Supplementary Material

"Survival Permanental Processes for Survival Analysis with Time-Varying Covariates"

Anonymous Author(s)
Affiliation
Address
email

S1 Derivation of MAP Estimator

We detail the derivation of the MAP estimator (12). The functional derivative of $S(x(\boldsymbol{y}), \underline{x}(\boldsymbol{y}))$ should be zero on the MAP estimator $\hat{x}(\boldsymbol{y})$:

$$\begin{aligned}
\delta S\big(\hat{x}(\boldsymbol{y}), \underline{\hat{x}}(\boldsymbol{y})\big) &= \int_{\mathcal{Y}} \left[\frac{\delta S}{\delta \hat{x}(\boldsymbol{y})} \delta x(\boldsymbol{y}) + \frac{\delta S}{\delta \underline{\hat{x}}(\boldsymbol{y})} \delta \underline{x}(\boldsymbol{y}) \right] d\boldsymbol{y} + O((\delta x)^2) \\
&\simeq \int_{\mathcal{Y}} \left[2 \sum_{j=1}^{J} \Delta_j \hat{x}(\boldsymbol{y}) \delta(\boldsymbol{y} - \boldsymbol{y}_j) - \sum_{n=1}^{N} \frac{2}{\hat{x}(\boldsymbol{y})} \delta(\boldsymbol{y} - \tilde{\boldsymbol{y}}_n) + \frac{1}{2} \underline{\hat{x}}(\boldsymbol{y}) \right] \delta x d\boldsymbol{y} \\
&\qquad + \int_{\mathcal{Y}} \frac{1}{2} \hat{x}(\boldsymbol{y}) \delta \underline{x} d\boldsymbol{y} \\
&= \int_{\mathcal{Y}} \left[2 \sum_{j=1}^{J} \Delta_j \hat{x}(\boldsymbol{y}) \delta(\boldsymbol{y} - \boldsymbol{y}_j) - \sum_{n=1}^{N} \frac{2}{\hat{x}(\boldsymbol{y})} \delta(\boldsymbol{y} - \tilde{\boldsymbol{y}}_n) + \underline{\hat{x}}(\boldsymbol{y}) \right] \delta x d\boldsymbol{y} = 0,
\end{aligned}$$

where the following relation was used,

$$\begin{aligned}
\int_{\mathcal{Y}} \hat{x}(\boldsymbol{y}) \delta \underline{x} d\boldsymbol{y} &= \int_{\mathcal{Y}} \hat{x}(\boldsymbol{y}) \int_{\mathcal{Y}} k^*(\boldsymbol{y}, \boldsymbol{y}') \delta x(\boldsymbol{y}') d\boldsymbol{y}' d\boldsymbol{y} \\
&= \int_{\mathcal{Y}} d\boldsymbol{y}' \delta x(\boldsymbol{y}') \int_{\mathcal{Y}} k^*(\boldsymbol{y}, \boldsymbol{y}') \hat{x}(\boldsymbol{y}) d\boldsymbol{y} \\
&= \int_{\mathcal{Y}} \underline{\hat{x}}(\boldsymbol{y}') \delta x d\boldsymbol{y}'. \quad \because) \, k^*(\boldsymbol{y}, \boldsymbol{y}') = k^*(\boldsymbol{y}', \boldsymbol{y})
\end{aligned}$$

Thus the following equation is derived,

$$\underline{\hat{x}}(\boldsymbol{y}) + 2 \sum_{j=1}^{J} \Delta_j \hat{x}(\boldsymbol{y}_j) \delta(\boldsymbol{y} - \boldsymbol{y}_j) = \sum_{n=1}^{N} \frac{2}{\hat{x}(\tilde{\boldsymbol{y}}_n)} \delta(\boldsymbol{y} - \tilde{\boldsymbol{y}}_n), \quad \boldsymbol{y} \in \mathcal{Y}. \tag{S1}$$

