STABILIZED NEURAL PREDICTION OF POTENTIAL OUTCOMES IN CONTINUOUS TIME

Konstantin Hess & Stefan Feuerriegel
Munich Center for Machine Learning
LMU Munich
{k.hess,feuerriegel}@lmu.de

ABSTRACT

Patient trajectories from electronic health records are widely used to estimate conditional average potential outcomes (CAPOs) of treatments over time, which then allows to personalize care. Yet, existing neural methods for this purpose have a key limitation: while some adjust for time-varying confounding, these methods assume that the time series are recorded in *discrete time*. In other words, they are constrained to settings where measurements and treatments are conducted at fixed time steps, even though this is unrealistic in medical practice. In this work, we aim to estimate CAPOs in *continuous time*. The latter is of direct practical relevance because it allows for modeling patient trajectories where measurements and treatments take place at arbitrary, irregular timestamps. We thus propose a new method called *stabilized continuous time inverse propensity network* (SCIP-Net). For this, we further derive stabilized inverse propensity weights for robust estimation of the CAPOs. To the best of our knowledge, our SCIP-Net is the first neural method that performs *proper adjustments for time-varying confounding in continuous time*.

1 INTRODUCTION

Estimating conditional average potential outcomes (CAPOs) of treatments is crucial to personalize treatment decisions in medicine (Feuerriegel et al., 2024). Such CAPOs are increasingly predicted based on patient data from electronic health records (Bica et al., 2021). This thus requires methods that can model the time dimension in patient trajectories and, therefore, estimate CAPOs ***over time***.

Existing neural methods for estimating CAPOs over time primarily model the patient trajectory in *discrete time* (e.g., Bica et al., 2020; Li et al., 2021; Lim et al., 2018; Melnychuk et al., 2022). As such, these methods make *unrealistic* assumptions that both health measurements and treatments occur on a fixed, regular schedule (such as, e.g., daily or hourly). However, both health measurements and treatments take place at arbitrary, irregular timestamps based on patient needs. For example, patients in a critical state may be subject to closer monitoring, so that measurements are recorded more frequently (Allam et al., 2021).

To account for arbitrary, irregular timestamps of both health measurements and treatments, methods are needed that correctly model the patient trajectory in *continuous time* (Lok, 2008; Røysland, 2011; Rytgaard et al., 2022). However, neural methods that operate in continuous time are scarce (see Sec. 2). Crucially, existing neural methods have a key limitation in that they *fail to properly account for time-varying confounding* (e.g., Seedat et al., 2022). This means that, for a sequence of future treatments, the corresponding confounders lie also in the future, are thus unobserved, and therefore need to be adjusted for. Yet, existing neural methods rely only on heuristics such as balancing, which targets an improper estimand and thus leads to estimates that are *biased*. To the best of our knowledge, there is <u>no</u> neural model that estimates CAPOs in continuous time while properly adjusting for time-varying confounding.

In this paper, we aim to estimate CAPOs for sequences of treatments ***in continuous time*** while properly adjusting for time-varying confounding. However, this is a non-trivial challenge, as this requires a method that can perform adjustments at arbitrary timestamps. While there are methods to adjust for time-varying confounding in discrete time, similar methods for continuous time are still lacking.

	Correct timestamps?	Adjustment for time-varying confounding?	Existing works
(1) Neural methods in discrete time	✗	✗	**CRN** (Bica et al., 2020), **CT** (Melnychuk et al., 2022)
	✗	✓	**RMSNs** (Lim et al., 2018), **G-Net** (Li et al., 2021)
(2) Neural methods in continuous time	✓	✗	**TE-CDE** (Seedat et al., 2022)
SCIP-Net (ours)	✓	✓	—

Table 1: Comparison of key neural methods for estimating CAPOs over time. Our SCIP-Net is the first method to perform proper adjustments for time-varying confounding in continuous time.

Therefore, we first derive a tractable expression for inverse propensity weighting (IPW) in continuous time. However, a direct application of IPW may suffer from severe overlap violations and thus lead to extreme weights. As a remedy, we further derive **stabilized IPW in continuous time**. We then use our stabilized IPW to propose a novel method, which we call **stabilized continuous time inverse propensity network (SCIP-Net)**. Unlike existing methods, ours is the first neural method to estimate CAPOs in continuous time while properly adjusting for time-varying confounding.

We make the following contributions:[1] (1) We introduce SCIP-Net, a novel neural method for estimating conditional average potential outcomes in continuous time. (2) We derive a tractable version of IPW in continuous time, which provides the theoretical foundation of our paper for proper adjustments for time-varying confounding. Further, we propose stabilized IPW in continuous time, which we then use in our SCIP-Net. (3) We demonstrate through extensive experiments that our SCIP-Net outperforms existing neural methods.

2 Related Work

Table 1 presents an overview of key neural methods for estimating CAPOs over time. An extended related work is in Supp. A.

Average vs. individualized estimation: Estimating *average* potential outcomes over time is a well-studied problem in classical statistics (e.g., Bang & Robins, 2005; Lok, 2008; Robins, 1986; 1999; Robins & Hernán, 2009; Røysland, 2011; Rytgaard et al., 2022; 2023; van der Laan & Gruber, 2012). However, these methods are *population-level* approaches and thus do *not* make individualized estimates at the patient level. Put simply, the observed history of an individual patient is ignored. Therefore, they are *not* suitable for personalized medicine. In contrast, our work (and the following overview) focuses on potential outcome estimation *conditional* on the observed patient history, which thus allows us to make individual-level estimates for personalized medicine.

(1) **Neural methods in discrete time:** Some neural methods for estimating CAPOs over time impose a *discrete time* model on the data (e.g., Bica et al., 2020; Li et al., 2021; Lim et al., 2018; Melnychuk et al., 2022). As such, these methods operate under the assumption of both fixed observation and treatment schedules, yet which is unrealistic in clinical settings. Instead, patient health is typically monitored at arbitrary, irregular timestamps, and the timing of treatments also takes place at arbitrary, irregular timestamps, which may directly depend on the health condition of a patient. Hence, methods in discrete time rely on a data model that is not flexible enough to account for arbitrary, irregular monitoring and treatment times, because of which their suitability in medical practice is limited.

(2) **Neural methods in continuous time:** Only few neural methods have been developed for estimating CAPOs in *continuous time*. Yet, existing methods have key limitations. One stream of methods (Hess et al., 2024b; Vanderschueren et al., 2023) ignores time-varying confounding and is thus <u>not</u> applicable to our setting.

To the best of our knowledge, there is only one neural method that works in continuous time and that is applicable to our setting: TE-CDE (Seedat et al., 2022). This method tries to handle time-varying confounding through balancing. However, balancing is a heuristic approach to adjust for time-varying confounding; in fact, balancing was originally proposed for variance reduction (Johansson et al., 2016) and may even increase bias (Melnychuk et al., 2024). Therefore, TE-CDE suffers from an *infinite data bias* that comes from the fact that is does *not* properly adjust for time-varying confounding.

[1] Code is available at `https://github.com/konstantinhess/SCIP-Net`.

Research gap: To the best of our knowledge, none of the above neural methods performs proper adjustments for time-varying confounding in continuous time. As a remedy, we propose SCIP-Net, which is the first neural method that estimates CAPOs in continuous time while properly adjusting for time-varying confounding.

3 Problem Formulation

Notation: Let $[0, \tau]$ be the time window. In the following, we assume every stochastic process $V_t = V(t)$ defined on $[0, \tau]$ to be *càdlàg*, and we let $V_{t-} = \lim_{s \to t} V_s$ denote the left time limit. Further, we write $\bar{V}_t = \{V_s\}_{s \leq t}$ and $\underline{V}_t = \{V_s\}_{s \geq t}$.

Setup (see Fig. 1): We consider outcomes $Y_t \in \mathbb{R}^{d_y}$, discrete treatments $A_t \in \{0, 1\}^{d_a}$, and covariates $X_t \in \mathbb{R}^{d_x}$, where we assume that X_t contains Y_s for all $s < t$. Without loss of generality, we assume that static covariates are included in X_t. At time t, we let $\pi_{0,t}(A_t)$ and $\mu_{0,t}(X_t)$ denote the observational treatment propensity and the covariate distribution, respectively. Both measurement times and treatment times are typically dynamic and do not follow fixed schedules. Rather, both are recorded at *arbitrary, irregular timestamps*.

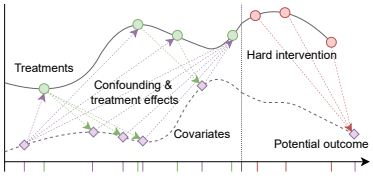


Figure 1: **Setup.** Shown are treatment and covariate trajectories in *continuous time*. Observational treatment assignments are confounded by covariates and, hence, estimating CAPOs requires adjustments.

To formalize the above in continuous time, we need to be able to model arbitrary timestamps, which increases the complexity compared to the discrete time setting considerably. Following Rytgaard et al. (2022; 2023), we let the counting processes N_t^x and N_t^a govern the times at which covariates are measured and at which treatments may be administered, respectively. For both $z \in \{a, x\}$, we let $\mathcal{T}_\tau^z = \{T_1^z, \ldots, T_{N_\tau^z}^z\}$ denote the set of jumping times of the process N_t^z with $T_{N_\tau^z}^z \leq \tau$. Further, we let Λ_0^z denote the cumulative intensity of N_t^z, and we let $\lambda_0^z(t)$ denote the corresponding intensity function. Further, we use the short-hand notation $\mathrm{d}t = [t, t + \mathrm{d}t)$ as in (Gill & Johansen, 1990; Rytgaard et al., 2022). Then, we have that

$$\lambda_0^z(t)\,\mathrm{d}t = \mathrm{d}\Lambda_0^z(t) = \mathbb{E}_{\mathbb{P}_0}[N^z(\mathrm{d}t)] = \mathbb{P}_0(N^z(\mathrm{d}t) = 1). \tag{1}$$

Observational likelihood: We write the full observed history up to time t as

$$\bar{H}_t = \{A_{T_j^a}, X_{T_j^x}, \bar{N}_t^a, \bar{N}_t^x : T_j^a \in \mathcal{T}_t^a, T_j^x \in \mathcal{T}_t^x\}. \tag{2}$$

Then, following (Rytgaard et al., 2022), we can write the observational likelihood of the data $\bar{H}_\tau$ via

$$\begin{aligned} \mathrm{d}\mathbb{P}_0(\bar{H}_\tau) = &\mu_{0,0}(X_0) \prod_{s \in (0,\tau]} \Big(\big(\mathrm{d}\Lambda_0^x(s \mid \bar{H}_{s-})\mu_{0,s}(X_s \mid \bar{H}_{s-})\big)^{N^x(\mathrm{d}s)} \big(1 - \mathrm{d}\Lambda_0^x(s \mid \bar{H}_{s-})\big)^{1-N^x(\mathrm{d}s)} \\ &\times \big(\mathrm{d}\Lambda_0^a(s \mid \bar{H}_{s-})\pi_{0,s}(A_s \mid \bar{H}_{s-})\big)^{N^a(\mathrm{d}s)} \big(1 - \mathrm{d}\Lambda_0^a(s \mid \bar{H}_{s-})\big)^{1-N^a(\mathrm{d}s)} \Big), \end{aligned} \tag{3}$$

where $\prod\!\!\!\!\!\!\int$ is the *geometric product integral*. Intuitively, the geometric product integral $\prod\!\!\!\!\!\!\int$ can be thought of as the infinitesimal limit of the discrete product operator Π. Importantly, the geometric product integral in Equation 3 is the natural way to describe joint likelihoods in *continuous time*. For more details, we provide a brief overview of product integration in Supp. B.

Objective: We are interested in estimating the response of the outcome variable Y_τ when intervening on the treatment sequence starting at time t, given an observed history $\bar{H}_{t-} = \bar{h}_{t-}$. For this, we adopt the potential outcomes framework (Neyman, 1923; Rubin, 1978). That is, we seek to estimate the *CAPO*

$$\mathbb{E}\Big[Y_\tau[\underline{a}_{*,t}, \underline{n}_{*,t}^a] \mid \bar{H}_{t-} = \bar{h}_{t-}\Big], \tag{4}$$

under interventions on both the treatment propensity $\underline{A}_t = \underline{a}_{*,t}$ and the treatment frequency $\underline{N}_t^a = \underline{n}_{*,t}^a$, given the history $\bar{H}_{t-} = \bar{h}_{t-}$. Here, the interventions are *hard interventions*, which is standard in the literature (e.g., Bica et al., 2020; Lim et al., 2018; Melnychuk et al., 2022; Seedat et al., 2022). That is, for $s \geq t$, we are interested in deterministic interventions of the form

$$A_s \sim \pi_{*,s}(A_s \mid \bar{H}_{t-} = \bar{h}_{t-}) = \mathbb{1}_{\{A_s = a_{*,s}\}}, \quad N_s^a \sim \mathrm{d}\Lambda_*^a(s \mid \bar{H}_{t-} = \bar{h}_{t-}) = \mathbb{1}_{\{t_{*,j}^a\}_{j=1}^J}(s)\,\mathrm{d}s, \tag{5}$$

where $a_{*,s} : \mathbb{R}^+ \to \mathcal{A}$ is a step-wise constant function with jumping points $\{t^a_{*,j}\}_{j=1}^J$.

Estimating the CAPO for a treatment sequence is notoriously challenging due to the fundamental problem of causal inference (Imbens & Rubin, 2015). This means, only factual outcomes Y_τ in are observed in the data, but not the potential outcomes when intervening on the treatment. In the following, we first define the interventional distribution for our objective and then ensure identifiability.

Interventional distribution: We now define the interventional distribution for our objective. For this, we rewrite Equation 4 by reweighting Y_τ under the observational distribution $\mathrm{d}\mathbb{P}_0$. For this, we follow Rytgaard et al. (2023) and first split the likelihood into two separate parts as

$$\mathrm{d}\mathbb{P}_0(\bar{H}_\tau) = \mathrm{d}\mathbb{P}_{Q_0,G_0}(\bar{H}_\tau) = \mu_{0,0}(X_0) \prod_{s\in(0,\tau]} \mathrm{d}G_{0,s}(\bar{H}_s)\, \mathrm{d}Q_{0,s}(\bar{H}_s), \tag{6}$$

where

$$\mathrm{d}G_{0,s}(\bar{H}_s) = \left(\mathrm{d}\Lambda^a_0(s \mid \bar{H}_{s-})\, \pi_{0,s}(A_s \mid \bar{H}_{s-})\right)^{N^a(\mathrm{d}s)} \left(1 - \mathrm{d}\Lambda^a_0(s \mid \bar{H}_{s-})\right)^{1-N^a(\mathrm{d}s)} \tag{7}$$

is the treatment part the we intervene on and where

$$\mathrm{d}Q_{0,s}(\bar{H}_s) = \left(\mathrm{d}\Lambda^x_0(s \mid \bar{H}_{s-})\mu_{0,s}(X_s \mid \bar{H}_{s-})\right)^{N^x(\mathrm{d}s)} \left(1 - \mathrm{d}\Lambda^x_0(s \mid \bar{H}_{s-})\right)^{1-N^x(\mathrm{d}s)} \tag{8}$$

remains unchanged. Then, we can write the interventional distribution as

$$\mathrm{d}\mathbb{P}_*(\bar{H}_\tau) = \mathrm{d}\mathbb{P}_{Q_0,G_*}(\bar{H}_\tau) = \mu_{0,0}(X_0) \prod_{s\in(0,\tau]} \mathrm{d}G_{*,s}(\bar{H}_s)\, \mathrm{d}Q_{0,s}(\bar{H}_s), \tag{9}$$

where

$$\mathrm{d}G_{*,s}(\bar{H}_s) = \left(\mathrm{d}\Lambda^a_*(s \mid \bar{H}_{s-})\, \pi_{*,s}(A_s \mid \bar{H}_{s-})\right)^{N^a(\mathrm{d}s)} \left(1 - \mathrm{d}\Lambda^a_*(s \mid \bar{H}_{s-})\right)^{1-N^a(\mathrm{d}s)}. \tag{10}$$

Identifiability: To ensure identifiability, we need to make the following assumptions (Lok, 2008; Robins & Hernán, 2009; Rytgaard et al., 2022) that are standard in the literature for estimating CAPOs over time (e.g., Seedat et al., 2022). *(i) Consistency:* Given an intervention on the treatment propensity and the frequency $[\underline{a}_{*,t}, \underline{n}^a_{*,t}]$, the observed outcome Y_τ coincides with the potential outcome $Y_\tau[\underline{a}_{*,t}, \underline{n}^a_{*,t}]$ under this intervention. *(ii) Positivity:* Given any history $\bar{H}_{t-}$, the Radon-Nikodỳm derivative $\mathrm{d}G_{*,t}/\mathrm{d}G_{0,t}$ exists. *(iii) Unconfoundedness:* Given any history $\bar{H}_{t-}$, the potential outcome is independent of the treatment assignment probability, that is, $Y_\tau[\underline{a}_{*,t}, \underline{n}^a_{*,t}] \perp (\underline{A}_t, \underline{N}^a_t) \mid \bar{H}_{t-}$.

