All reviewers

We would like to thank all the reviewers for their positive assessment of our work. We are also grateful for pointing out the typos and providing writing suggestions — we will incorporate them in the updated version.

Reviewer 1

- **Experimental results.**
  We would like to politely disagree with your statement that "the proposed model seems to underperform the existing and simple baseline models on most datasets". In the density estimation experiments (Tables 1 & 2), our method achieves the best or the second best score in 19/26 cases. RNN is the only model besides TriTPP that consistently achieves good results across all the datasets. In the NLL experiment TriTPP dominates the simpler baselines (IP, RP, MRP models) on Hawkes1, Hawkes2, SC, PUBG, Reddit-C, Reddit-S, Twitter (7/13), and achieves nearly identical scores on the remaining datasets (6/13). Similarly, we convincingly outperform the Hawkes model on Hawkes2, SC, IPP, MRP, RP, PUBG, Reddit-S, Yelp1, Yelp2 datasets (9/13), get slightly better scores on Taxi and Twitter (2/13) and get worse scores only on the Hawkes1 and Reddit-C datasets (2/13).
  We would also like to reiterate that the parallelism of our model is benefecial not only for faster training on long sequences. Rather, fast parallel sampling opens new applications for TPPs that were completely off the charts for existing approaches (where sampling can be hundreds of times slower).

Reviewer 2

- **Choice of the block size.**
  We found that increasing the block size beyond 16 for the TriTPP model did not improve the performance. In contrast, the RNN did benefit from larger hidden sizes (32 or 64). Therefore, we allowed the RNN to have more parameters in the density estimation experiment (Section 6.2), not to give an unfair advantage to TripTPP. In the scalability experiment (Section 6.1) we made sure that both models have approximately the same number of parameters.
- **Additional nonlinearities.**
  We found that stacking the nonlinearities (splines) did not improve the performance on the validation set. One explanation for this can be that the splines do not benefit from depth, as their flexibility can be increased by simply increasing the number of knots.
- **Using SGD & tuning the optimization procedure.**
  Thank you, we will run more experiments to see if this will allow us to improve the performance.
- **More examples & intuition for TPPs and triangular maps.**
  Thank you for the excellent suggestion — we will add more examples as well as the following discussion to help the readers build intuition on this topic.
  The function Λ captures the global trend by rescaling the time (e.g., if there are more events during the weekend than on a weekday). The pairwise difference matrix $\boldsymbol{D}$ computes the "rescaled" inter-event times after the global trend component Λ has been removed. The block-diagonal matrices $\boldsymbol{B_1}, ..., \boldsymbol{B}_L$ capture the interactions between events, which allows our model to capture self-exciting or self-correcting behavior. The function Φ models the distribution of the "raw" inter-event times (this may correspond to a recurring pattern, as in a renewal process). Finally, the cumulative sum matrix $\boldsymbol{C}$ sums up the transformed inter-event times to obtain the arrival times of the base unit Poisson process.

Reviewer 3

- **Sampling from a Poisson process.**
  We would like to point out that we can sample from a unit-rate homogeneous Poisson process in parallel as following (using Pytorch as an example):

```
tau = torch.empty(N).exponential_(1.0)   # draw inter-event times from Expo(1)
t = tau.cumsum(dim=-1)                   # obtain HPP arrival times
```

  Both steps can be done in parallel in $O(N)$ operations, including the cumulative sum (Blelloch, 1990. "Prefix sums and their applications").