By applying operator $\mathcal{K}$ to (S1), we obtain a linear integral equation that derives the MAP estimator $\hat{x}(\boldsymbol{y})$ as follows,

$$\hat{x}(\boldsymbol{y}) + 2 \sum_{j=1}^{J} \Delta_j \hat{x}(\boldsymbol{y}_j) k(\boldsymbol{y}, \boldsymbol{y}_j) = 2 \sum_{n=1}^{N} k(\boldsymbol{y}, \tilde{\boldsymbol{y}}_n) \hat{x}(\tilde{\boldsymbol{y}}_n)^{-1}, \quad \boldsymbol{y} \in \mathcal{Y}.$$

The linearity of the integral equation permits a representation of the form

$$\hat{x}(\boldsymbol{y}) = 2 \sum_{n=1}^{N} h(\boldsymbol{y}, \tilde{\boldsymbol{y}}_n) \hat{x}(\tilde{\boldsymbol{y}}_n)^{-1},$$

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

where $h(\boldsymbol{y}, \boldsymbol{y}')$ is a positive semi-definite kernel that solves the following equation,

$$h(\boldsymbol{y}, \boldsymbol{y}') + 2\sum_{j=1}^{J} k(\boldsymbol{y}, \boldsymbol{y}_j)\Delta_j h(\boldsymbol{y}_j, \boldsymbol{y}') = k(\boldsymbol{y}, \boldsymbol{y}').$$

Note that the derivations follow Kim [2].

S2 Model Configuration

S2.1 Synthetic Data

Survival Permanental Process (SurvPP)

For all the experiments in Section 4, we set the number of features for Random feature map [4] (M), the learning parameter (lr), and the stop condition (G) for Adam [3] as follows:

$$M = 500, \quad lr = 0.05, \quad G < 10^{-5},$$

We applied to SurvPP a multiplicative Gaussian kernel

$$k(\boldsymbol{y}, \boldsymbol{y}') = \prod_{d=1}^{2} e^{-(\theta(y_d - y'_d))^2}, \quad \boldsymbol{y} = (y_1, y_2),$$

where the hyper-parameter θ was optimized for each data by maximizing the marginal likelihood through grid search. In the experiments with synthetic data, we selected a set of nine values for θ for the grid points,

$$\theta \in \{0.1,\ 0.2,\ 0.5,\ 0.7,\ 1.0,\ 2.0,\ 5.0,\ 7.0,\ 10.0\}.$$

We implemented SurvPP by using TensorFlow-2.10. A MacBook Pro with 12-core CPU (Apple M2 Max) was used, where GPU was set as off (`tf.device('/cpu:0')`) for a fair comparison with the benchmarks.

Cox Proportional Hazards Model (CoxPH)

We implemented CoxPH through package `survival.coxph` in R-4.2.3 (LGPL-3) [5]. The calls in `coxph` to fit a model and compute a base hazard function were

$$\begin{aligned} &> \texttt{cfit} = \texttt{coxph}(\texttt{Surv}(\texttt{Start}, \texttt{Stop}, \texttt{Event}) \sim \texttt{cov1} + \texttt{cov2}, \texttt{df}) \\ &> \texttt{sfit} = \texttt{survfit}(\texttt{cfit}, \texttt{list}(\texttt{cov1} = 0, \texttt{cov2} = 0)), \end{aligned}$$

where `df` was the survival data in a counting process format.

Generalized Boosted Model (GBM)

We implemented GBM through package `gbm3.gbmt` (GPL) [1] in R-4.2.3. The call in `gbmt` to fit a model was

$$\begin{aligned} > \texttt{gfit} = \texttt{gbmt}(&\texttt{Surv}(\texttt{Start}, \texttt{Stop}, \texttt{Event}) \sim \texttt{cov1} + \texttt{cov2}, \texttt{data} = \texttt{df}, \\ &\texttt{distribution} = \texttt{gbm\_dist}(\text{“CoxPH”}), \\ &\texttt{cv\_folds} = 10, \texttt{train\_params} = \texttt{params}, \\ &\texttt{par\_details} = \texttt{gbmParallel}(\texttt{num\_threads} = 12)), \end{aligned}$$

where `params` represents the hyperparameter. We selected a set of nine hyperparameters for the grid search,

$$\texttt{num\_trees} \in \{500, 1000, 2000\} \ \times\ \texttt{shrinkage} \in \{0.001, 0.005, 0.01\},$$

and found one that minimized the cross validation error (`gfit$valid.error`), where `num_trees` and `shrinkage` represent the number of trees and the shrinkage/learning rate, respectively.