Proposition 1. *Under assumptions (i)–(iii), we can estimate the CAPO from observational data (i.e., from data sampled under $\mathrm{d}\mathbb{P}_0$) via* ***inverse propensity weighting****, that is,*

$$\mathbb{E}\Big[Y_\tau[\underline{a}_{*,t}, \underline{n}^a_{*,t}] \;\Big|\; \bar{H}_{t-} = \bar{h}_{t-}\Big] = \mathbb{E}\Big[Y_\tau \prod_{s\geq t} W_s \;\Big|\; (\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,}), \bar{H}_{t-} = \bar{h}_{t-}\Big], \tag{11}$$

where the inverse propensity weights for $s \geq t$ are defined as

$$W_s \equiv w_s(\bar{H}_s) = \frac{\mathrm{d}G_{*,s}(\bar{H}_s)}{\mathrm{d}G_{0,s}(\bar{H}_s)}. \tag{12}$$

Proof. See Supp. D. □

Proposition 1 is important for the rest of our paper: it tells us that we can estimate CAPOs in continuous time from data sampled under $\mathrm{d}\mathbb{P}_0$. For this, we need to quantify the change in measure from the observational distribution $\mathrm{d}\mathbb{P}_0$ to the interventional distribution $\mathrm{d}\mathbb{P}_*$, which is given by Equation 12.

However, the above formulation is based on a product integral $\prod\!\!\!\!\int$, which is *not* computationally tractable. ⇒ We later derive a novel, *tractable* formulation where we rewrite Equation 11 using the product operator $\prod$ (Sec. 4.1). Further, inverse propensity weights W_s may lead to extreme weights and, hence, unstable performance. This is a known issue because settings over time are prone to low overlap (Frauen et al., 2025; Lim et al., 2018). ⇒ We later derive novel *stabilized weights* that are tailored to our continuous time setting (Sec. 4.2).

4 SCIP-NET

In this section, we introduce our SCIP-Net. It is designed to perform proper adjustments for time-varying confounding in continuous time.

Objective: Our objective is to find the optimal parameters[2] $\hat{\phi}$ of a neural network m_ϕ via

$$\hat{\phi} = \arg\min_{\phi} \mathbb{E}_{\mathbb{P}_*} \left[\left(Y_\tau[\underline{a}_{*,t}, \underline{n}^a_{*,t}] - m_\phi(\underline{A}_t, \underline{N}^a_t, \bar{H}_{t-}) \right)^2 \,\middle|\, \bar{H}_{t-} = \bar{h}_{t-} \right] \tag{13}$$

$$= \arg\min_{\phi} \mathbb{E}_{\mathbb{P}_0} \left[\left(Y_\tau - m_\phi(\underline{A}_t, \underline{N}^a_t, \bar{H}_{t-}) \right)^2 \prod_{s \geq t} W_s \,\middle|\, (\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t}), \bar{H}_{t-} = \bar{h}_{t-} \right]. \tag{14}$$

Note that the above objective makes use of inverse propensity weights W_s. However, these weights suffer from two drawbacks: they are (1) intractable as they rely on product integrals $\prod$, and (2) they may lead to unstable performance. As a remedy, we (1) derive a tractable expression for this product integral (Sec. 4.1), and we further (2) introduce *stabilized weights* (Sec. 4.2). Finally, we present our neural architecture (Sec. 4.3) and how to perform inference (Sec. 4.4).

4.1 REWRITING THE OBJECTIVE FOR COMPUTATIONAL TRACTABILITY

We now derive a tractable expression to compute our *unstabilized weights* W_s in Equation 12.

Proposition 2. *Let $t^a_{*,0} = t$ for notational convenience. The unstabilized weights in Equation 12 satisfy*

$$\prod_{s \geq t} W_s \,\Big|\, \left((\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t}), \bar{H}_{t-} = \bar{h}_{t-} \right) = \prod_{j=1}^{J} W_{t^a_{*,j}} \,\Big|\, \left((\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t}), \bar{H}_{t-} = \bar{h}_{t-} \right), \tag{15}$$

where

$$W_{t^a_{*,j}} = \frac{\exp \int_{s \in [t^a_{*,j-1}, t^a_{*,j})} \lambda^a_0(s \mid \bar{H}_{s-}) \, \mathrm{d}s}{\lambda^a_0(t^a_{*,j} \mid \bar{H}_{t^a_{*,j}-}) \, \pi_{0,t^a_{*,j}}(a_{*,t^a_{*,j}} \mid \bar{H}_{t^a_{*,j}-})}. \tag{16}$$

Proof. See Supp. D. □

Proposition 2 has important implications for tractability: Given a history $\bar{H}_{t-} = \bar{h}_{t-}$ and the future sequence of treatments $(\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t})$, the product integral $\prod$ of the unstabilized inverse propensity weights W_s reduces to a finite product $\prod$. This product includes both treatment propensities and treatment intensities. In Sec. 4.3, we show how to learn these quantities from data.

Importantly, the unstabilized weights in Equation 16 are already sufficient to adjust for time-varying confounding. However, as they may lead to extreme weights, we now propose *stabilized weights*.

4.2 STABILIZED WEIGHTS

In the following, we first define our *stabilized weights* (Def. 1). Then, we show that the optimal parameters $\hat{\phi}$ in Equation 14 are the same, regardless of whether the original, unstabilized inverse propensity weights from above are used or our stabilized weights (Proposition 3). Finally, we present a tractable expression to compute the stabilized weights (Proposition 4).

Definition 1. *For $s \geq t$, let the scaling factor Ξ_s be given by the ratio of the marginal transition probabilities of treatment, that is,*

$$\Xi_s \equiv \xi_s(\bar{A}_s, \bar{N}^a_s) = \frac{\mathrm{d}G_{0,s}(\bar{A}_s, \bar{N}^a_s)}{\mathrm{d}G_{*,s}(\bar{A}_s, \bar{N}^a_s)}. \tag{17}$$

We define the ***stabilized weights*** $\widetilde{W}_s$ *as*

$$\widetilde{W}_s = \Xi_s W_s. \tag{18}$$

[2]Throughout, we refer to the weights of neural nets as *parameters* to make the distinction to inverse propensity weights clear.

The idea of stabilized weights (e.g., Lim et al., 2018) is that the marginal transition probabilities Ξ_s, on average over the population, *downscale* the inverse propensity weights. Importantly, the scaling factors are not conditioned on the individual history and, therefore, do not change the objective.

To formalize this, we make use of the fact that the optimal parameters in Equation 14 are *invariant* to multiplicative scaling of the optimization problem with respect to constant scaling factors. We summarize this in the following proposition.

Proposition 3. *The optimal parameters $\hat{\phi}$ in Equation 14 can equivalently be obtained by*

$$\hat{\phi} = \arg\min_{\phi} \mathbb{E}_{\mathbb{P}_0}\left[\left(Y_\tau - m_\phi(\underline{A}_t, \underline{N}_t^a, \bar{H}_{t-})\right)^2 \prod_{s \geq t} \widetilde{W}_s \,\middle|\, (\underline{A}_t, \underline{N}_t^a) = (\underline{a}_{*,t}, \underline{n}_{*,t}^a), \bar{H}_{t-} = \bar{h}_{t-}\right]. \tag{19}$$

Proof. See Supp. D. □

Proposition 3 guarantees that we can substitute the original, unstabilized weights W_s with the stabilized version $\widetilde{W}_s$. As the scaling factors Ξ_s downscale W_s, we reduce the risk of receiving extreme inverse propensity weights, and, thus, obtain a more stable objective.

The results from Proposition 3 still rely on product integrals $\prod$. However, as in Proposition 2, we now derive an equivalent, tractable expression that relies on the product operator Π instead.

Proposition 4. *Let $t_{*,0}^a = t$ for notational convenience. The scaling factor Ξ_s from Equation 17 then satisfies*

$$\prod_{s \geq t} \Xi_s \,\Big|\, \left((\underline{A}_t, \underline{N}_t^a) = (\underline{a}_{*,t}, \underline{n}_{*,t}^a), (\bar{A}_{t-}, \bar{N}_{t-}^a) = (\bar{a}_{t-}, \bar{n}_{t-}^a)\right) \tag{20}$$

$$= \prod_{j=1}^{J} \Xi_{t_{*,j}^a} \,\Big|\, \left((\underline{A}_t, \underline{N}_t^a) = (\underline{a}_{*,t}, \underline{n}_{*,t}^a), (\bar{A}_{t-}, \bar{N}_{t-}^a) = (\bar{a}_{t-}, \bar{n}_{t-}^a)\right), \tag{21}$$

where

$$\Xi_{t_{*,j}^a} = \frac{\lambda_0^a(t_{*,j}^a \mid \bar{A}_{t_{*,j}^a -}, \bar{N}_{t_{*,j}^a -}^a)\, \pi_{0,t_{*,j}^a}(a_{t_{*,j}^a} \mid \bar{A}_{t_{*,j}^a -}, \bar{N}_{t_{*,j}^a -}^a)}{\exp \int_{s \in [t_{*,j-1}^a, t_{*,j}^a)} \lambda_0^a(s \mid \bar{A}_{s-}, \bar{N}_{s-}^a)\, \mathrm{d}s}. \tag{22}$$

Proof. See Supp. D. □

Together, Propositions 2, 3 and 4 yield a (1) tractable objective function that relies on (2) stabilized inverse propensity weights. As we show in the following Sec. 4.3, we can estimate the stabilized weights from data. Thereby, we present our SCIP-Net, which adjusts for time-varying confounding in continuous time.

4.3 Neural Architecture

Overview: We now introduce the neural architecture of our SCIP-Net, which consists of four components (see Fig. 2): The (S) **stabilization network** learns an estimator of the scaling factors $\xi_s(\cdot)$ from Equation 17. The (W) **weight network** learns an estimator of the unstabilized inverse propensity weights $w_s(\cdot)$ from Equation 12. Combining both, we have an estimator for the stabilized weights $\widetilde{w}_s(\cdot)$ from Equation 18. The (E) **encoder** learns a representation of the observed history, which is then passed to the decoder. Finally, the (D) **decoder** takes the learned representations and the stabilized inverse propensity weights as input to estimate the CAPOs $\mathbb{E}\left[Y_\tau[\underline{a}_{*,t}, \underline{n}_{*,t}^a] \mid \bar{H}_{t-} = \bar{h}_{t-}\right]$.

Backbones: All components (S), (W), (E), and (D) use neural controlled differential equations (CDEs) (Kidger et al., 2020; Morrill et al., 2021) as backbones. Neural CDEs have several benefits for our SCIP-Net. First, neural CDEs process data in continuous time. Second, neural CDEs update their hidden states as data becomes available over time. We provide a brief introduction in Supp. C.

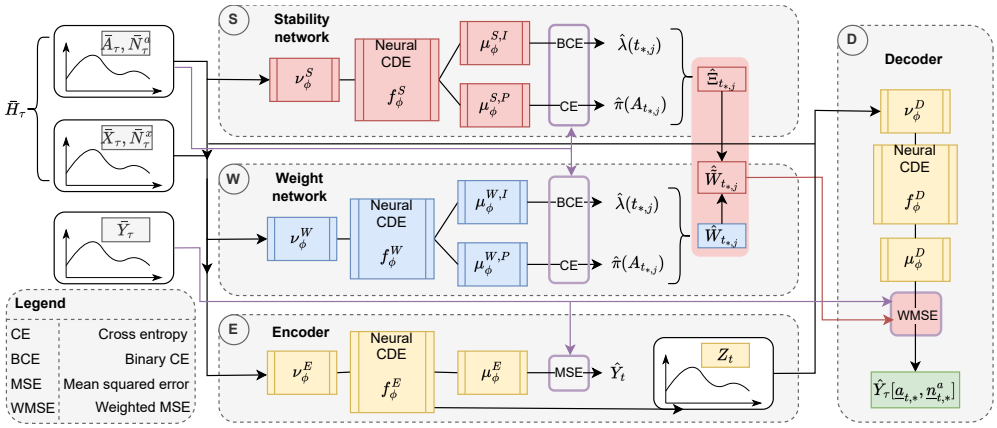


Figure 2: **Neural architecture of our SCIP-Net.**

In the following, we denote the *training samples* by $\{\bar{h}_{i,\tau}\}_{i=1}^n$ and the *test samples* by $\{(\bar{h}_{k,t-}, \underline{a}_{*,t}, \underline{n}^a_{*,t})\}_{k=1}^m$. Reassuringly, we emphasize that $\bar{h}_{i,t-}, \bar{h}_{k,t-}$ are realizations from the *observational distribution* $\mathrm{d}\mathbb{P}_0$, whereas $(\underline{a}_{*,t}, \underline{n}^a_{*,t})$ is the interventional treatment sequence.

(S) **Stability network:** The stability network learns an estimator $\hat{\xi}_s(\cdot)$ for the scaling factors $\Xi_{t^a_{*,j}}$ from Proposition 4. It consists of a linear input layer ν^S_ϕ, a neural vector field f^S_ϕ, and two linear output layers $\mu^{S,I}_\phi$ and $\mu^{S,P}_\phi$, which estimate the treatment intensity and propensity, respectively.

Training: The stability network receives treatment decisions and treatment times $(\bar{A}_\tau, \bar{N}^a_\tau) = (\bar{a}_{i,\tau}, \bar{n}^a_{i,\tau})$. We distinguish two cases. For $0 \leq t < \tau$, the latent representation $z^S_{i,t}$ evolves as

$$z^S_{i,t} = z^S_{i,0} + \int_0^t f^S_\phi(z_{i,s}, s)\, \mathrm{d}[a_{i,s-}, n^a_{i,s-}], \tag{23}$$

where $\mathrm{d}[a_{i,s-}, n^a_{i,s-}]$ denotes Riemann-Stieltjes integration with respect to the control path and $z^S_{i,0} = \nu^S_\phi(a_{i,0}, n^a_{i,0})$. *Case (1):* The latent representation is then passed to the intensity layer $\mu^{S,I}_\phi$ in order to estimate the probability whether a treatment decision is made at time t. For this, our SCIP-Net optimizes the binary cross entropy (BCE) loss

$$\mathcal{L}^{S,I}_t(\phi) = \mathrm{BCE}\Big(\mu^{S,I}_\phi(z^S_{i,t}), \mathrm{d}n^a_{i,t}\Big). \tag{24}$$

Thereby, the intensity layer $\mu^{S,I}_\phi$ learns an estimator of the treatment intensity function via

$$\hat{\lambda}^a_0(t \mid \bar{A}_{t-}, \bar{N}^a_{t-}) = \mu^{S,I}_\phi(Z^S_t). \tag{25}$$

Case (2): If a treatment decision is made at time $T^a_j = t^a_{i,j}$, the latent representation is additionally passed through the propensity output layer $\mu^{S,P}_\phi$ to estimate which treatment is administered. For this, our SCIP-Net minimizes the cross entropy (CE) loss

$$\mathcal{L}^{S,P}_j(\phi) = \mathrm{CE}\Big(\mu^{S,P}_\phi(z^S_{i,t^a_{i,j}}), a_{i,t^a_{i,j}}\Big), \tag{26}$$

Hence, the propensity layer learns an estimator of the propensity score via

$$\hat{\pi}^a_0(a_t \mid \bar{A}_{t-}, \bar{N}^a_{t-}) = \mu^{S,P}_\phi(Z^S_t). \tag{27}$$

Scaling factor: After training, the stability network *again* receives the training samples $(\bar{A}_\tau, \bar{N}^a_\tau) = (\bar{a}_{i,\tau}, \bar{n}^a_{i,\tau})$. Following Proposition 4, it then computes

$$\prod_{s \geq t} \hat{\xi}_s(\bar{a}_{i,s}, \bar{n}^a_{i,s}) \tag{28}$$

$$= \prod_{j=1}^J \frac{\hat{\lambda}^a_0(t^a_{i,j} \mid \bar{A}_{t^a_{i,j}-} = \bar{a}_{i,t^a_{i,j}-}, \bar{N}^a_{t^a_{i,j}-} = \bar{n}^a_{i,t^a_{i,j}-})\, \hat{\pi}_{0,t^a_{i,j}}(a_{t^a_{i,j}} \mid \bar{A}_{t^a_{i,j}-} = \bar{a}_{i,t^a_{i,j}-}, \bar{N}^a_{t^a_{i,j}-} = \bar{n}^a_{i,t^a_{i,j}-})}{\exp \int_{s \in [t^a_{i,j-1}, t^a_{i,j})} \hat{\lambda}^a_0(s \mid \bar{A}_{s-} = \bar{a}_{i,s-}, \bar{N}^a_s = \bar{n}^a_{i,s-})\, \mathrm{d}s},$$

where we can use an arbitrary quadrature scheme to compute the integral in the denominator.

(W) **Weight network:** The weight network learns an estimator $\hat{w}_s(\cdot)$ for the unstabilized weights $W_{t^a_{*,j}}$ from Proposition 2. It also consists of a linear input layer ν_ϕ^W, a neural vector field f_ϕ^W, and two linear output layers $\mu_\phi^{W,I}$ and $\mu_\phi^{W,P}$, which are trained to estimate the treatment intensity and propensity, respectively.

