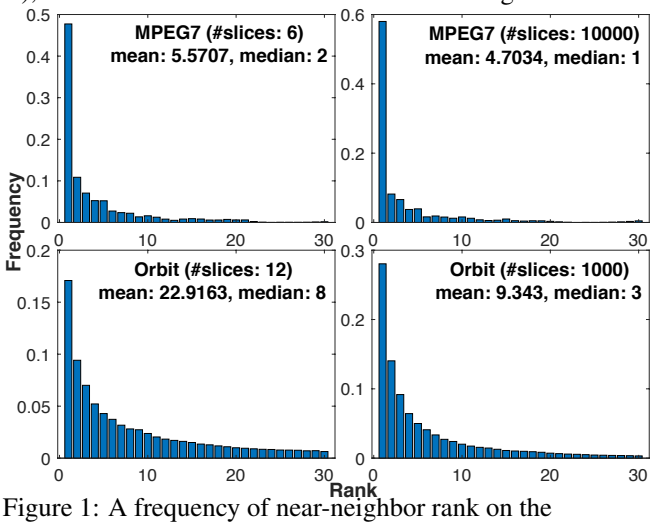

Figure 1: A frequency of near-neighbor rank on the $W_2$ space for the nearest neighbor w.r.t. TSW.

♦*[...]experiment where the metric is used in a more conventional OT problem such as color transfer or generative modeling.* In the experiments, we used RBF kernels ($k = \exp(-td)$ for a given metric $d$) with SVM which usually improves on $k$-NN results. We will add $k$-NN results. We are now considering color transfer and barycenter applications. Gradients of TSW w.r.t. supports and weights of empirical measures can be recovered pending some choices in how interpolations are defined.♦*a single multi paneled figure [...] word embedding experiment[...]presented before the TDA* Many thanks for your suggestions. We will incorporate them.
Reviewer #3: ♦*What is most troubling is that the paper seems to be completely unaware any literature of embedding points into a distribution over metrics, and claims some standard and well-known techniques and novelties[...]* We understand your point and will do everything to correct this misunderstanding. This was caused by a lack of care in the presentation of §4. This was not the message we wanted to convey. We will rewrite this section following your comments. As you have gathered from our algorithms, approximating an arbitrary metric using trees is *not* a key goal in our submission, our goal is stated in $\ell$.41–44 in §.1. Much like 1D projections do not offer interesting properties from a distortion perspective but remain useful for SW, we do believe that trees with large distortion can still remain useful. This is because metric approximations are used within another computation (Wasserstein) and therefore we do not gain from overfitting too much our trees so that they match the true metric, as long as they provide guidance on the optimal assignment. We will insist more on the importance of sampling tree metrics randomly, both for low-dimensional in §6.1 *and* high-dimensional §6.2 regimes. ♦*Definite-negativity is mentioned and highlighted[...] explain why is it important to you. Is this to ensure that the kernel is positive-definite?* Negative definiteness of a distance means essentially that the space is flat and that positive definite kernels can be easily derived from them, following Berg et al.'s theorem. This is why kernel methods kick in from §.6 (or Gaussian processes as per Reviewer #2's suggestion). We will clarify this motivation following your comment. ♦*[...]which kernel did you actually use in the experiments – exp(-TW) or exp(-TSW)?[...]kernel is called $k_{\mathrm{TW}}$ and not $k_{\mathrm{TSW}}$?[...]TSW is also negative-definite, simply because the average of $l_1$-metrics is an $l_1$-metric – right?)* In the experiments, we used the kernel $\exp(-td_{\mathrm{TSW}})$ as stated in $\ell$.225–227 for §6.1, and $\ell$.289–292 for §6.2. We will define $k_{\mathrm{TSW}}$, and rename $k_{\mathrm{TW}}$ to $k_{\mathrm{TSW}}$ in the experiments following your suggestion. Indeed, averaging of negative definite functions is trivially negative definite. Hence, $d_{\mathrm{TSW}}$ is negative definite. We will clarify it in the updated version. ♦*[...]set the number of clusters in each level?[...]connection to fast Gauss transform [...]more related than any other clustering method[...]use farthest-point clustering[...]downstream motivation and does it effect the results?* We fixed the same number of clusters $\kappa$ when performing the farthest-point clustering (replace step 9 in Algorithm 1) at different height levels. $\kappa$ is typically chosen via cross-validation. Moreover, we also illustrate the effect of $\kappa$ in applications in Figure 5 (and Figures 5–7 in the supplement). In general, one can apply any favorite clustering methods. We used the farthest-point clustering due to its fast computation, i.e. $O(n \log \kappa)$ as stated in $\ell$.148–149 to construct practical tree metrics for applications with high-dimensional data space, e.g. documents with word embedding applications in §6.2.