[1] https://rdrr.io/github/gbm-developers/gbm3/

Random Forest-based Model (RFM)

We implemented RFM through package `LTRCforests` in R-4.2.3 (GPL) [6]. The call to fit a model was

$$> \texttt{rfit} = \texttt{ltrcrrf}(\texttt{Surv}(\texttt{Start}, \texttt{Stop}, \texttt{Event}) \sim \texttt{cov1} + \texttt{cov2}, \texttt{data} = \texttt{df}, \\ \texttt{id} = \texttt{ID}, \texttt{mtry} = \texttt{ceiling}(10), \texttt{ntree} = 100).$$

S2.2 Real-world Data

Survival Permanental Process (SurvPP)

For the experiments in Section S4, we set the number of features for Random feature map [4] (M), the learning parameter (lr), and the stop condition (G) for Adam [3] as follows:

$$M = 500, \quad lr = 50, \quad G < 10^{-5},$$

We applied to SurvPP a multiplicative Gaussian kernel

$$k(\boldsymbol{y}, \boldsymbol{y}') = \prod_{d=1}^{13} e^{-(\theta(y_d - y'_d))^2}, \quad \boldsymbol{y} = (y_1, \ldots, y_{13}),$$

where the hyper-parameter θ was optimized for each data by maximizing the marginal likelihood through grid search. In the experiments with synthetic data, we selected a set of nine values for θ for the grid points,

$$\theta \in \{0.1,\ 0.2,\ 0.5,\ 0.7,\ 1.0,\ 2.0,\ 5.0,\ 7.0,\ 10.0\}.$$

Here, we normalized the 13 covariates of PBC data so that $y_d \to 0.1(y_d - \text{mean}[y_d])/\text{std}[y_d]$.

As in the experiments on synthetic data, GPU was set as off (`tf.device('/cpu:0')`) for a fair comparison with the benchmarks.

Cox Proportional Hazards Model (CoxPH)

We implemented CoxPH through package `survival.coxph` in R-4.2.3 (LGPL-3) [5]. The calls in `coxph` to fit a model and compute a base hazard function were

$$> \texttt{cfit} = \texttt{coxph}(\texttt{Surv}(\texttt{Start}, \texttt{Stop}, \texttt{Event}) \sim \texttt{age} + \texttt{edema} + \texttt{alk.phos} + \texttt{chol} + \texttt{ast} \\ + \texttt{platelet} + \texttt{spiders} + \texttt{hepato} + \texttt{ascites} + \texttt{albumin} + \texttt{bili} + \texttt{protime}, \texttt{df})$$

$$> \texttt{sfit} = \texttt{survfit}(\texttt{cfit}, \texttt{list}(\texttt{age} = 0, \texttt{edema} = 0, \texttt{alk.phos} = 0, \texttt{chol} = 0, \texttt{ast} = 0, \\ \texttt{platelet} = 0, \texttt{spiders} = 0, \texttt{hepato} = 0, \texttt{ascites} = 0, \texttt{albumin} = 0, \\ \texttt{bili} = 0, \texttt{protime} = 0)),$$

where `df` was the survival data in a counting process format.

Generalized Boosted Model (GBM)

We implemented GBM through package `gbm3.gbmt` (GPL) [2] in R-4.2.3. The call in `gbmt` to fit a model was

$$> \texttt{gfit} = \texttt{gbmt}(\texttt{Surv}(\texttt{Start}, \texttt{Stop}, \texttt{Event}) \sim \texttt{age} + \texttt{edema} + \texttt{alk.phos} + \texttt{chol} + \texttt{ast} \\ + \texttt{platelet} + \texttt{spiders} + \texttt{hepato} + \texttt{ascites} + \texttt{albumin} + \texttt{bili} \\ + \texttt{protime}, \texttt{data} = \texttt{df}, \\ \texttt{distribution} = \texttt{gbm\_dist}(\text{“}\texttt{CoxPH}\text{”}), \\ \texttt{cv\_folds} = 10, \texttt{train\_params} = \texttt{params}, \\ \texttt{par\_details} = \texttt{gbmParallel}(\texttt{num\_threads} = 12)),$$