Training: The weight network receives samples $\bar{H}_\tau = \bar{h}_{i,\tau}$. After $h_{i,0}$ is transformed into $z_{i,0}^W$ via ν_ϕ^W, the latent state $z_{i,t}^W$ of the weight network evolves as

$$\mathrm{d}z_{i,t}^W = z_{i,0}^W + \int_0^t f_\phi^W(z_{i,s}^W, s)\,\mathrm{d}[h_{i,s-}], \tag{29}$$

where $\mathrm{d}[h_{i,s-}]$ denotes Riemann-Stieltjes integration w.r.t. the control path. *Case (1):* As for the stability network, the weight network estimates the probability of a treatment decision at time t through the intensity layer $\mu_\phi^{W,I}$ via

$$\mathcal{L}_t^{W,I} = \mathrm{BCE}\Big(\mu_\phi^{W,I}(z_{i,t}^W), \mathrm{d}n_{i,t}^a\Big), \tag{30}$$

and, hence, learns an estimator of the treatment intensity function via

$$\hat{\lambda}_0^a(t \mid \bar{H}_{t-}) = \mu_\phi^{T,I}(Z_t^W). \tag{31}$$

Case (2): If a treatment decision is made at time $T_j^a = t_{i,j}^a$, the latent state is also passed through the propensity layer $\mu_\phi^{W,P}$ and jointly trained via

$$\mathcal{L}_j^{W,P} = \mathrm{CE}\Big(\mu_\phi^{W,P}(z_{i,t_{i,j}^a}^W), a_{i,t_{i,j}^a}\Big), \tag{32}$$

such that our SCIP-Net learns an estimator of the propensity score as

$$\hat{\pi}_0^a(a_t \mid \bar{H}_{t-}) = \mu_\phi^{W,P}(Z_t^W). \tag{33}$$

Inverse propensity weight: After training, the weight network *again* receives the training samples $\bar{H}_\tau = \bar{h}_{i,\tau}$ and estimates the unstabilized inverse propensity weights according to Proposition 2 via

$$\prod_{s \geq t} \hat{w}_s(\bar{h}_{i,s-}) = \prod_{j=1}^J \frac{\exp \int_{s \in [t_{i,j-1}^a, t_{i,j}^a)} \hat{\lambda}_0^a(s \mid \bar{H}_{s-} = \bar{h}_{i,s-})\,\mathrm{d}s}{\hat{\lambda}_0^a(t_{i,j}^a \mid \bar{H}_{t_{i,j}^a-} = \bar{h}_{t_{i,j}^a-})\,\hat{\pi}_{0,t_{i,j}^a}(a_{i,t_{i,j}^a} \mid \bar{H}_{t_{*,j}^a-} = \bar{h}_{t_{i,j}^a-})}. \tag{34}$$

(E) **Encoder:** The encoder computes a latent representation of the history, which is then passed to the decoder. It consists of a linear input layer ν_ϕ^E, a neural vector field f_ϕ^E, and an output layer μ_ϕ^E.

Training: The encoder receives samples $\bar{H}_\tau = \bar{h}_{i,\tau}$. First, it transforms $\bar{h}_{i,0}$ into $z_{i,0}^E$ via ν_ϕ^E. The latent state $z_{i,t}^E$ then evolves according to

$$z_{i,t}^E = z_{i,0}^E + \int_0^t f_\phi^E(z_{i,s}^E, s)\,\mathrm{d}[h_{i,s-}], \tag{35}$$

where $\mathrm{d}[h_{i,s-}]$ denotes Riemann-Stieltjes integration w.r.t. the control path. At jumping times $n_{i,t}^x$, we pass the latent state $z_{i,t}$ to the encoder output layer μ_ϕ^E and minimize the mean squared error (MSE) loss for outcomes at the jumping times via

$$\mathcal{L}_t^E = \mathrm{MSE}\Big(\mu_\phi^E(z_{i,t}^E, a_{i,t}), y_{i,t}\Big). \tag{36}$$

(D) **Decoder:** The decoder receives as input: (i) the encoded history of the encoder and (ii) the future sequence of treatments. Then, it outputs an estimate of the CAPOs by adjusting for time-varying confounding. It has a linear input layer ν_ϕ^D, a neural vector field f_ϕ^D and an output layer μ_ϕ^D.

Training: During training, the decoder receives the final latent representation $z_{i,t}^E$ of the encoder as well as the observed treatments $(\underline{A}_t, \underline{N}_t^a) = (\underline{a}_{i,t}, \underline{n}_{i,t}^a)$. It then transforms the encoder representation through $z_{i,t}^D = \nu_\phi^D(z_{i,t}^E)$ and computes

$$z_{i,\tau}^D = z_{i,t}^D + \int_t^\tau f_\phi^D(z_{i,s}^D, s)\, \mathrm{d}[a_{i,s-}, n_{i,s-}^a], \tag{37}$$

where $\mathrm{d}[a_{i,s-}, n_{i,s-}^a]$ denotes Riemann-Stieltjes integration w.r.t. the control path. At time τ, we pass $z_{i,\tau}^D$ through the linear output layer μ_ϕ^D. Importantly, the decoder is trained by minimizing the MSE loss **weighted by the stabilized weights**, i.e.,

$$\mathcal{L}_\tau^D = \prod_{j=1}^J \hat{\tilde{w}}_{t_{i,j}^a} \mathrm{MSE}\Big(\mu_\phi^D(z_{i,\tau}^D, a_{i,\tau}), y_{i,\tau}\Big), \tag{38}$$

where

$$\hat{\tilde{w}}_{t_{i,j}^a} \equiv \hat{\tilde{w}}_{t_{i,j}^a}(\bar{h}_{i,t_{i,j}^a}) = \hat{\xi}_{t_{i,j}^a}(\bar{a}_{i,t_{i,j}^a}, \bar{n}_{i,t_{i,j}^a}^a)\hat{w}_{t_{i,j}^a}(\bar{h}_{i,t_{i,j}^a}), \tag{39}$$

using the stability network $\hat{\xi}_s(\cdot)$ and the weight network $\hat{w}_s(\cdot)$, respectively. By Proposition 3, we thereby target the optimal model parameters $\hat{\phi}$ which, unlike existing methods, explicitly adjust for time-varying confounding in continuous time.

4.4 Inference

In order to estimate CAPOs for an observed history $\bar{H}_{t-} = \bar{h}_{k,t-}$ and a future sequence of treatments $(\underline{A}_t, \underline{N}_t^a) = (\underline{a}_{*,t}, \underline{n}_{*,t}^a)$, we first encode the history via

$$z_{k,t}^E = z_{k,0}^E + \int_0^t f_\phi^E(z_{k,s}^E, s)\, \mathrm{d}[h_{k,s-}]. \tag{40}$$

This latent representation is then passed to the decoder along with $(\underline{a}_{*,t}, \underline{n}_{*,t}^a)$ in order to compute the final representation

$$z_{k,\tau}^D = \nu_\phi^D(z_{k,t}^E) + \int_t^\tau f_\phi^D(z_{k,s}^D, s)\, \mathrm{d}[a_{*,s-}, n_{*,s-}^a]. \tag{41}$$

Finally, the output layer μ_ϕ^D of the decoder is used to estimate the CAPO at time τ, given the history $\bar{H}_{t-} = \bar{h}_{k,t-}$, via

$$\hat{\mathbb{E}}\Big[Y_\tau[\underline{a}_{*,t}, \underline{n}_{*,t}^a] \mid \bar{H}_{t-} = \bar{h}_{k,t-}\Big] = \mu_\phi^D(z_{k,\tau}^D, a_{k,\tau}). \tag{42}$$

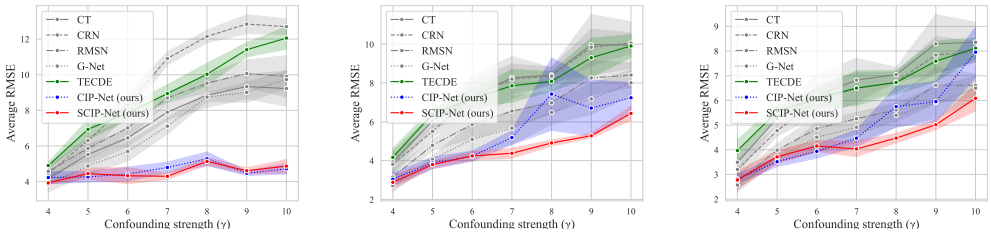
5 Numerical Experiments

Baselines: We now demonstrate the performance of our SCIP-Net against key neural baselines for estimating CAPOs over time (see Table 1). Importantly, our choice of baselines and datasets is consistent with prior literature (e.g., Bica et al., 2020; Lim et al., 2018; Melnychuk et al., 2022; Seedat et al., 2022). Further, we report the performance of the CIP-Net ablation, where we directly train the decoder with the unstabilized weights. This allows us to understand the performance gain of our stabilized weights. Note that the CIP-Net ablation is still a new method (as no other neural method performs proper adjustments for time-varying confounding in continuous time).

Datasets: We use a (i) synthetic dataset based on a **tumor growth model** (Geng et al., 2017), and a (ii) semi-synthetic dataset based on the **MIMIC-III dataset** (Johnson et al., 2016). For both datasets, the outcomes are simulated, so that we have access to the ground-truth *potential* outcomes, which allows for comparing the performance in terms of root mean squared error (RMSE). We report the mean $\pm$ the standard deviation over five runs with different seeds. We perform rigorous hyperparameter tuning for all baselines to ensure a fair comparison (see Supp. G).

Tumor growth data: Tumor growth models are widely used for estimating CAPOs over time. It is a model for the evolution of lung cancer Y_t under radio therapy treatment A_t^c and chemo therapy treatment A_t^r (details in Supp. E.1). Here, we follow Vanderschueren et al. (2023), where observation

times are at arbitrary, irregular timestamps. However, our setup has two differences: (i) We are not primarily interested in informative sampling times. Instead, we consider a scenario with observation times completely at random. Hence, observation times follow a Hawkes process with constant intensity. (ii) Further, we do not consider an RCT setting. Instead, treatment assignments are confounded via $A_t^c, A_t^r \sim \text{Ber}((\gamma / D_{\text{max}}(\bar{D}_{15}(\bar{Y}_{t-1} - \bar{D}_{\text{max}}/2))$, where γ controls the confounding strength, and D_{max} and $\bar{D}_{15}$ are the maximum and average tumor diameter over the last 15 days, respectively. For evaluation, we then generate *ground truth potential outcomes* under hard interventions that would be unobserved in real-world data. For this, we randomly sample treatment sequences, irrespective of the history, as is done in (Melnychuk et al., 2022). Below, we vary the prediction horizons (up to three days ahead) and the confounding strength.We provide details in Supp. E.1.



(a) One day ahead prediction (b) Two days ahead prediction (c) Three days ahead prediction

Figure 3: **Performance for the tumor growth model.** We compare different forecast horizons and different confound strengths. Shown is the average RMSE of the potential outcomes under hard interventions over five seeds. *Our proposed SCIP-Net performs best, followed by our CIP-Net.*

The results are in Fig. 3. We make the following observations: **(1)** Our proposed SCIP-Net performs best. The performance gains become especially obvious for increasing levels of time-varying confounding because ours is the first method to perform proper adjustments in continuous time. **(2)** Our proposed SCIP-Net performs more robust than the CIP-Net ablation, which demonstrates the effectiveness of our stabilized weights over the unstabilized weights. The stabilized weights help estimating more accurately especially for larger prediction horizons and strong confounding. **(3)** Nevertheless, our CIP-Net, has a competitive performance. **(4)** The only baseline designed for continuous time is TE-CDE (shown in green), which we outperform by a large margin. **(5)** All other neural baselines are instead designed for discrete time (shown in gray) and are outperformed clearly.

MIMIC-III data: Our experiments are based on the MIMIC-III extract from Wang et al. (2020). Here, we use real-world covariates at irregular measurement timestamps, and then simulate treatments and outcomes, respectively. This is done analogous to (Melnychuk et al., 2022), such that we have access to the ground-truth potential outcomes. For the outcome variable, we additionally apply a random observation mask in order to mimic observations at arbitrary, irregular timestamps. We provide more details in Supp. E.2.

Prediction window	CT	CRN	RMSNs	G-Net	TE-CDE	**CIP-Net** (ours)	**SCIP-Net** (ours)	**Rel. improvement**
$(\tau - t) = 1$ hours	1.052 ± 0.069	1.049 ± 0.065	1.075 ± 0.074	1.021 ± 0.069	0.915 ± 0.025	$\mathbf{0.876 \pm 0.041}$	0.877 ± 0.044	+4.1%
$(\tau - t) = 2$ hours	1.196 ± 0.272	1.088 ± 0.374	1.130 ± 0.274	1.095 ± 0.335	0.784 ± 0.145	0.785 ± 0.117	$\mathbf{0.634 \pm 0.148}$	+19.1%
$(\tau - t) = 3$ hours	1.444 ± 0.232	1.262 ± 0.355	1.300 ± 0.304	1.330 ± 0.198	1.240 ± 0.242	1.291 ± 0.400	$\mathbf{1.089 \pm 0.322}$	+12.2%

Table 2: **Performance for MIMIC-III.** Reported is the average RMSE of the potential outcomes under hard interventions over five seeds. We highlight the relative improvement of our SCIP-Net over existing baselines. Again, *our proposed SCIP-Net performs best*.

Table 2 shows the results for different prediction windows. **(1)** Our SCIP-Net has the lowest error among all methods and performs thus again best. **(2)** Our CIP-Net ablation, which is a new method in itself, again has highly competitive performance. **(3)** Yet, the ablation shows that using stabilized weights as in SCIP-Net (as opposed to the unstabilized weights as in CIP-Net) makes a large contribution to the overall performance.

Conclusion: To the best of our knowledge, SCIP-Net is the first neural method for *estimating conditional average potential outcomes through proper adjustments for time-varying confounding in continuous time*. For this, we first derive a *tractable expression for inverse propensity weighting* in continuous time. Then, we propose *stabilized weights* in continuous time to stabilize the training objective. Our experiments show that our SCIP-Net has clear benefits over existing baselines when observation times and treatment times take place at arbitrary, irregular timestamps.

REFERENCES

Ahmed Allam, Stefan Feuerriegel, Michael Rebhan, and Michael Krauthammer. Analyzing patient trajectories with artificial intelligence. *Journal of Medical Internet Research*, 23(12):e29812, 2021.

Heejung Bang and James M. Robins. Doubly robust estimation in missing data and causal inference models. *Biometrics*, 61(4):962–973, 2005.

Agamirza E. Bashirov, Emine Mısırlı, Yücel Tandoğdu, and Ali Özyapıcı. On modeling with multiplicative differential equations. *Applied Mathematics - A Journal of Chinese Universities*, 26:425 – 438, 2011.

Samuel L. Battalio, David E. Conroy, Walter Dempsey, Peng Liao, Marianne Menictas, Susan Murphy, Inbal Nahum-Shani, Tianchen Qian, Santosh Kumar, and Bonnie Spring. Sense2Stop: A micro-randomized trial using wearable sensors to optimize a just-in-time-adaptive stress management intervention for smoking relapse prevention. *Contemporary Clinical Trials*, 109:106534, 2021.

Inci Baytas, Cao Xiao, Xi Zhang, Fei Wang, Anil Jain, and Jiayu Zhou. Patient subtyping via time-aware lstm networks. In *KDD*, 2017.

Ioana Bica, Ahmed M. Alaa, James Jordon, and Mihaela van der Schaar. Estimating counterfactual treatment outcomes over time through adversarially balanced representations. In *ICLR*, 2020.

Ioana Bica, Ahmed M. Alaa, Craig Lambert, and Mihaela van der Schaar. From real-world patient data to individualized treatment effects using machine learning: Current and future methods to address underlying challenges. *Clinical Pharmacology and Therapeutics*, 109(1):87–100, 2021.

Edward De Brouwer, Jaak Simm, Adam Arany, and Yves Moreau. GRU-ODE-Bayes: Continuous modeling of sporadically-observed time series. In *NeurIPS*, 2019.

Zhengping Che, Sanjay Purushotham, KyungHyun Cho, David Sontag, and Yan Liu. Recurrent neural networks for multivariate time series with missing values. *Scientific Reports*, 8(1):6085, 2018.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *NeurIPS*, 2018.

Yuqi Chen, Kan Ren, Yansen Wang, Yuchen Fang, Weiwei Sun, and Dongsheng Li. Contiformer: Continuous-time transformer for irregular time series modeling. In *NeurIPS*, volume 36, 2023.

Alicia Curth and Mihaela van der Schaar. Nonparametric estimation of heterogeneous treatment effects: From theory to learning algorithms. In *AISTATS*, 2021.

Nan Du, Hanjun Dai, Rakshit Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. Recurrent marked temporal point processes: Embedding event history to vector. In *KDD*, 2016.

Stefan Feuerriegel, Dennis Frauen, Valentyn Melnychuk, Jonas Schweisthal, Konstantin Hess, Alicia Curth, Stefan Bauer, Niki Kilbertus, Isaac S. Kohane, and Mihaela van der Schaar. Causal machine learning for predicting treatment outcomes. *Nature Medicine*, 30:958–968, 2024.

Dennis Frauen, Tobias Hatt, Valentyn Melnychuk, and Stefan Feuerriegel. Estimating average causal effects from patient trajectories. In *AAAI*, 2023.

Dennis Frauen, Konstantin Hess, and Stefan Feuerriegel. Model-agnostic meta-learners for estimating heterogeneous treatment effects over time. In *ICLR*, 2025.

Crispin W. Gardiner. *Handbook of stochastic methods for physics, chemistry and the natural sciences*. Springer-Verlag, Berlin; Heidelberg, 1985. ISBN 3540156070.

Changran Geng, Harald Paganetti, and Clemens Grassberger. Prediction of treatment response for combined chemo- and radiation therapy for non-small cell lung cancer patients using a biomathematical model. *Scientific Reports*, 7(1):13542, 2017.

Richard D. Gill and Soren Johansen. A survey of product-integration with a view toward application in survival analysis. *Annals of Statistics*, 18(4), 1990.

Eldad Haber and Lars Ruthotto. Stable architectures for deep neural networks. *Inverse Problems*, 34(1):014004, 2018.

Tobias Hatt and Stefan Feuerriegel. Sequential deconfounding for causal inference with unobserved confounders. *arXiv preprint*, arXiv:2104.09323, 2021a.

Tobias Hatt and Stefan Feuerriegel. Estimating average treatment effects via orthogonal regularization. In *CIKM*, 2021b.

Konstantin Hess, Dennis Frauen, Valentyn Melnychuk, and Stefan Feuerriegel. G-transformer for estimating conditional average potential outcomes over time. *arXiv preprint*, arXiv:2405.21012, 2024a.

Konstantin Hess, Valentyn Melnychuk, Dennis Frauen, and Stefan Feuerriegel. Bayesian neural controlled differential equations for treatment effect estimation. In *ICLR*, 2024b.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8): 1735–1780, 1997.

Çağlar Hızlı, ST John, Anne Juuti, Tuure Saarinen, Kirsi Pietiläinen, and Pekka Marttinen. Causal modeling of policy interventions from sequences of treatments and outcomes. In *ICML*, 2023.

Guido W. Imbens and Donald B. Rubin. *Causal inference for statistics, social, and biomedical sciences: An introduction*. Cambridge University Press, Cambridge, 2015. ISBN 9781139025751.

Fredrik D. Johansson, Uri Shalit, and David Sonntag. Learning representations for counterfactual inference. In *ICML*, 2016.

Alistair E. W. Johnson, Tom J. Pollard, Lu Shen, Li-wei H. Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G. Mark. MIMIC-III, a freely accessible critical care database. *Scientific Data*, 3(1):160035, 2016.

Nathan Kallus, Xiaojie Mao, and Angela Zhou. Interval estimation of individual-level causal effects under unobserved confounding. In *AISTATS*, 2019.

Patrick Kidger, James Morrill, James Foster, and Terry Lyons. Neural controlled differential equations for irregular time series. In *NeurIPS*, 2020.

Patrick Kidger, James Foster, Xuechen Li, Harald Oberhauser, and Terry Lyons. Neural SDEs as Infinite-Dimensional GANs. In *ICML*, 2021.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Milan Kuzmanovic, Tobias Hatt, and Stefan Feuerriegel. Deconfounding temporal autoencoder: Estimating treatment effects over time using noisy proxies. In *ML4H*, 2021.

Rui Li, Stephanie Hu, Mingyu Lu, Yuria Utsumi, Prithwish Chakraborty, Daby M. Sow, Piyush Madan, Jun Li, Mohamed Ghalwash, Zach Shahn, and Li-wei Lehman. G-Net: A recurrent network approach to G-computation for counterfactual prediction under a dynamic treatment regime. In *ML4H*, 2021.

Bryan Lim, Ahmed M. Alaa, and Mihaela van der Schaar. Forecasting treatment responses over time using recurrent marginal structural networks. In *NeurIPS*, 2018.

Judith J. Lok. Statistical modeling of causal effects in continuous time. *Annals of Statistics*, 36(3), 2008.

Yiping Lu, Aoxiao Zhong, Quanzheng Li, and Bin Dong. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. In *ICML*, 2018.

Yuchen Ma, Valentyn Melnychuk, Jonas Schweisthal, and Stefan Feuerriegel. DiffPO: A causal diffusion model for learning distributions of potential outcomes. In *NeurIPS*, 2024.

Farokh Marvasti. *Nonuniform sampling*. Springer US, 2001, 2001. ISBN 9781461512295.

Valentyn Melnychuk, Dennis Frauen, and Stefan Feuerriegel. Causal transformer for estimating counterfactual outcomes. In *ICML*, 2022.

Valentyn Melnychuk, Dennis Frauen, and Stefan Feuerriegel. Bounds on representation-induced confounding bias for treatment effect estimation. In *ICLR*, 2024.

James Morrill, Patrick Kidger, Lingyi Yang, and Terry Lyons. Neural controlled differential equations for online prediction tasks. *arXiv preprint*, 2106.11028, 2021.

Elizabeth Murray, Eric B. Hekler, Gerhard Andersson, Linda M. Collins, Aiden Doherty, Chris Hollis, Daniel E. Rivera, Robert West, and Jeremy C. Wyatt. Evaluating Digital Health Interventions: Key Questions and Approaches. *American Journal of Preventive Medicine*, 51(5):843–851, 2016.

Jerzy Neyman. On the application of probability theory to agricultural experiments. *Annals of Agricultural Sciences*, 10:1–51, 1923.

Yilmazcan Özyurt, Mathias Kraus, Tobias Hatt, and Stefan Feuerriegel. AttDMM: An attentive deep Markov model for risk scoring in intensive care units. In *KDD*. 2021.

James M. Robins. A new approach to causal inference in mortality studies with a sustained exposure period: Application to control of the healthy worker survivor effect. *Mathematical Modelling*, 7: 1393–1512, 1986.

James M. Robins. Robust estimation in sequentially ignorable missing data and causal inference models. *Proceedings of the American Statistical Association on Bayesian Statistical Science*, pp. 6–10, 1999.

James M. Robins and Miguel A. Hernán. *Estimation of the causal effects of time-varying exposures*. Chapman & Hall/CRC handbooks of modern statistical methods. CRC Press, Boca Raton, 2009.

Kjetil Røysland. A martingale approach to continuous-time marginal structural models. *Bernoulli*, 17(3):895 – 915, 2011.

Donald B. Rubin. Inference and missing data. *Biometrika*, 63(3):581, 1976.

Donald B. Rubin. Bayesian inference for causal effects: The role of randomization. *Annals of Statistics*, 6(1):34–58, 1978.

Helene C. Rytgaard, Thomas A. Gerds, and Mark J. van der Laan. Continuous-time targeted minimum loss-based estimation of intervention-specific mean outcomes. *The Annals of Statistics*, 2022.

Helene C. Rytgaard, Frank Eriksson, and Mark J van der Laan. Estimation of time-specific intervention effects on continuously distributed time-to-event outcomes by targeted maximum likelihood estimation. *Biometrics*, 79(4):3038–3049, 2023.

Maresa Schröder, Dennis Frauen, Jonas Schweisthal, Konstantin Hess, Valentyn Melnychuk, and Stefan Feuerriegel. Conformal prediction for causal effects of continuous treatments. *arXiv preprint*, arXiv:2407.03094, 2024.

Peter Schulam and Suchi Saria. Reliable decision support using counterfactual models. In *NeurIPS*, 2017.

Nabeel Seedat, Fergus Imrie, Alexis Bellot, Zhaozhi Qian, and Mihaela van der Schaar. Continuous-time modeling of counterfactual outcomes using neural controlled differential equations. In *ICML*, 2022.

Uri Shalit, Fredrik D. Johansson, and David Sontag. Estimating individual treatment effect: Generalization bounds and algorithms. In *ICML*, 2017.

Oleksandr Shchur, Ali Caner Türkmen, Tim Januschowski, and Stephan Günnemann. Neural temporal point processes: A review. In *IJCAI*, 2021.

Yi Shirakawa, Toru; Li, Yulun Wu, Sky Qiu, Yuxuan Li, Mingduo Zhao, Hiroyasu Iso, and Mark van der Laan. Longitudinal targeted minimum loss-based estimation with temporal-difference heterogeneous transformer. In *ICML*, 2024.

Hossein Soleimani, Adarsh Subbaswamy, and Suchi Saria. Treatment-response models for counterfactual reasoning with continuous-time, continuous-valued interventions. In *UAI*, 2017.

Mark J. van der Laan and Susan Gruber. Targeted minimum loss based estimation of causal effects of multiple time point interventions. *The International Journal of Biostatistics*, 8(1):Article 9, 2012.

Toon Vanderschueren, Alicia Curth, Wouter Verbeke, and Mihaela van der Schaar. Accounting for informative sampling when learning to forecast treatment outcomes over time. In *ICML*, 2023.

Shirly Wang, Matthew B.A. McDermott, Geeticka Chauhan, Marzyeh Ghassemi, Michael C. Hughes, and Tristan Naumann. MIMIC-extract: A data extraction, preprocessing, and representation pipeline for MIMIC-III. In *CHIL*, 2020.

Yanbo Xu, Yanxun Xu, and Suchi Saria. A non-parametric bayesian approach for estimating treatment-response curves from sparse time series. In *ML4H*, 2016.

A EXTENDED RELATED WORK

Irregular time series: Outside of the causal literature, stochastic processes in continuous time and irregular time series are a heavily studied area of research, in particular in terms of stochastic processes (Gardiner, 1985). Directly related to the continuous time literature are non-uniformly spaced observations and their impact on inference (Marvasti, 2001). Irregular observation times can also be seen as a special case of missing data (Rubin, 1976). More recently, neural network architectures handling irregularity involve explicit incorporation of time gaps into recurrent units (Baytas et al., 2017), or otherwise handling missing values in recurrent neural networks (Che et al., 2018; Brouwer et al., 2019). Further, transformers have been adapted for irregular time series (Chen et al., 2023) and neural point processes Du et al. (2016); Shchur et al. (2021). Finally, and directly related to our work, neural controlled differential equations (Kidger et al., 2020; 2021; Morrill et al., 2021) can directly handle irregular time series by design. However, these works focus on traditional estimation and *not* causal inference tasks. In other words, the above methods solve a *different* task and would be *biased* in our setting.

Conditional average potential outcomes in the static setting: A large body of research focuses on methods for estimating conditional average potential outcomes in the static setting (e.g., Curth & van der Schaar, 2021; Hatt & Feuerriegel, 2021b; Johansson et al., 2016; Kallus et al., 2019; Ma et al., 2024; Schröder et al., 2024; Shalit et al., 2017). However, they cannot adequately address the complexities of time-varying data, which are crucial in healthcare applications where patient conditions evolve over time (e.g., EHRs (Allam et al., 2021; Bica et al., 2021), wearable devices (Battalio et al., 2021; Murray et al., 2016)).

Nonparametric methods in continuous time: Some nonparametric methods have been proposed (Hatt & Feuerriegel, 2021a; Hızlı et al., 2023; Schulam & Saria, 2017; Soleimani et al., 2017; Xu et al., 2016), but they suffer from scalability issues. Further, some struggle with static and high-dimensional covariates, complex outcome distributions, or impose additional identifiability assumptions. As a result, we focus on neural methods for their flexibility and scalability.

Additional research directions: Hess et al. (2024b) propose a neural method for uncertainty quantification of conditional average potential outcomes in continuous time. Further, Vanderschueren et al. (2023) develop a framework to account for bias due to informative observation times. Both approaches are *orthogonal* to our work and do *not* focus on time-varying confounding. Further, additional research directions in discrete time consider nonparametric learners (Frauen et al., 2025) and robust estimation of the G-computation formula (Hess et al., 2024a). Other works focus on learning treatment effects from noisy proxies (Kuzmanovic et al., 2021). Finally, there are efforts to leverage neural methods for estimating average potential outcomes (Frauen et al., 2023; Shirakawa et al., 2024). However, these methods are *not* applicable to our setting, as the do *not* operate in continuous time and do *not* focus on estimating CAPOs.

B PRODUCT INTEGRATION

Product integration is a useful tool for describing stochastic processes and their joint likelihoods in continuous time (e.g., in survival analysis; see Gill & Johansen, 1990; Rytgaard et al., 2022; 2023). In the following, we provide a brief introduction to product integration. Importantly, the type that we refer to is also known as the *geometric integral* or the *multiplicative integral* (Bashirov et al., 2011).

The product integral $\prod$ can intuitively be thought of as the infinitesimal limit of the product operator Π, or, equivalently, as the multiplicative version of the Riemann-integral $\int$. That is, the standard Riemann-integral is defined as

$$\int\limits_{[a,b]} f(x)\,\mathrm{d}x = \lim_{\Delta x \to 0} \sum_i f(\bar{x}_i)\Delta x, \tag{43}$$

where $\Delta x = x_{i+1} - x_i$ and $\bar{x} \in (x_i, x_{i+1})$ for a partition $\bigcup_i [x_i, x_{i+1}]$ of $[a, b]$.

The **product-integral** is analogously defined as

$$\prod\limits_{[a,b]} f(x)^{\mathrm{d}x} = \lim_{\Delta x \to 0} \prod_i f(\bar{x}_i)^{\Delta x}. \tag{44}$$

For the proofs in Supp. D, we make use of the identity

$$\prod\limits_{[a,b]} f(x)^{\mathrm{d}x} = \exp\Big(\int\limits_{[a,b]} \log f(x)\,\mathrm{d}x\Big), \tag{45}$$

which holds since

$$\prod\limits_{[a,b]} f(x)^{\mathrm{d}x} = \lim_{\Delta x \to 0} \prod_i f(\bar{x})^{\Delta x} \tag{46}$$

$$= \lim_{\Delta x \to 0} \exp\Big(\log \prod_i f(\bar{x}_i)^{\Delta x}\Big) \tag{47}$$

$$= \lim_{\Delta x \to 0} \exp\Big(\sum_i \log f(\bar{x}_i)^{\Delta x}\Big) \tag{48}$$

$$= \exp\Big(\lim_{\Delta x \to 0} \sum_i \log f(\bar{x}_i)^{\Delta x}\Big) \tag{49}$$

$$= \exp\Big(\int\limits_{[a,b]} \log f(x)\,\mathrm{d}x\Big). \tag{50}$$

Similar to the additivity of the standard integral $\int$, the product integral $\prod$ is *multiplicative*. That is, we have that for $a < b < c$

$$\prod\limits_{[a,c]} f(x)^{\mathrm{d}x} = \prod\limits_{[a,b]} f(x)^{\mathrm{d}x} \prod\limits_{[b,c]} f(x)^{\mathrm{d}x}, \tag{51}$$

which can be shown with the above exp-log identity

$$\prod_{[a,c]} f(x)^{\mathrm{d}x} = \exp\Big(\int_{[a,c]} \log f(x)\,\mathrm{d}x\Big) \tag{52}$$

$$= \exp\Big(\int_{[a,b]} \log f(x)\,\mathrm{d}x + \int_{[b,c]} \log f(x)\,\mathrm{d}x\Big) \tag{53}$$

$$= \exp\Big(\int_{[a,b]} \log f(x)\,\mathrm{d}x\Big) \exp\Big(\int_{[b,c]} \log f(x)\,\mathrm{d}x\Big) \tag{54}$$

$$= \prod_{[a,b]} f(x)^{\mathrm{d}x} \prod_{[b,c]} f(x)^{\mathrm{d}x}. \tag{55}$$

C NEURAL DIFFERENTIAL EQUATIONS

We provide a brief summary of neural ordinary differential equations and neural controlled differential equations, similar to that in (Hess et al., 2024b).

Neural ODEs: Neural ordinary differential equations (ODEs) (Chen et al., 2018; Haber & Ruthotto, 2018; Lu et al., 2018) integrate neural networks with ordinary differential equations. In a neural ODE, the neural network $f_\phi(\cdot)$ defines the vector field of the initial value problem

$$Z_t = \int_0^t f_\phi(Z_s, s)\,\mathrm{d}s, \quad Z_0 = X. \tag{56}$$

Thereby, a neural ODE captures the continuous evolution of hidden states Z_t over a time scale. Thereby, it learns a continuous flow of transformations, where the input $X = Z_0$ is passed through an ODE solver to obtain the output $\hat{Y} = Z_\tau$ (possibly, after another output transformation).

While neural ODEs are generally suitable for describing a continuous time evolution, they have an important limitation: *all data need to be captured in the initial value*. Hence, they are not capable of updating their hidden states as new data becomes available over time, as is the case for, e.g., electronic health records.

Neural CDEs: Neural controlled differential equations (CDEs) (Kidger et al., 2020) overcome the above limitation. Put simply, one can think of them as a continuous-time counterpart to recurrent neural networks. Given a path of data $X_t \in \mathbb{R}^{d_x}$, $t \in [0, \tau]$, a neural CDE consists of an embedding network $\nu_\phi(\cdot)$, a readout network $\mu_\phi(\cdot)$, and a neural vector field f_ϕ. Then, the neural CDE is defined as

$$\hat{Y} = \mu_\phi(Z_\tau), \quad Z_t = \int_0^t f_\phi(Z_s, s)\,\mathrm{d}[X_s], \quad t \in (0, \tau] \quad \text{with} \quad Z_0 = \nu_\phi(X_0), \tag{57}$$

where $Z_t \in \mathbb{R}^{d_z}$ and $f_\phi(Z_t, t) \in \mathbb{R}^{d_z \times d_x}$. The integral here is a *Riemann-Stieltjes integral*, where $f_\phi(Z_s, s)\,\mathrm{d}[X_s]$ corresponds to matrix multiplication. In this context, the neural differential equation is *controlled* by the process $[X_s]$. Importantly, one can rewrite it (under some regularity conditions) as

$$Z_t = \int_0^t f_\phi(Z_s, s)\frac{\mathrm{d}X_s}{\mathrm{d}s}\,\mathrm{d}s, \quad t \in (0, \bar{T}]. \tag{58}$$

Calculating the time derivative requires a C^1-representation of the data X_t for all $t \in [0, \tau]$. As a result, irregularly sampled observations $((t_0, X_0), (t_1, X_1), \ldots, (t_n, X_n)$ must be interpolated over time, producing a continuous representation X_t. Here, we can use any interpolation scheme such as linear interpolation (Morrill et al., 2021) as in TE-CDE (Seedat et al., 2022). Then, the neural CDE essentially reduces to a neural ODE for optimization.