where `params` represents the hyperparameter. We selected a set of nine hyperparameters for the grid search,

$$\texttt{num\_trees} \in \{500, 1000, 2000\} \ \times \ \texttt{shrinkage} \in \{0.001, 0.005, 0.01\},$$

and found one that minimized the cross validation error (`gfit$valid.error`), where `num_trees` and `shrinkage` represent the number of trees and the shrinkage/learning rate, respectively.

[2] https://rdrr.io/github/gbm-developers/gbm3/

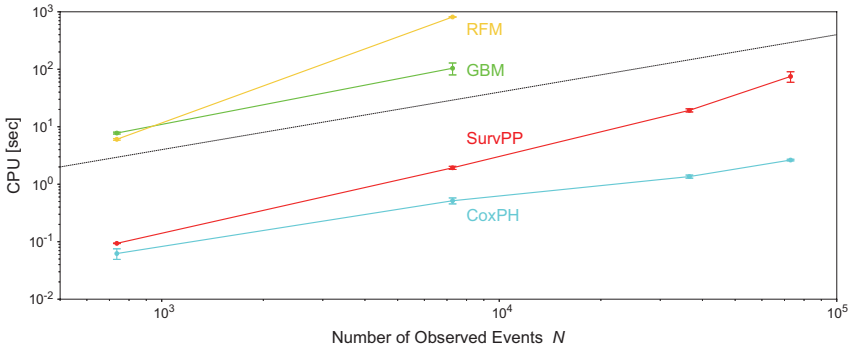

Figure S1: The CPU times demanded for estimating a hazard function regarding the number of observed events. The error bars represent the standard deviations across 10 trials. The dashed line represents a line of $CPU \propto N$ as reference. For GBM and SurvPP, the average cpu times over 9-points grid search of the hyperparameter are displayed. The CPU times of GBM and RFM exceeded 10^3 seconds with $N > 10^4$, and the estimations were given up.

Random Forest-based Model (RFM)

We implemented RFM through package `LTRCforests` in R-4.2.3 (GPL) [6]. The call to fit a model was

$$> \texttt{rfit} = \texttt{ltrcrrf}(\texttt{Surv}(\texttt{Start}, \texttt{Stop}, \texttt{Event}) \sim \texttt{age} + \texttt{edema} + \texttt{alk.phos} + \texttt{chol} + \texttt{ast} + \texttt{platelet} + \texttt{spiders} + \texttt{hepato} + \texttt{ascites} + \texttt{albumin} + \texttt{bili} + \texttt{protime}, \texttt{data} = \texttt{df}, \texttt{id} = \texttt{ID}, \texttt{stepFactor} = 1.5).$$

S3 Experiment on Larger Data Sets

To examine the computation scalability of the compared models regarding the number of observed events N, we created data sets with the user size $U \in \{10^3, 10^4, 5 \cdot 10^4, 10^5\}$ according to the nonlinear scenario (see Section 4.1),

$$\lambda_{non}(t) = h(t) \exp\big[2 - 5(y_1^2(t) + y_2^2(t))\big], \quad h(t) = 2 \cdot t^{3/2},$$

which resulted in the data sets with $N \in \{818, 8082, 40660, 81066\}$. Here, we set J_u as 10 for the counting process format of data. For each dataset, we randomly split the U individuals into 10 subgroups, repeated assigning 9 subgroups to training data, and conducted 10 trials of evaluations of the CPU times demanded for estimating a hazard function. Figure S1 displays the CPU times as function of N of training data. It shows that the computation of CoxPH scaled linearly with N clearly, while that of SurvPP seems to be a little more than linear. It is because that each iteration of gradient descent algorithm scaled linearly with N, but the number of iterations to meet a stop condition $G < 10^{-5}$ increased moderately with N. Among the non-parametric approaches (SurvPP, GBM, and RFM), our SurvPP achieved the fastest computation, which was hundreds of times faster than the others regardless of the number of observed events N.