The **key difference** to neural ODEs is, however, that the neural vector field is controlled by sequentially incoming data and, therefore, updates the hidden states as data becomes available over time. This is a **clear advantage over ODEs**, which require capturing all data in the initial value and because of which neural ODEs are *not* suitable.

D PROOFS OF PROPOSITIONS

Proposition 1. *Under assumptions (i)–(iii), we can estimate the conditional average potential outcome from observational data (i.e., from data sampled under* $\mathrm{d}\mathbb{P}_0$*) via* ***inverse propensity weighting****, that is,*

$$\mathbb{E}\Big[Y_\tau[\underline{a}_{*,t}, \underline{n}^a_{*,t}] \;\Big|\; \bar{H}_{t-} = \bar{h}_{t-}\Big] = \mathbb{E}\Big[Y_\tau \prod_{s\geq t} W_s \;\Big|\; (\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,}), \bar{H}_{t-} = \bar{h}_{t-}\Big], \tag{59}$$

where the inverse propensity weights for $s \geq t$ *are defined as*

$$W_s \equiv w_s(\bar{H}_s) = \frac{\mathrm{d}G_{*,s}(\bar{H}_s)}{\mathrm{d}G_{0,s}(\bar{H}_s)}. \tag{60}$$

Proof. The proof follows similar ideas as in (Rytgaard et al., 2022):

$$\mathbb{E}\Big[Y_\tau[\underline{a}_{*,t}, \underline{n}^a_{*,t}] \mid \bar{H}_{t-} = \bar{h}_{t-}\Big] \tag{61}$$

$$= \int y_\tau[\underline{a}_{*,t}, \underline{n}^a_{*,t}] \prod_{s\geq t} \mathrm{d}Q_{0,s}(\bar{h}_s)\, \mathrm{d}G_{0,s}(\bar{h}_s) \tag{62}$$

$$= \int y_\tau[\underline{a}_{*,t}, \underline{n}^a_{*,t}] \prod_{s\geq t} \mathrm{d}Q_{0,s}(\bar{h}_s)\, \mathrm{d}G_{*,s}(\bar{h}_s) \tag{63}$$

$$\underbrace{=}_{\text{(iii)}} \int y_\tau[\underline{a}_{*,t}, \underline{n}^a_{*,t}] \prod_{s\geq t} \mathrm{d}Q_{0,s}(\bar{h}_s \mid (\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t}))\, \mathrm{d}G_{*,s}(\bar{h}_s \mid (\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t})) \tag{64}$$

$$\underbrace{=}_{\text{(i)}} \int y_\tau \prod_{s\geq t} \mathrm{d}Q_{0,s}(\bar{h}_s \mid (\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t}))\, \mathrm{d}G_{*,s}(\bar{h}_s \mid (\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t})) \tag{65}$$

$$\underbrace{=}_{\text{(ii)}} \int y_\tau \prod_{s\geq t} \frac{\mathrm{d}G_{*,s}(\bar{h}_s \mid (\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t}))}{\mathrm{d}G_{0,s}(\bar{h}_s \mid (\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t}))}$$

$$\times \prod_{s\geq t} \mathrm{d}Q_{0,s}(\bar{h}_s \mid (\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t}))\, \mathrm{d}G_{0,s}(\bar{h}_s \mid (\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t})) \tag{66}$$

$$= \mathbb{E}\Big[Y_\tau \prod_{s\geq t} w_s(\bar{H}_s \mid (\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t})) \;\Big|\; (\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t}), \bar{H}_{t-} = \bar{h}_{t-}\Big] \tag{67}$$

$$= \mathbb{E}\Big[Y_\tau \prod_{s\geq t} w_s(\bar{H}_s) \;\Big|\; (\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t}), \bar{H}_{t-} = \bar{h}_{t-}\Big] \tag{68}$$

$$= \mathbb{E}\Big[Y_\tau \prod_{s\geq t} W_s \;\Big|\; (\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t}), \bar{H}_{t-} = \bar{h}_{t-}\Big], \tag{69}$$

where Equation 66 uses the multiplicativity of the product integral. □

Proposition 2. *Let $t^a_{*,0} = t$ for notational convenience. The unstabilized weights in Equation 12 satisfy*

$$\prod_{s \geq t} W_s \,\Big|\, \Big((\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t}), \bar{H}_{t-} = \bar{h}_{t-} \Big) = \prod_{j=1}^{J} W_{t^a_{*,j}} \,\Big|\, \Big((\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t}), \bar{H}_{t-} = \bar{h}_{t-} \Big), \tag{70}$$

where

$$W_{t^a_{*,j}} = \frac{\exp \int_{s \in [t^a_{*,j-1}, t^a_{*,j})} \lambda^a_0(s \mid \bar{H}_{s-}) \, \mathrm{d}s}{\lambda^a_0(t^a_{*,j} \mid \bar{H}_{t^a_{*,j}-}) \, \pi_{0,t^a_{*,j}}(a_{*,t^a_{*,j}} \mid \bar{H}_{t^a_{*,j}-})}. \tag{71}$$

Proof. The inverse propensity weight in Equation 12 is by definition given by

$$\prod_{s \geq t} \frac{\mathrm{d}G_{*,s}(\bar{H}_s)}{\mathrm{d}G_{0,s}(\bar{H}_s)} = \prod_{s \geq t} \left(\frac{\mathrm{d}\Lambda^a_*(s \mid \bar{H}_{s-}) \pi_{*,s}(A_s \mid \bar{H}_{s-})}{\mathrm{d}\Lambda^a_0(s \mid \bar{H}_{s-}) \pi_{0,s}(A_s \mid \bar{H}_{s-})} \right)^{N^a(\mathrm{d}s)} \left(\frac{1 - \mathrm{d}\Lambda^a_*(s \mid \bar{H}_{s-})}{1 - \mathrm{d}\Lambda^a_0(s \mid \bar{H}_{s-})} \right)^{1 - N^a(\mathrm{d}s)}. \tag{72}$$

We simplify the first part via

$$\prod_{s \geq t} \left(\frac{\mathrm{d}\Lambda^a_*(s \mid \bar{H}_{s-}) \pi_{*,s}(A_s \mid \bar{H}_{s-})}{\mathrm{d}\Lambda^a_0(s \mid \bar{H}_{s-}) \pi_{0,s}(A_s \mid \bar{H}_{s-})} \right)^{N^a(\mathrm{d}s)} \tag{73}$$

$$= \prod_{s \geq t} \left(\frac{\lambda^a_*(s \mid \bar{H}_{s-}) \pi_{*,s}(A_s \mid \bar{H}_{s-}) \, \mathrm{d}s}{\lambda^a_0(s \mid \bar{H}_{s-}) \pi_{0,s}(A_s \mid \bar{H}_{s-}) \, \mathrm{d}s} \right)^{N^a(\mathrm{d}s)} \tag{74}$$

$$= \exp \left[\int_{s \geq t} \log \left(\frac{\lambda^a_*(s \mid \bar{H}_{s-}) \pi_{*,s}(A_s \mid \bar{H}_{s-})}{\lambda^a_0(s \mid \bar{H}_{s-}) \pi_{0,s}(A_s \mid \bar{H}_{s-})} \right) N^a(\mathrm{d}s) \right] \tag{75}$$

$$= \exp \left[\sum_{T^a_j \in \mathcal{T}^a_\tau \setminus \mathcal{T}^a_t} \log \left(\frac{\lambda^a_*(T^a_j \mid \bar{H}_{T^a_j-}) \pi_{*,T^a_j}(A_{T^a_j} \mid \bar{H}_{T^a_j-})}{\lambda^a_0(T^a_j \mid \bar{H}_{T^a_j-}) \pi_{0,T^a_j}(A_{T^a_j} \mid \bar{H}_{T^a_j-})} \right) \right] \tag{76}$$

$$= \prod_{T^a_j \in \mathcal{T}^a_\tau \setminus \mathcal{T}^a_t} \frac{\lambda^a_*(T^a_j \mid \bar{H}_{T^a_j-}) \pi_{*,T^a_j}(A_{T^a_j} \mid \bar{H}_{T^a_j-})}{\lambda^a_0(T^a_j \mid \bar{H}_{T^a_j-}) \pi_{0,T^a_j}(A_{T^a_j} \mid \bar{H}_{T^a_j-})}. \tag{77}$$

Now, since we condition on $\underline{N}^a_t = \underline{n}^a_{*,t}$, we know that the jumping times are given by

$$\mathcal{T}^a_\tau \setminus \mathcal{T}^a_t = \{t^a_{*,j}\}_{j=1}^J. \tag{78}$$

Hence, we have that

$$\prod_{T^a_j \in \mathcal{T}^a_\tau} \frac{\lambda^a_*(T^a_j \mid \bar{H}_{T^a_j-}) \pi_{*,T^a_j}(A_{T^a_j} \mid \bar{H}_{T^a_j-})}{\lambda^a_0(T^a_j \mid \bar{H}_{T^a_j-}) \pi_{0,T^a_j}(A_{T^a_j} \mid \bar{H}_{T^a_j-})} \,\Big|\, \Big((\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t}), \bar{H}_{t-} = \bar{h}_{t-} \Big) \tag{79}$$

$$= \prod_{j=1}^{J} \frac{\lambda^a_*(t^a_{*,j} \mid \bar{H}_{t^a_{*,j}-}) \pi_{*,t^a_{*,j}}(a_{*,t^a_{*,j}} \mid \bar{H}_{t^a_{*,j}-})}{\lambda^a_0(t^a_{*,j} \mid \bar{H}_{t^a_{*,j}-}) \pi_{0,t^a_{*,j}}(a_{*,t^a_{*,j}} \mid \bar{H}_{t^a_{*,j}-})} \,\Big|\, \Big((\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t}), \bar{H}_{t-} = \bar{h}_{t-} \Big) \tag{80}$$

$$= \prod_{j=1}^{J} \Big(\lambda^a_0(t^a_{*,j} \mid \bar{H}_{t^a_{*,j}-}) \pi_{0,t^a_{*,j}}(a_{*,t^a_{*,j}} \mid \bar{H}^a_{t^a_{*,j}-}) \Big)^{-1} \,\Big|\, \Big((\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t}), \bar{H}_{t-} = \bar{h}_{t-} \Big). \tag{81}$$

Further, we can simplify the second part via

$$\prod_{s\geq t}\left(\frac{1-\mathrm{d}\Lambda_*^a(s\mid\bar{H}_{s-})}{1-\mathrm{d}\Lambda_0^a(s\mid\bar{H}_{s-})}\right)^{1-N^a(\mathrm{d}s)} \tag{82}$$

$$=\prod_{s\geq t}\left(\frac{1-\mathbb{P}_*(N^a(\mathrm{d}s)=1\mid\bar{H}_{s-})}{1-\mathbb{P}_0(N^a(\mathrm{d}s)=1\mid\bar{H}_{s-})}\right)^{1-N^a(\mathrm{d}s)} \tag{83}$$

$$=\lim_{K\to\infty}\prod_{k=1}^{K}\left(\frac{\mathbb{P}_*(N^a([t_k,t_{k+1}))=0\mid\bar{H}_{t_k-})}{\mathbb{P}_0(N^a([t_k,t_{k+1}))=0\mid\bar{H}_{t_k-})}\right)^{1-N^a([t_k,t_{k+1}))} \tag{84}$$

$$=\exp\Big[-\int_{s\geq t}\lambda_*^a(s\mid\bar{H}_{s-})\,\mathrm{d}s\Big]/\exp\Big[-\int_{s\geq t}\lambda_0^a(s\mid\bar{H}_{s-})\,\mathrm{d}s\Big] \tag{85}$$

$$=\exp\Big[\int_{s\geq t}\lambda_0^a(s\mid\bar{H}_{s-})\,\mathrm{d}s-\int_{s\geq t}\lambda_*^a(s\mid\bar{H}_{s-})\,\mathrm{d}s\Big], \tag{86}$$

where $\bigcup_{k=1}^{K}[t_k,t_{k+1})$ is a disjoint partition of $[t,\tau]$ with $\lim_{K\to\infty}\max_{k\leq K}(t_{k+1}-t_k)=0$. Finally, again by conditioning on $\underline{N}_t^a=\underline{n}_{*,t}^a$, we have that

$$\int_{s\geq t}\lambda_*^a(s\mid\bar{H}_{s-})\,\mathrm{d}s\;\Big|\;\Big(\underline{N}_{*,t}^a=\underline{n}_{*,t}^a\Big)=\int_{s\geq t}\mathbb{1}_{\{t_{*,j}^a\}_{j=1}^J}(s)\,\mathrm{d}s\;\Big|\;\Big(\underline{N}_{*,t}^a=\underline{n}_{*,t}^a\Big)=0, \tag{87}$$

which leaves

$$\prod_{s\geq t}\left(\frac{1-\mathrm{d}\Lambda_*^a(s\mid\bar{H}_{s-})}{1-\mathrm{d}\Lambda_0^a(s\mid\bar{H}_{s-})}\right)^{1-N^a(\mathrm{d}s)}\;\Big|\;\Big((\underline{A}_t,\underline{N}_t^a)=(\underline{a}_{*,t},\underline{n}_{*,t}^a),\bar{H}_{t-}=\bar{h}_{t-}\Big) \tag{88}$$

$$=\exp\Big[\int_{s\geq t}\lambda_0^a(s\mid\bar{H}_{s-})\,\mathrm{d}s\Big]\;\Big|\;\Big((\underline{A}_t,\underline{N}_t^a)=(\underline{a}_{*,t},\underline{n}_{*,t}^a),\bar{H}_{t-}=\bar{h}_{t-}\Big). \tag{89}$$

Combining Equation 81 with Equation 89, the unstabilized weights are then given by

$$\prod_{s\geq t}W_s\;\Big|\;\Big((\underline{A}_t,\underline{N}_t^a)=(\underline{a}_{*,t},\underline{n}_{*,t}^a),\bar{H}_{t-}=\bar{h}_{t-}\Big) \tag{90}$$

$$=\prod_{s\geq t}\frac{\mathrm{d}G_{*,s}(\bar{H}_s)}{\mathrm{d}G_{0,s}(\bar{H}_s)}\;\Big|\;\Big((\underline{A}_t,\underline{N}_t^a)=(\underline{a}_{*,t},\underline{n}_{*,t}^a),\bar{H}_{t-}=\bar{h}_{t-}\Big) \tag{91}$$

$$\begin{aligned}&=\exp\Big[\int_{s\geq t}\lambda_0^a(s\mid\bar{H}_{s-})\,\mathrm{d}s\Big]\\&\quad\times\prod_{j=1}^{J}\Big(\lambda_0^a(t_{*,j}^a\mid\bar{H}_{t_{*,j}^a-})\pi_{0,t_{*,j}^a}(a_{*,t_{*,j}^a}\mid\bar{H}_{t_{*,j}^a-})\Big)^{-1}\;\Big|\;\Big((\underline{A}_t,\underline{N}_t^a)=(\underline{a}_{*,t},\underline{n}_{*,t}^a),\bar{H}_{t-}=\bar{h}_{t-}\Big)\end{aligned} \tag{92}$$

$$=\prod_{j=1}^{J}\frac{\exp\int_{s\in[t_{*,j-1}^a,t_{*,j}^a)}\lambda_0^a(s\mid\bar{H}_{s-})\,\mathrm{d}s}{\lambda_0^a(t_{*,j}^a\mid\bar{H}_{t_{*,j}^a-})\pi_{0,t_{*,j}^a}(a_{*,t_{*,j}^a}\mid\bar{H}_{t_{*,j}^a-})}\;\Big|\;\Big((\underline{A}_t,\underline{N}_t^a)=(\underline{a}_{*,t},\underline{n}_{*,t}^a),\bar{H}_{t-}=\bar{h}_{t-}\Big) \tag{93}$$

$$=\prod_{j=1}^{J}W_{t_{*,j}^a}\;\Big|\;\Big((\underline{A}_t,\underline{N}_t^a)=(\underline{a}_{*,t},\underline{n}_{*,t}^a),\bar{H}_{t-}=\bar{h}_{t-}\Big). \tag{94}$$

□

Definition 1. *For $s \geq t$, let the scaling factor Ξ_s be given by the ratio of the marginal transition probabilities of treatment, that is,*

$$\Xi_s \equiv \xi_s(\bar{A}_s, \bar{N}_s^a) = \frac{\mathrm{d}G_{0,s}(\bar{A}_s, \bar{N}_s^a)}{\mathrm{d}G_{*,s}(\bar{A}_s, \bar{N}_s^a)}. \tag{95}$$

We define the ***stabilized weights*** $\widetilde{W}_s$ *as*

$$\widetilde{W}_s = \Xi_s W_s. \tag{96}$$

Proposition 3. *The optimal parameters $\hat{\phi}$ in Equation 14 can equivalently be obtained by*

$$\hat{\phi} = \arg\min_{\phi} \mathbb{E}_{\mathbb{P}_0}\left[\left(Y_\tau - m_\phi(\underline{A}_t, \underline{N}_t^a, \bar{H}_{t-})\right)^2 \prod_{s \geq t} \widetilde{W}_s \,\middle|\, (\underline{A}_t, \underline{N}_t^a) = (\underline{a}_{*,t}, \underline{n}_{*,t}^a), \bar{H}_{t-} = \bar{h}_{t-}\right]. \tag{97}$$

Proof. The scaling factor is a constant conditionally on the future and past treatment propensity and frequency, i.e.,

$$\begin{aligned}
&\Xi_s \mid (\underline{A}_t, \underline{N}_t^a) = (\underline{a}_{*,t}, \underline{n}_{*,t}^a), \bar{H}_{t-} = \bar{h}_{t-} && (98)\\
=&\Xi_s \mid (\underline{A}_t, \underline{N}_t^a) = (\underline{a}_{*,t}, \underline{n}_{*,t}^a), (A_{t-}, \bar{N}_{t-}^a) = (\bar{a}_{t-}, \bar{n}_{t-}^a) && (99)\\
=&\xi_s([\bar{a}_{*,s}, \bar{a}_{t-}], [\bar{n}_{*,s}^a, \bar{n}_{t-}^a]) && (100)\\
\equiv&\text{const.}, && (101)
\end{aligned}$$

where we use

$$[\bar{a}_{*,s}, \bar{a}_{t-}] = \Big(\bigcup_{t \leq r \leq t} a_{*,r}\Big) \cup \Big(\bigcup_{r<t} a_r\Big) \quad \text{and} \quad [\underline{n}_{*,s}^a, \underline{n}_{*,t-}^a] = \Big(\bigcup_{t \leq r \leq t} n_{*,r}^a\Big) \cup \Big(\bigcup_{r<t} n_r^a\Big) \tag{102}$$

for the concatenation of interventional and observational treatments at time $s \geq t$. Hence, with $\Xi_s \in \mathbb{R}^+$, we can use linearity of the expectation and multiplicativity of the product integral, such that

$$\begin{aligned}
&\arg\min_{\phi} \mathbb{E}_{\mathbb{P}_0}\left[\left((Y_\tau - m_\phi(\underline{A}_t, \underline{N}_t^a, \bar{H}_{t-}))^2 \prod_{s \geq t} \widetilde{W}_s\right) \,\middle|\, (\underline{A}_t, \underline{N}_t^a) = (\underline{a}_{*,t}, \underline{n}_{*,t}^a), \bar{H}_{t-} = \bar{h}_{t-}\right] && (103)\\
=&\arg\min_{\phi} \mathbb{E}_{\mathbb{P}_0}\left[\left((Y_\tau - m_\phi(\underline{A}_t, \underline{N}_t^a, \bar{H}_{t-}))^2 \prod_{s \geq t} \Xi_s W_s\right) \,\middle|\, (\underline{A}_t, \underline{N}_t^a) = (\underline{a}_{*,t}, \underline{n}_{*,t}^a), \bar{H}_{t-} = \bar{h}_{t-}\right] && (104)\\
=&\arg\min_{\phi} \Bigg\{\Big(\prod_{s \geq t} \xi_s([\bar{a}_{*,s}, \bar{a}_{t-}], [\bar{n}_{*,s}^a, \bar{n}_{t-}^a])\Big) && (105)\\
&\times \Big(\mathbb{E}_{\mathbb{P}_0}\Big[(Y_\tau - m_\phi(\underline{A}_t, \underline{N}_t^a, \bar{H}_{t-}))^2 \prod_{s \geq t} W_s \,\Big|\, (\underline{A}_t, \underline{N}_t^a) = (\underline{a}_{*,t}, \underline{n}_{*,t}^a), \bar{H}_{t-} = \bar{h}_{t-}\Big]\Big)\Bigg\} && (106)\\
=&\arg\min_{\phi} \mathbb{E}_{\mathbb{P}_0}\Big[(Y_\tau - m_\phi(\underline{A}_t, \underline{N}_t^a, \bar{H}_{t-}))^2 \prod_{s \geq t} W_s \,\Big|\, (\underline{A}_t, \underline{N}_t^a) = (\underline{a}_{*,t}, \underline{n}_{*,t}^a), \bar{H}_{t-} = \bar{h}_{t-}\Big] && (107)\\
=&\hat{\phi}. && (108)
\end{aligned}$$

□

Proposition 4. *Let $t^a_{*,0} = t$ for notational convenience. The scaling factor Ξ_s from Equation 17 then satisfies*

$$\prod_{s \geq t} \Xi_s \,\Big|\, \Big((\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t}), (\bar{A}_{t-}, \bar{N}^a_{t-}) = (\bar{a}_{t-}, \bar{n}^a_{t-}) \Big) \tag{109}$$

$$= \prod_{j=1}^{J} \Xi_{t^a_{*,j}} \,\Big|\, \Big((\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t}), (\bar{A}_{t-}, \bar{N}^a_{t-}) = (\bar{a}_{t-}, \bar{n}^a_{t-}) \Big), \tag{110}$$

where

$$\Xi_{t^a_{*,j}} = \frac{\lambda^a_0(t^a_{*,j} \mid \bar{A}_{t^a_{*,j}-}, \bar{N}^a_{t^a_{*,j}-})\, \pi_{0,t^a_{*,j}}(a_{t^a_{*,j}} \mid \bar{A}_{t^a_{*,j}-}, \bar{N}^a_{t^a_{*,j}-})}{\exp \int_{s \in [t^a_{*,j-1}, t^a_{*,j})} \lambda^a_0(s \mid \bar{A}_{s-}, \bar{N}^a_{s-})\, \mathrm{d}s}. \tag{111}$$

Proof. For the proof, we follow the steps as in the proof of Proposition 2.

By definition, Ξ_s satisfies

$$\prod_{s \geq t} \Xi_s = \prod_{s \geq t} \frac{\mathrm{d}G_{0,s}(\bar{A}_s, \bar{N}^a_s)}{\mathrm{d}G_{*,s}(\bar{A}_s, \bar{N}^a_s))} \tag{112}$$

$$= \prod_{s \geq t} \left(\frac{\mathrm{d}\Lambda^a_0(s \mid \bar{A}_{s-}, \bar{N}^a_{s-}))\pi_{0,s}(A_s \mid \bar{A}_{s-}, \bar{N}^a_{s-})}{\mathrm{d}\Lambda^a_*(s \mid \bar{A}_{s-}, \bar{N}^a_{s-})\pi_{*,s}(A_s \mid \bar{A}_{s-}, \bar{N}^a_{s-})} \right)^{N^a(\mathrm{d}s)} \left(\frac{1 - \mathrm{d}\Lambda^a_0(s \mid \bar{A}_{s-}, \bar{N}^a_{s-})}{1 - \mathrm{d}\Lambda^a_*(s \mid \bar{A}_{s-}, \bar{N}^a_{s-})} \right)^{1 - N^a(\mathrm{d}s)} \tag{113}$$

First, we simplify

$$\prod_{s \geq t} \left(\frac{\mathrm{d}\Lambda^a_0(s \mid \bar{A}_{s-}, \bar{N}^a_{s-})\pi_{0,s}(A_s \mid \bar{A}_{s-}, \bar{N}^a_{s-})}{\mathrm{d}\Lambda^a_*(s \mid \bar{A}_{s-}, \bar{N}^a_{s-})\pi_{*,s}(A_s \mid \bar{A}_{s-}, \bar{N}^a_{s-})} \right)^{N^a(\mathrm{d}s)} \tag{114}$$

$$= \prod_{s \geq t} \left(\frac{\lambda^a_0(s \mid \bar{A}_{s-}, \bar{N}^a_{s-})\pi_{0,s}(A_s \mid \bar{A}_{s-}, \bar{N}^a_{s-})\, \mathrm{d}s}{\lambda^a_*(s \mid \bar{A}_{s-}, \bar{N}^a_{s-})\pi_{*,s}(A_s \mid \bar{A}_{s-}, \bar{N}^a_{s-})\, \mathrm{d}s} \right)^{N^a(\mathrm{d}s)} \tag{115}$$

$$= \exp \left[\int_{s \geq t} \log \left(\frac{\lambda^a_0(s \mid \bar{A}_{s-}, \bar{N}^a_{s-})\pi_{0,s}(A_s \mid \bar{A}_{s-}, \bar{N}^a_{s-})}{\lambda^a_*(s \mid \bar{A}_{s-}, \bar{N}^a_{s-})\pi_{*,s}(A_s \mid \bar{A}_{s-}, \bar{N}^a_{s-})} \right) N^a(\mathrm{d}s) \right] \tag{116}$$

$$= \exp \left[\sum_{T^a_j \in \mathcal{T}^a_\tau \setminus \mathcal{T}^a_t} \log \left(\frac{\lambda^a_0(T^a_j \mid \bar{A}_{T^a_j-}, \bar{N}^a_{T^a_j-})\pi_{0,T^a_j}(A_{T^a_j} \mid \bar{A}_{T^a_j-}, \bar{N}^a_{T^a_j-})}{\lambda^a_*(T^a_j \mid \bar{A}_{T^a_j-}, \bar{N}^a_{T^a_j-})\pi_{*,T^a_j}(A_{T^a_j} \mid \bar{A}_{T^a_j-}, \bar{N}^a_{T^a_j-})} \right) \right] \tag{117}$$

$$= \prod_{T^a_j \in \mathcal{T}^a_\tau \setminus \mathcal{T}^a_t} \frac{\lambda^a_0(T^a_j \mid \bar{A}_{T^a_j-}, \bar{N}^a_{T^a_j-})\pi_{0,T^a_j}(A_{T^a_j} \mid \bar{A}_{T^a_j-}, \bar{N}^a_{T^a_j-})}{\lambda^a_*(T^a_j \mid \bar{A}_{T^a_j-}, \bar{N}^a_{T^a_j-})\pi_{*,T^a_j}(A_{T^a_j} \mid \bar{A}_{T^a_j-}, \bar{N}^a_{T^a_j-})}. \tag{118}$$

Conditionally on $\underline{N}^a_t = \underline{n}^a_{*,t}$, the jumping times are fixed, i.e.,

$$\mathcal{T}^a_\tau \setminus \mathcal{T}^a_t = \{t^a_{*,j}\}_{j=1}^J. \tag{119}$$

Therefore, it follows that

$$\prod_{T^a_j \in \mathcal{T}^a_\tau} \frac{\lambda^a_0(T^a_j \mid \bar{A}_{T^a_j-}, \bar{N}^a_{T^a_j-})\pi_{0,T^a_j}(A_{T^a_j} \mid \bar{A}_{T^a_j-}, \bar{N}^a_{T^a_j-})}{\lambda^a_*(T^a_j \mid \bar{A}_{T^a_j-}, \bar{N}^a_{T^a_j-})\pi_{*,T^a_j}(A_{T^a_j} \mid \bar{A}_{T^a_j-}, \bar{N}^a_{T^a_j-})} \,\Big|\, \Big((\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t}), (\bar{A}_{t-}, \bar{N}^a_{t-}) = (\bar{a}_{t-}, \bar{n}^a_{t-}) \Big) \tag{120}$$

$$= \prod_{j=1}^{J} \frac{\lambda^a_0(t^a_{*,j} \mid \bar{A}_{t^a_{*,j}-}, \bar{N}^a_{t^a_{*,j}-})\pi_{0,t^a_{*,j}}(a_{*,t^a_{*,j}} \mid \bar{A}_{t^a_{*,j}-}, \bar{N}^a_{t^a_{*,j}-})}{\lambda^a_*(t^a_{*,j} \mid \bar{A}_{t^a_{*,j}-}, \bar{N}^a_{t^a_{*,j}-})\pi_{*,t^a_{*,j}}(a_{*,t^a_{*,j}} \mid \bar{A}_{t^a_{*,j}-}, \bar{N}^a_{t^a_{*,j}-})} \,\Big|\, \Big((\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t}), (\bar{A}_{t-}, \bar{N}^a_{t-}) = (\bar{a}_{t-}, \bar{n}^a_{t-}) \Big) \tag{121}$$

$$= \prod_{j=1}^{J} \lambda^a_0(t^a_{*,j} \mid \bar{A}_{t^a_{*,j}-}, \bar{N}^a_{t^a_{*,j}-})\pi_{0,t^a_{*,j}}(a_{*,t^a_{*,j}} \mid \bar{A}_{t^a_{*,j}-}, \bar{N}^a_{t^a_{*,j}-}) \,\Big|\, \Big((\underline{A}_t, \underline{N}^a_t) = (\underline{a}_{*,t}, \underline{n}^a_{*,t}), (\bar{A}_{t-}, \bar{N}^a_{t-}) = (\bar{a}_{t-}, \bar{n}^a_{t-}) \Big). \tag{122}$$

For the second part, we also follow the steps as in Proposition 2, that is,

$$\prod_{s\geq t}\left(\frac{1-\mathrm{d}\Lambda_0^a(s\mid \bar{A}_{s-},\bar{N}_{s-}^a)}{1-\mathrm{d}\Lambda_*^a(s\mid \bar{A}_{s-},\bar{N}_{s-}^a)}\right)^{1-N^a(\mathrm{d}s)} \tag{123}$$

$$=\prod_{s\geq t}\left(\frac{1-\mathbb{P}_0(N^a(\mathrm{d}s)=1\mid \bar{A}_{s-},\bar{N}_{s-})}{1-\mathbb{P}_*(N^a(\mathrm{d}s)=1\mid \bar{A}_{s-},\bar{N}_{s-})}\right)^{1-N^a(\mathrm{d}s)} \tag{124}$$

$$=\lim_{K\to\infty}\prod_{k=1}^{K}\left(\frac{\mathbb{P}_0(N^a([t_k,t_{k+1}))=0\mid \bar{A}_{t_k-},\bar{N}_{t_k-})}{\mathbb{P}_*(N^a([t_k,t_{k+1}))=0\mid \bar{A}_{t_k-},\bar{N}_{t_k-})}\right)^{1-N^a([t_k,t_{k+1}))} \tag{125}$$

$$=\exp\Big[-\int_{s\geq t}\lambda_0^a(s\mid \bar{A}_{s-},\bar{N}_{s-}^a)\,\mathrm{d}s\Big]/\exp\Big[-\int_{s\geq t}\lambda_*^a(s\mid \bar{A}_{s-},\bar{N}_{s-}^a)\,\mathrm{d}s\Big] \tag{126}$$

$$=\exp\Big[\int_{s\geq t}\lambda_*^a(s\mid \bar{A}_{s-},\bar{N}_{s-}^a)\,\mathrm{d}s-\int_{s\geq t}\lambda_0^a(s\mid \bar{A}_{s-},\bar{N}_{s-}^a)\,\mathrm{d}s\Big], \tag{127}$$

where $\bigcup_{k=1}^{K}[t_k,t_{k+1})$ is a disjoint partition of $[t,\tau]$ with $\lim_{K\to\infty}\max_{k\leq K}(t_{k+1}-t_k)=0$. Finally, again by conditioning on $\underline{N}_t^a=\underline{n}_{*,t}^a$, we have that

$$\int_{s\geq t}\lambda_*^a(s\mid \bar{A}_{s-},\bar{N}_{s-})\,\mathrm{d}s\;\Big|\;\Big(\underline{N}_t^a=\underline{n}_{*,t}^a\Big)=\int_{s\geq t}\mathbb{1}_{\{t_{*,j}^a\}_{j=1}^{J}}(s)\,\mathrm{d}s\;\Big|\;\Big(\underline{N}_t^a=\underline{n}_{*,t}^a\Big)=0, \tag{128}$$

which leaves

$$\prod_{s\geq t}\left(\frac{1-\mathrm{d}\Lambda_0^a(s\mid \bar{A}_{s-},\bar{N}_{s-})}{1-\mathrm{d}\Lambda_*^a(s\mid \bar{A}_{s-},\bar{N}_{s-})}\right)^{1-N^a(\mathrm{d}s)}\;\Big|\;\Big((\underline{A}_t,\underline{N}_t^a)=(\underline{a}_{*,t},\underline{n}_{*,t}^a),(\bar{A}_{t-},\bar{N}_{t-}^a)=(\bar{a}_{t-},\bar{n}_{t-}^a)\Big) \tag{129}$$

$$=\exp\Big[-\int_{s\geq t}\lambda_0^a(s\mid \bar{A}_{s-},\bar{N}_{s-})\,\mathrm{d}s\Big]\;\Big|\;\Big((\underline{A}_t,\underline{N}_t^a)=(\underline{a}_{*,t},\underline{n}_{*,t}^a),(\bar{A}_{t-},\bar{N}_{t-}^a)=(\bar{a}_{t-},\bar{n}_{t-}^a)\Big). \tag{130}$$