S4 Experiment on Real-world Data

We examined the validity of SurvPP against the benchmark models on real-world survival data set, *Mayo Clinic Primary Biliary Cholangitis Data*, provided by R package `survival` (LGPL-3) [5]. 312 patients with primary biliary cirrhosis (PBC) were enrolled in a randomized medical trial at the Mayo Clinic between 1974 and 1984 [1], where events were the time of death. In the data set, twelve time-varying covariates were measured at entry and at yearly intervals, which include age at entry (`age`), alkaline phosphotase (`alk.phos`), logarithm of serum albumin (`albumin`), presence of ascites (`ascites`), aspartate aminotransferase (`ast`), logarithm of serum bilirubin (`bili`), serum cholesterol (`chol`), condition of edema (`edema`), presence of hepatomegaly or enlarged liver (`hepato`), platelet count (`platelet`), logarithm of prothrombin time (`protime`) and presence or absence of spiders (`spiders`). As with the experiments on synthetic data, we randomly split the

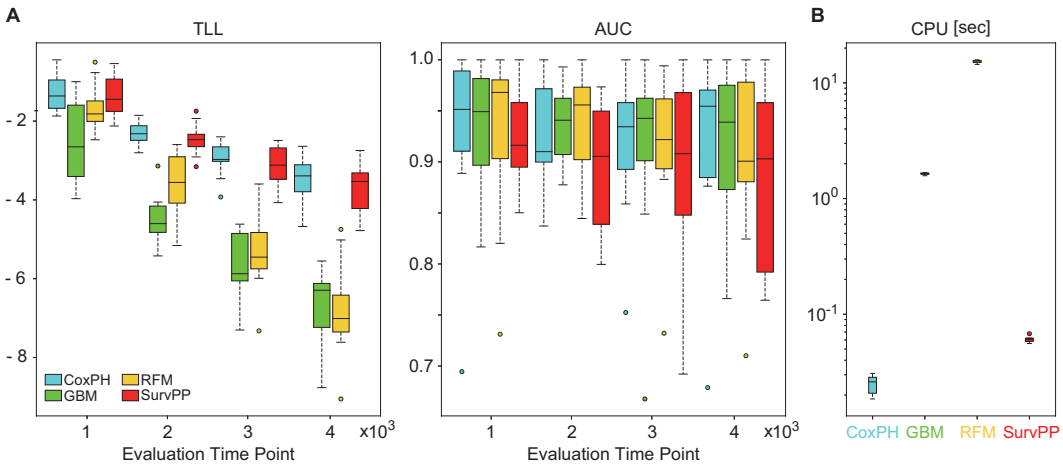

Figure S2: Performances on the real-world dataset. (A) Box plot of TLL and AUC as functions of evaluation time points: the higher, the better. (B) Box plot of CPU times demanded for estimating a hazard function: the lower, the better.

312 individuals into10 subgroups, assigned one to test and the others to training data, and conducted 10-fold cross evaluation of the predictive performances.

Figure S2 displays the predictive performances on the real-world dataset, showing that all compared models achieved comparable performances at AUC, while SurvPP and CoxPH outperformed significantly at TLL against the other non-parametric models. We confirmed that among the non-parametric models, our SurvPP was the best on both the predictive accuracy and the computational time.

References

[1] E. Rolland Dickson, Patricia M. Grambsch, Thomas R. Fleming, Lloyd D. Fisher, and Alice Langworthy. Prognosis in primary biliary cirrhosis: Model for decision making. *Hepatology*, 10(1):1–7, 1989.

[2] Hideaki Kim. Fast Bayesian inference for Gaussian Cox processes via path integral formulation. In *Advances in Neural Information Processing Systems 34*, 2021.

[3] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[4] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems 20*, 2007.

[5] Terry M. Therneau and Thomas Lumley. Package 'survival'. *R Top Doc*, 128(10):28–33, 2015.

[6] Weichi Yao, Halina Frydman, Denis Larocque, and Jeffrey S. Simonoff. Ensemble methods for survival data with time-varying covariates. *arXiv preprint arXiv:2006.00567*, 2020.