Finally, we combine equation 122 with equation 130. Hence, the scaling factors satisfy

$$\prod_{s\geq t}\Xi_s\;\Big|\;\Big((\underline{A}_t,\underline{N}_t^a)=(\underline{a}_{*,t},\underline{n}_{*,t}^a),(\bar{A}_{t-},\bar{N}_{t-}^a)=(\bar{a}_{t-},\bar{n}_{t-}^a)\Big) \tag{131}$$

$$=\prod_{s\geq t}\frac{\mathrm{d}G_{0,s}(\bar{A}_s,\bar{N}_s^a)}{\mathrm{d}G_{*,s}(\bar{A}_s,\bar{N}_s^a)}\;\Big|\;\Big((\underline{A}_t,\underline{N}_t^a)=(\underline{a}_{*,t},\underline{n}_{*,t}^a),(\bar{A}_{t-},\bar{N}_{t-}^a)=(\bar{a}_{t-},\bar{n}_{t-}^a)\Big) \tag{132}$$

$$=\exp\Big[-\int_{s\geq t}\lambda_0^a(s\mid \bar{A}_{s-},\bar{N}_{s-}^a)\,\mathrm{d}s\Big]$$
$$\times\prod_{j=1}^{J}\lambda_0^a(t_{*,j}^a\mid \bar{A}_{t_{*,j}^a-},\bar{N}_{t_{*,j}^a-}^a)\pi_{0,t_{*,j}^a}(a_{*,t_{*,j}^a}\mid \bar{A}_{t_{*,j}^a-},\bar{N}_{t_{*,j}^a-}^a)\;\Big|\;\Big((\underline{A}_t,\underline{N}_t^a)=(\underline{a}_{*,t},\underline{n}_{*,t}^a),(\bar{A}_{t-},\bar{N}_{t-}^a)=(\bar{a}_{t-},\bar{n}_{t-}^a)\Big) \tag{133}$$

$$=\prod_{j=1}^{J}\frac{\lambda_0^a(t_{*,j}^a\mid \bar{A}_{t_{*,j}^a-},\bar{N}_{t_{*,j}^a-}^a)\pi_{0,t_{*,j}^a}(a_{*,t_{*,j}^a}\mid \bar{A}_{t_{*,j}^a-},\bar{N}_{t_{*,j}^a-}^a)}{\exp\int_{s\in[t_{*,j-1}^a,t_{*,j}^a)}\lambda_0^a(s\mid \bar{A}_{s-},\bar{N}_{s-}^a)\,\mathrm{d}s}\;\Big|\;\Big((\underline{A}_t,\underline{N}_t^a)=(\underline{a}_{*,t},\underline{n}_{*,t}^a),(\bar{A}_{t-},\bar{N}_{t-}^a)=(\bar{a}_{t-},\bar{n}_{t-}^a)\Big) \tag{134}$$

$$=\prod_{j=1}^{J}\Xi_{t_{*,j}^a}\;\Big|\;\Big((\underline{A}_t,\underline{N}_t^a)=(\underline{a}_{*,t},\underline{n}_{*,t}^a),(\bar{A}_{t-},\bar{N}_{t-}^a)=(\bar{a}_{t-},\bar{n}_{t-}^a)\Big). \tag{135}$$

□

E DATA GENERATION

E.1 TUMOR GROWTH DATA

The tumor data used in Sec. 5 was simulated based on the lung cancer model proposed by Geng et al. (2017), which has been previously used in several works (Lim et al., 2018; Bica et al., 2020; Li et al., 2021; Melnychuk et al., 2022; Seedat et al., 2022; Vanderschueren et al., 2023).

Specifically, we adopt the simulation framework introduced by Vanderschueren et al. (2023), which includes irregularly spaced observations. However, different to their work, we explicitly add confounding bias to the treatment assignment (that is, both treatment times and treatment choice).

The tumor volume is the outcome variable. It evolves over time according to the ordinary differential equation

$$\mathrm{d}Y_t = \left(1 + \underbrace{\rho \log\left(\frac{K}{Y_t}\right)}_{\text{Tumor growth}} - \underbrace{\alpha_c c_t}_{\text{Chemotherapy}} - \underbrace{(\alpha_r d_t + \beta_r d_t^2)}_{\text{Radiotherapy}} + \underbrace{\epsilon_t}_{\text{Noise}}\right) Y_t \,\mathrm{d}t, \tag{136}$$

where ρ is the tumor growth rate, K represents the carrying capacity, and α_c, α_r, and β_r control the effects of chemotherapy and radiotherapy, respectively. The term ϵ_t introduces randomness into the dynamics. The parameters were drawn following the distributions as in Geng et al. (2017), with details provided in Table 3. The variables c_t and d_t represent chemotherapy and radiotherapy treatments, respectively, and follow previous works (Lim et al., 2018; Bica et al., 2020; Seedat et al., 2022). Time t is measured in days.

	Variable	Parameter	Distribution	Value (μ, σ^2)
Tumor growth	Growth parameter Carrying capacity	ρ K	Normal Constant	$(7.00 \times 10^{-5}, 7.23 \times 10^{-3})$ 30
Radiotherapy	Radio cell kill Radio cell kill	α_r β_r	Normal –	$(0.0398, 0.168)$ Set to $\beta_r = 10 \times \alpha_r$
Chemotherapy	Chemo cell kill	α_c	Normal	$(0.028, 7.00 \times 10^{-4})$
Noise	–	ϵ_t	Normal	$(0, 0.01^2)$

Table 3: Parameter details for the synthetic data generating process.

The radiation dosage d_t and chemotherapy drug concentration c_t are applied with probabilities The treatments are administered according to the following, history dependent treatment probabilities

$$A_t^c, A_t^r \sim \text{Ber}\left(\sigma\left(\frac{\gamma}{D_{\text{max}}}(\bar{D}_{15}(\bar{Y}_{t-1} - \bar{D}_{\text{max}}/2\right)\right), \tag{137}$$

where γ controls the **confounding strength**, D_{max} is the maximum tumor volume, $\bar{D}_{15}$ the average tumor diameter of the last 15 time steps, and γ controls the confounding strength. For test data, we want to evaluate the potential outcomes under hard interventions. Hence, uniformly sample a random treatment sequence and apply it to the outcome, irrespective of the history, as is done in (Melnychuk et al., 2022).

Importantly, this treatment assignment process is *only used for training and validation*. For testing, we randomly sample *hard interventions* as described in Sec. 3. Thereby, we can directly investigate how all baselines perform under time-varying confounding.

We add an observation process that randomly masks away observations of the outcome variable. Hence, at some days, the tumor diameter remains *unobserved*. Irregular observations times are the domain that continuous time methods are tailored for.

For this, our setup is consistent with Vanderschueren et al. (2023), who define the observation process as a Hawkes process with intensity $\lambda_0^y(t)$ given by

$$\lambda_0^y(t) = \text{sigmoid}\left[\omega\left(\frac{\bar{D}_t}{D_{\text{ref}}} - \frac{1}{2}\right)\right], \tag{138}$$

where ω determines the informativeness of the sampling, $D_{\text{ref}} = 13$ cm represents the reference tumor diameter, and $\bar{D}_t$ is the average tumor diameter over the past 15 days. In this work, however, our main focus is **not** informative sampling, which is an orthogonal research direction. Therefore, we opted for setting the informativeness parameter to $\omega = 0$. Thereby, we are in the setting that is known as *sampling completely a random*.

Following Kidger et al. (2020), we added a multivariate counting variable that counts the number of observations up to each day, respectively. We normalized this counting variable with the maximum time scale τ.

Finally, consistent with (Lim et al., 2018; Bica et al., 2020; Seedat et al., 2022; Vanderschueren et al., 2023), we introduced patient heterogeneity by modeling distinct subgroups. Each subgroup differs in their average treatment response, characterized by the mean of the normal distributions. Specifically, for subgroup A, we increased the mean of α_r by 10%, and for subgroup B, we increased the mean of α_c by 10%.

The observed time window for training, validation, and testing is set to $\tau = 30$ days. We generate 1000 observations for training, validation, and testing, respectively.

E.2 MIMIC-III DATA

For our semi-synthetic experiments in Sec. 5, we upon the MIMIC-extract dataset (Wang et al., 2020), which is based on the MIMIC-III database (Johnson et al., 2016) and widely used in research (e.g., Özyurt et al., 2021). Importantly, measurements in this dataset have irregular timestamps for different covariates. Therefore, we can directly use this missingness without artificially introduces any masking process for covariates.

In our setup, we use 9 time-varying covariates (i.e., vital signs) alongside the static covariates gender, ethnicity, and age. As we are interested in conditional average potential outcomes, we need to introduce a synthetic data outcome generation process. For this, we simulate a two-dimensional outcome variable for training and validation purposes and generate interventional outcomes for testing. As defined in Sec. 3, we add past observed outcomes to the list of covariates. Hence, have have a $d_x = 14$-dimensional covariate space. The outcome-generation process follows (Melnychuk et al., 2022):

Simulating untreated outcomes: We first simulate two untreated outcomes $\tilde{Y}_t^j$ for $j = 1, 2$ as follows:

$$\tilde{Y}_t^j = \alpha_s^j \text{B-spline}(t) + \alpha_g^j g^j(t) + \alpha_f^j f_Y^j(X_t) + \epsilon_t, \tag{139}$$

where α_s^j, α_g^j, and α_f^j are weight parameters. Here, B-spline(t) is drawn from a mixture of three cubic splines, and $f_Y^j(\cdot)$ is a random function approximated using random Fourier features from a Gaussian process.

Simulating treatment assignments: We simulate $d_a = 3$ synthetic treatments A_t^l for $l = 1, 2, 3$ according to:

$$A_t^l \sim \text{Ber}(p_t^l), \quad p_t^l = \sigma\left(\gamma_Y^l Y_{t-1}^{A,l} + \gamma_X^l f_Y^l(X_t) + b^l\right), \tag{140}$$

where γ_Y^l and γ_X^l are parameters that control the influence of past treatments and covariates on treatment assignment. $Y_t^{A,l}$ represents a summary of previously treated outcomes, b^l is a bias term, and $f_Y^l(\cdot)$ is another random function sampled using a random Fourier features approximation of a Gaussian process. For test data, we want to evaluate the potential outcomes under hard interventions. Hence, uniformly sample a random treatment sequence and apply it to the outcome, irrespective of the history, as is done in (Melnychuk et al., 2022).

Applying treatments to outcomes: Finally, for training and validation, treatments are applied to the untreated outcomes $\tilde{Y}t^j$ using:

$$Y_t^j = \tilde{Y}t^j + \sum_{i=t-\omega^l}^{t} \frac{\min_{l=1,\ldots,d_a} \mathbb{1}_{A_i^l=1} p_i^l \beta^{l,j}}{(\omega^l - i)^2}, \tag{141}$$

where ω^l defines the duration of the treatment effect window, and $\beta^{l,j}$ determines the maximum effect of treatment A^l on outcome Y_t^j. Importantly, we do not follow this treatment assignment mechanism for testing. Instead, as we are interested in estimating conditional average potential outcomes, we randomly assign *hard interventions* as in Sec. 3.

Masking the outcome: Finally, we add an observation mask to the outcome variable Y_t^j. For this, we randomly mask away the outcome variable with observation probability $p = 0.15$

In our experiments in Sec. 5, we used 1000 samples for training, validation and testing, respectively. For testing, we simulate 50 different intervention sequences per patient. The time window was set between $30 \leq T \leq 50$.

F ADDITIONAL RESULTS

F.1 VARIATION OF OBSERVATION INTENSITIES

In the following, we evaluate the stability of our SCIP-Net for different sampling intensities. For this, we use the tumor growth model as in Section 5. In our main study, observation times follow a history-dependent intensity process as informed by prior literature (Vanderschueren et al., 2023). Here, the observation probabilities $\lambda_0^y(t)$ are given by

$$\lambda_0^y(t) = \text{sigmoid}\left[\omega\left(\frac{\bar{D}_t}{D_{\text{ref}}} - \frac{1}{2}\right)\right], \tag{142}$$

where ω determines the informativeness of the sampling, $D_{\text{ref}} = 13$ cm represents the reference tumor diameter, and $\bar{D}_t$ is the average tumor diameter over the past 15 days. More details are provided in Supplement E.1.

In our main study in Section 5, we focused on the sampling completely at random setting, where we set the informativeness parameter $\omega = 0$. In the following, we increase the informativeness parameter up to $\omega = 0.5$. Tables 4 and 5 show the performance of our SCIP-Net against the baselines. We find that our SCIP-Net has very robust performance and consistently outperforms the baselines.

Informativeness of observation times	Prediction window in days	G-Net (Li et al., 2021)	CT (Melnychuk et al., 2022)	RMSNs (Lim et al., 2018)	CRN (Bica et al., 2020)	TE-CDE (Seedat et al., 2022)	SCIP-Net (ours)
$\omega = 0.0$	1	5.69 ± 1.09	6.45 ± 0.95	7.01 ± 1.29	8.31 ± 1.23	7.86 ± 0.80	$\mathbf{4.33 \pm 0.89}$
	2	5.10 ± 1.64	7.30 ± 2.18	5.87 ± 1.45	7.32 ± 1.59	7.21 ± 0.88	$\mathbf{4.24 \pm 0.44}$
	3	4.51 ± 1.20	6.19 ± 1.72	4.86 ± 1.36	5.97 ± 1.21	6.10 ± 0.77	$\mathbf{4.15 \pm 1.20}$
$\omega = 0.1$	1	6.65 ± 0.86	6.64 ± 0.82	7.88 ± 1.12	8.15 ± 0.68	7.95 ± 1.90	$\mathbf{5.63 \pm 2.15}$
	2	6.19 ± 1.44	6.09 ± 1.43	6.37 ± 1.19	7.73 ± 1.66	7.69 ± 1.56	$\mathbf{4.57 \pm 0.82}$
	3	5.26 ± 1.20	5.24 ± 1.09	5.16 ± 0.97	6.31 ± 1.29	6.57 ± 1.29	$\mathbf{4.18 \pm 0.53}$
$\omega = 0.2$	1	6.57 ± 0.87	6.33 ± 0.51	7.21 ± 1.16	8.56 ± 1.70	7.83 ± 0.49	$\mathbf{5.47 \pm 1.72}$
	2	6.14 ± 1.49	5.86 ± 1.26	6.67 ± 1.41	7.91 ± 1.37	7.65 ± 1.56	$\mathbf{4.69 \pm 0.41}$
	3	5.22 ± 1.23	5.05 ± 0.99	5.43 ± 1.11	6.45 ± 1.11	6.48 ± 1.24	$\mathbf{4.30 \pm 0.42}$
$\omega = 0.3$	1	6.40 ± 1.05	6.26 ± 0.73	7.01 ± 1.37	7.54 ± 0.59	8.65 ± 1.31	$\mathbf{5.48 \pm 2.12}$
	2	6.05 ± 1.64	5.71 ± 1.31	6.69 ± 1.72	7.76 ± 1.29	7.61 ± 1.66	$\mathbf{4.74 \pm 0.27}$
	3	5.13 ± 1.33	4.91 ± 0.99	5.39 ± 1.34	6.32 ± 1.07	6.49 ± 1.44	$\mathbf{4.25 \pm 0.45}$
$\omega = 0.4$	1	6.35 ± 0.39	6.27 ± 0.81	6.86 ± 0.77	8.51 ± 0.75	7.66 ± 0.98	$\mathbf{5.97 \pm 1.76}$
	2	6.02 ± 1.20	5.79 ± 1.37	6.57 ± 1.53	8.12 ± 1.66	7.48 ± 1.92	$\mathbf{4.73 \pm 0.63}$
	3	5.12 ± 0.97	4.98 ± 1.04	5.38 ± 1.22	6.51 ± 1.31	6.34 ± 1.62	$\mathbf{4.28 \pm 0.66}$
$\omega = 0.5$	1	$\mathbf{5.97 \pm 0.39}$	5.98 ± 0.83	7.07 ± 0.45	9.29 ± 1.03	8.56 ± 1.24	6.39 ± 1.52
	2	5.64 ± 1.10	5.51 ± 1.37	6.57 ± 1.34	7.96 ± 1.61	7.68 ± 1.98	$\mathbf{4.77 \pm 0.89}$
	3	4.82 ± 0.91	4.81 ± 1.07	5.38 ± 1.09	6.50 ± 1.29	6.46 ± 1.46	$\mathbf{4.29 \pm 0.69}$

Table 4: **Informative sampling:** Performance for the tumor growth model with irregular sampling times and confounding strength $\boldsymbol{\gamma = 8}$. We vary the informative sampling parameter ω. Our SCIP-Net has robust performance and consistently outperforms the baselines.

Informativeness of observation times	Prediction window in days	G-Net (Li et al., 2021)	CT (Melnychuk et al., 2022)	RMSNs (Lim et al., 2018)	CRN (Bica et al., 2020)	TE-CDE (Seedat et al., 2022)	SCIP-Net (ours)
$\omega = 0.0$	1	8.74 ± 0.49	8.85 ± 1.39	9.52 ± 0.98	12.15 ± 0.71	10.02 ± 1.22	$\mathbf{5.13 \pm 0.44}$
	2	6.48 ± 1.23	8.41 ± 0.75	6.98 ± 1.16	8.35 ± 0.91	8.09 ± 0.99	$\mathbf{4.91 \pm 0.31}$
	3	5.39 ± 0.98	7.05 ± 0.58	5.61 ± 0.93	6.74 ± 0.80	6.73 ± 0.95	$\mathbf{4.47 \pm 0.39}$
$\omega = 0.1$	1	9.16 ± 0.32	8.71 ± 0.73	10.08 ± 0.60	11.11 ± 1.06	11.15 ± 0.65	$\mathbf{4.99 \pm 0.74}$
	2	7.45 ± 1.08	6.77 ± 0.64	7.35 ± 1.04	8.91 ± 1.11	8.42 ± 1.25	$\mathbf{4.84 \pm 0.81}$
	3	6.17 ± 0.95	5.75 ± 0.71	5.91 ± 0.85	7.14 ± 0.89	6.95 ± 1.09	$\mathbf{4.53 \pm 0.90}$
$\omega = 0.2$	1	8.99 ± 0.32	8.84 ± 0.99	9.34 ± 0.83	11.30 ± 0.98	10.92 ± 0.11	$\mathbf{5.37 \pm 1.47}$
	2	7.32 ± 1.11	6.99 ± 0.32	7.69 ± 1.03	8.99 ± 1.02	8.53 ± 0.95	$\mathbf{5.03 \pm 0.50}$
	3	6.08 ± 0.98	5.89 ± 0.45	6.18 ± 0.87	7.20 ± 0.85	6.99 ± 0.84	$\mathbf{4.59 \pm 0.47}$
$\omega = 0.3$	1	8.81 ± 0.43	8.65 ± 0.99	9.42 ± 0.72	11.28 ± 0.90	10.91 ± 0.71	$\mathbf{6.33 \pm 2.31}$
	2	7.25 ± 1.14	6.84 ± 0.41	7.89 ± 1.05	9.00 ± 0.99	8.74 ± 0.83	$\mathbf{6.04 \pm 2.14}$
	3	6.02 ± 0.99	$\mathbf{5.80 \pm 0.57}$	6.33 ± 0.92	7.19 ± 0.79	7.23 ± 0.74	5.98 ± 2.10
$\omega = 0.4$	1	8.59 ± 0.32	8.42 ± 1.60	8.74 ± 1.02	11.46 ± 0.39	11.30 ± 1.04	$\mathbf{4.50 \pm 0.38}$
	2	7.08 ± 1.03	6.63 ± 0.62	7.08 ± 0.69	8.84 ± 0.97	8.47 ± 1.19	$\mathbf{4.79 \pm 0.71}$
	3	5.86 ± 0.88	5.65 ± 0.42	5.73 ± 0.64	7.10 ± 0.78	6.91 ± 0.82	$\mathbf{4.62 \pm 0.88}$
$\omega = 0.5$	1	8.11 ± 0.42	8.19 ± 1.43	8.65 ± 1.25	11.96 ± 0.37	11.38 ± 0.64	$\mathbf{4.80 \pm 0.72}$
	2	6.73 ± 1.11	6.51 ± 0.77	7.29 ± 0.49	9.02 ± 1.00	8.60 ± 1.10	$\mathbf{4.94 \pm 0.59}$
	3	5.60 ± 0.95	5.57 ± 0.55	5.89 ± 0.51	7.22 ± 0.83	7.07 ± 0.92	$\mathbf{4.72 \pm 0.73}$

Table 5: **Informative sampling:** Performance for the tumor growth model with irregular sampling times and confounding strength $\boldsymbol{\gamma = 6}$. As in Table 4, we vary the informative sampling parameter ω. Our SCIP-Net again demonstrates superior performance and outperforms all baselines.

F.2 ABLATION STUDY: SDIP-NET FOR DISCRETE TIME

Confounding strength	Prediction window in days	**G-Net** (Li et al., 2021)	**CT** (Melnychuk et al., 2022)	**RMSNs** (Lim et al., 2018)	**CRN** (Bica et al., 2020)	**SDIP-Net** (ours)
$\gamma = 4$	1	1.86 ± 0.15	2.44 ± 0.24	1.73 ± 0.17	2.44 ± 0.22	$\mathbf{1.53 \pm 0.12}$
	2	2.17 ± 0.51	$\mathbf{2.11 \pm 0.52}$	2.62 ± 0.65	2.78 ± 0.68	3.40 ± 0.80
	3	2.22 ± 0.49	$\mathbf{1.97 \pm 0.47}$	2.32 ± 0.57	2.55 ± 0.64	2.74 ± 0.68
$\gamma = 5$	1	2.27 ± 0.35	2.88 ± 0.31	2.10 ± 0.26	3.37 ± 0.54	$\mathbf{1.86 \pm 0.18}$
	2	$\mathbf{2.90 \pm 0.39}$	2.93 ± 0.44	3.74 ± 0.41	4.03 ± 0.33	4.78 ± 0.36
	3	2.91 ± 0.37	$\mathbf{2.79 \pm 0.47}$	3.30 ± 0.36	3.68 ± 0.35	3.87 ± 0.32
$\gamma = 6$	1	2.59 ± 0.40	3.37 ± 0.46	2.60 ± 0.42	4.14 ± 0.62	$\mathbf{1.93 \pm 0.28}$
	2	$\mathbf{3.31 \pm 0.41}$	3.36 ± 0.58	4.43 ± 0.66	4.97 ± 0.89	5.58 ± 1.14
	3	3.31 ± 0.41	$\mathbf{3.15 \pm 0.58}$	3.87 ± 0.59	4.39 ± 0.76	4.48 ± 0.84
$\gamma = 7$	1	3.06 ± 0.63	4.13 ± 0.43	3.14 ± 0.52	5.04 ± 0.58	$\mathbf{2.23 \pm 0.49}$
	2	$\mathbf{3.61 \pm 0.54}$	3.89 ± 0.91	5.11 ± 1.07	5.48 ± 1.18	5.96 ± 1.39
	3	$\mathbf{3.51 \pm 0.57}$	3.75 ± 1.17	4.31 ± 0.93	4.79 ± 1.08	4.70 ± 1.06
$\gamma = 8$	1	3.37 ± 0.77	4.39 ± 0.62	3.61 ± 0.65	5.80 ± 0.87	$\mathbf{2.42 \pm 0.53}$
	2	$\mathbf{3.74 \pm 0.58}$	3.86 ± 0.63	5.56 ± 0.77	5.95 ± 0.99	6.14 ± 0.94
	3	$\mathbf{3.68 \pm 0.46}$	3.79 ± 0.71	4.64 ± 0.61	5.15 ± 0.72	4.92 ± 0.78
$\gamma = 9$	1	3.52 ± 0.71	4.72 ± 0.57	3.88 ± 0.69	6.34 ± 0.81	$\mathbf{2.58 \pm 0.64}$
	2	$\mathbf{4.41 \pm 1.13}$	4.78 ± 1.16	7.00 ± 1.50	6.87 ± 1.46	7.17 ± 1.47
	3	$\mathbf{4.28 \pm 1.07}$	4.80 ± 1.48	5.70 ± 1.22	5.90 ± 1.30	5.72 ± 1.21
$\gamma = 10$	1	3.81 ± 0.86	5.23 ± 0.58	4.23 ± 1.00	6.61 ± 1.02	$\mathbf{2.74 \pm 0.67}$
	2	$\mathbf{4.72 \pm 0.53}$	5.07 ± 1.09	8.32 ± 1.22	7.47 ± 1.10	7.52 ± 1.16
	3	$\mathbf{3.54 \pm 0.42}$	4.69 ± 0.86	6.35 ± 0.89	6.25 ± 0.82	5.94 ± 0.79

Table 6: **Ablation study:** Performance for the tumor growth model with regular sampling times. Our SDIP-Net ablation has comparable performance to the state-of-the-art baselines for estimating CAPOs in discrete time.

We perform an ablation study, where we use our stabilized inverse propensity weights in the discrete-time setting. For this, we use the tumor growth data as in Section 5. However, we do not apply an observation mask. Instead, all timestamps are observed.

Here, we use an LSTM (Hochreiter & Schmidhuber, 1997) as the neural backbone in order to demonstrate that our approach is also applicable to other neural backbones. Our *stabilized discrete time inverse propensity network (SDIP-Net)* has comparable performance to the baselines. Table 6 shows the results, which thus confirm the effectiveness of our approach. Nevertheless, we emphasize that our approach is tailored for the continuous time setting, and **not** for the more unrealistic discrete-time setting.

G HYPERPARAMETER TUNING

In order to ensure a fair comparison of all methods, we close follow hyperparameter tuning as in (Melnychuk et al., 2022) and (Hess et al., 2024a). In particular, we performed a random grid search. Below, we report the tuning grid for each method. Importantly, all methods are only tuned on *factual* data. For optimization, we use Adam (Kingma & Ba, 2015). Both TE-CDE (Seedat et al., 2022) and our SCIP-Net used a simple Euler quadrature and linear interpolation for the Neural CDE control path (Morrill et al., 2021). For the neural CDE of both methods and the integrated intensity in our SCIP-Net, we did not tune the grid size of the solver. Further, we emphasize that different quadrature schemes may impact training time. Fortunately, we did not encounter large differences in performance between schemes of different orders and, hence, opted for Euler integration.

Runtime: All methods were trained on $1\times$ NVIDIA A100-PCIE-40GB. On average, training our SCIP-Net took 29.4 minutes on tumor growth data and approximately 1.7 hours on MIMIC-III data, which is comparable to the baselines.

Method	Component	Hyperparameter	Tuning range
CRN (Bica et al., 2020)	Encoder	LSTM layers (J)	1
		Learning rate (η)	0.01, 0.001, 0.0001
		Minibatch size	64, 128, 256
		LSTM hidden units (d_h)	$0.5d_{yxa}$, $1d_{yxa}$, $2d_{yxa}$, $3d_{yxa}$, $4d_{yxa}$
		Balanced representation size (d_z)	$0.5d_{yxa}$, $1d_{yxa}$, $2d_{yxa}$, $3d_{yxa}$, $4d_{yxa}$
		FC hidden units (n_{HC})	$0.5d_z$, $1d_z$, $2d_z$, $3d_z$, $4d_z$
		LSTM dropout rate (p)	0.1, 0.2
		Number of epochs (n_e)	50
	Decoder	LSTM layers (J)	1
		Learning rate (η)	0.01, 0.001, 0.0001
		Minibatch size	256, 512, 1024
		LSTM hidden units (d_h)	Balanced representation size of encoder
		Balanced representation size (d_z)	$0.5d_{yxa}$, $1d_{yxa}$, $2d_{yxa}$, $3d_{yxa}$, $4d_{yxa}$
		FC hidden units (n_{HF})	$0.5d_z$, $1d_z$, $2d_z$, $3d_z$, $4d_z$
		LSTM dropout rate (p)	0.1, 0.2
		Number of epochs (n_e)	50
CT (Melnychuk et al., 2022)	(end-to-end)	Transformer blocks (J)	1,2
		Learning rate (η)	0.01, 0.001, 0.0001
		Minibatch size	64, 128, 256
		Attention heads (n_h)	1
		Transformer units (d_h)	$1d_{yxa}$, $2d_{yxa}$, $3d_{yxa}$, $4d_{yxa}$
		Balanced representation size (d_z)	$0.5d_{yxa}$, $1d_{yxa}$, $2d_{yxa}$, $3d_{yxa}$, $4d_{yxa}$
		Feed-forward hidden units (n_{HF})	$0.5d_z$, $1d_z$, $2d_z$, $3d_z$, $4d_z$
		Sequential dropout rate (p)	0.1, 0.2
		Max positional encoding (l_{max})	15
		Number of epochs (n_e)	50
RMSNs (Lim et al., 2018)	Propensity treatment network	LSTM layers (J)	1
		Learning rate (η)	0.01, 0.001, 0.0001
		Minibatch size	64, 128, 256
		LSTM hidden units (d_h)	$0.5d_{yxa}$, $1d_{yxa}$, $2d_{yxa}$, $3d_{yxa}$, $4d_{yxa}$
		LSTM dropout rate (p)	0.1, 0.2
		Max gradient norm	0.5, 1.0, 2.0
		Number of epochs (n_e)	50
	Propensity history network / Encoder	LSTM layers (J)	1
		Learning rate (η)	0.01, 0.001, 0.0001
		Minibatch size	64, 128, 256
		LSTM hidden units (d_h)	$0.5d_{yxa}$, $1d_{yxa}$, $2d_{yxa}$, $3d_{yxa}$, $4d_{yxa}$
		LSTM dropout rate (p)	0.1, 0.2
		Max gradient norm	0.5, 1.0, 2.0
		Number of epochs (n_e)	50
	Decoder	LSTM layers (J)	1
		Learning rate (η)	0.01, 0.001, 0.0001
		Minibatch size	256, 512, 1024
		LSTM hidden units (d_h)	$1d_{yxa}$, $2d_{yxa}$, $4d_{yxa}$, $8d_{yxa}$, $16d_{yxa}$
		LSTM dropout rate (p)	0.1, 0.2
		Max gradient norm	0.5, 1.0, 2.0, 4.0
		Number of epochs (n_e)	50
G-Net (Li et al., 2021)	(end-to-end)	LSTM layers (J)	1
		Learning rate (η)	0.01, 0.001, 0.0001
		Minibatch size	64, 128, 256
		LSTM hidden units (d_h)	$0.5d_{yxa}$, $1d_{yxa}$, $2d_{yxa}$, $3d_{yxa}$, $4d_{yxa}$
		LSTM output size (d_z)	$0.5d_{yxa}$, $1d_{yxa}$, $2d_{yxa}$, $3d_{yxa}$, $4d_{yxa}$
		Feed-forward hidden units (n_{HF})	$0.5d_z$, $1d_z$, $2d_z$, $3d_z$, $4d_z$
		LSTM dropout rate (p)	0.1, 0.2
		Number of epochs (n_e)	50
TE-CDE (Seedat et al., 2022)	Encoder	Neural CDE (Kidger et al., 2020) hidden layers (J)	1
		Learning rate (η)	0.01, 0.001, 0.0001
		Minibatch size	64, 128, 256
		Neural CDE hidden units (d_h)	$0.5d_{yxa}$, $1d_{yxa}$, $2d_{yxa}$, $3d_{yxa}$, $4d_{yxa}$
		Balanced representation size (d_z)	$0.5d_{yxa}$, $1d_{yxa}$, $2d_{yxa}$, $3d_{yxa}$, $4d_{yxa}$
		Feed-forward hidden units (n_{HF})	$0.5d_z$, $1d_z$, $2d_z$, $3d_z$, $4d_z$
		Neural CDE dropout rate (p)	0.1, 0.2
		Number of epochs (n_e)	50
	Decoder	Neural CDE hidden layers (J)	1
		Learning rate (η)	0.01, 0.001, 0.0001
		Minibatch size	256, 512, 1024
		Neural CDE hidden units (d_h)	Balanced representation size of encoder
		Balanced representation size (d_z)	$0.5d_{yxa}$, $1d_{yxa}$, $2d_{yxa}$, $3d_{yxa}$, $4d_{yxa}$
		Feed-forward hidden units (n_{HF})	$0.5d_z$, $1d_z$, $2d_z$, $3d_z$, $4d_z$
		Neural CDE dropout rate (p)	0.1, 0.2
		Number of epochs (n_e)	50
SCIP-Net (ours)	Weight network	Neural CDE (Kidger et al., 2020) hidden layers (J)	1
		Learning rate (η)	0.01, 0.001, 0.0001
		Minibatch size	64, 128, 256
		Neural CDE hidden units (d_h)	$0.5d_{yxa}$, $1d_{yxa}$, $2d_{yxa}$, $3d_{yxa}$, $4d_{yxa}$
		Neural CDE dropout rate (p)	0.1, 0.2
		Max gradient norm	0.5, 1.0, 2.0
		Number of epochs (n_e)	50
	Treatment network / Encoder	Neural CDE hidden layers (J)	1
		Learning rate (η)	0.01, 0.001, 0.0001
		Minibatch size	64, 128, 256
		Neural CDE hidden units (d_h)	$0.5d_{yxa}$, $1d_{yxa}$, $2d_{yxa}$, $3d_{yxa}$, $4d_{yxa}$
		Neural CDE dropout rate (p)	0.1, 0.2
		Max gradient norm	0.5, 1.0, 2.0
		Number of epochs (n_e)	50
	Decoder	Neural CDE hidden layers (J)	1
		Learning rate (η)	0.01, 0.001, 0.0001
		Minibatch size	256, 512, 1024
		Neural CDE hidden units (d_h)	$1d_{yxa}$, $2d_{yxa}$, $4d_{yxa}$, $8d_{yxa}$, $16d_{yxa}$
		Neural CDE dropout rate (p)	0.1, 0.2
		Max gradient norm	0.5, 1.0, 2.0, 4.0
		Number of epochs (n_e)	50

Table 7: Following (Melnychuk et al., 2022), we let $d_{yxa} = d_y + d_x + d_a$ be the overall input size. Further, d_z is the hidden representation size of our SCIP-Net, and corresponds to the balanced representation size of TE-CDE (Seedat et al., 2022), CRN (Bica et al., 2020), and CT (Melnychuk et al., 2022), and the LSTM output size of G-Net (Li et al., 2021).