Continuous-Time Decision Transformer for Healthcare Applications

Zhiyue Zhang
Johns Hopkins University

Hongyuan Mei
Toyota Technological Institute at Chicago

Yanxun Xu
Johns Hopkins University

Abstract

Offline reinforcement learning (RL) is a promising approach for training intelligent medical agents to learn treatment policies and assist decision making in many healthcare applications, such as scheduling clinical visits and assigning dosages for patients with chronic conditions. In this paper, we investigate the potential usefulness of Decision Transformer (Chen et al., 2021)—a new offline RL paradigm—in medical domains where decision making in continuous time is desired. As Decision Transformer only handles discrete-time (or turn-based) sequential decision making scenarios, we generalize it to Continuous-Time Decision Transformer that not only considers the past clinical measurements and treatments but also the timings of previous visits, and learns to suggest the timings of future visits as well as the treatment plan at each visit. Extensive experiments on synthetic datasets and simulators motivated by real-world medical applications demonstrate that Continuous-Time Decision Transformer is able to outperform competitors and has clinical utility in terms of improving patients' health and prolonging their survival by learning high-performance policies from logged data generated using policies of different levels of quality.

1 INTRODUCTION

Sequential decision making in continuous time based on patients' disease progression is common in many healthcare and medical applications. For example, people living with HIV need to take antiretroviral drugs every day indefinitely to suppress HIV viral loads since there is no cure for HIV infection (D'Souza et al., 2019). However, the long term use of certain antiretroviral drugs could lead to drug resistance, causing HIV treatments to fail (Pennings, 2013). Therefore, physicians need to determine when to switch treatments if needed. Another example is managing clinical care for patients with chronic conditions (e.g., kidney transplantation). Patients after kidney transplantation need to follow up with physicians frequently, therefore optimizing their follow-up schedules and prescribing the right dosage of immunosuppressive drugs (e.g., tacrolimus) can have a significant impact on patients' survival (Israni et al., 2014). To determine the optimal sequential decisions, the gold standard is to conduct randomized clinical trials. However, it is usually impractical because data collection is either expensive due to the requirement of a huge sample size especially when decisions are made in continuous time, or unethical (e.g., assigning patients to obviously inferior or even dangerous treatments). As an alternative, we can utilize previously-collected observational data to estimate sequential actions and make decisions; this is known as offline reinforcement learning (RL).

Related Work Over the last two decades, there has been growing interest in applying offline RL for medical applications, including Sepsis (Komorowski et al., 2018; Raghu et al., 2017; Peng et al., 2018; Fatemi et al., 2021), HIV (Ernst et al., 2006; Parbhoo et al., 2017), Schizophrenia (Shortreed et al., 2011), mechanical ventilation (Prasad et al., 2017), and hypotension (Futoma et al., 2020). We refer the readers to Gottesman et al. (2019) and Yu et al. (2021) for guidelines and a thorough review of RL in health care. There are also extensive studies in statistics using the framework of potential outcomes to optimize sequential treatment assignments from observational data (Orellana et al., 2010; Zhao et al., 2015; Clifton & Laber, 2020; Carpenter et al., 2020). Yet, most of these methods optimize decisions at pre-defined schedules and do not consider the timing of actions. For sequential actions in continuous time, we could discretize time and apply standard offline RL algorithms. However, such discretization will cause information loss and may not be feasible when the time horizon is too long (e.g., kidney transplantation example) or too many decision points are needed (e.g., the decision of weaning off ventilator for ICU patients can be made on a minute scale). Previous work has explored marked temporal point processes (MTPPs) (Aalen et al., 2008) to model discrete events in continuous time in the RL framework (Hua et al., 2021), nevertheless, it relies on strong parametric assumptions and specific model

Proceedings of the 26th International Conference on Artificial Intelligence and Statistics (AISTATS) 2023, Valencia, Spain. PMLR: Volume 206. Copyright 2023 by the author(s).

structures based on clinical knowledge for the kidney transplantation, making it hard to generalize.

Researchers in the deep learning field have also attempted deep RL methods that are more flexible to handle continuous time. Upadhyay et al. (2018) use deep MTPPs to characterize both the agent actions and environment feedbacks. The development of Neural-ODEs (Chen et al., 2018) has also inspired a number of model-based continuous-time RL methods: Du et al. (2020) propose to use Neural-ODEs to learn the dynamics of semi-Markov decision processes (MDPs) and partially observable MDPs in continuous time; Yildiz et al. (2021) utilize Bayesian Neural-ODEs to build dynamics models that further account for epistemic uncertainty. However, these methods rely on online interactions and cannot learn solely from logged data in the offline setting. Recently, Fatemi et al. (2022) study value-based offline RL for healthcare in the semi-MDP framework that extends the state-of-the-art MDP-based offline algorithms (Fujimoto et al., 2019). While this work suitably handles irregular action durations in the offline setting, its sophisticated formulation (using a more convoluted value function definition and semi-MDP value update rules) and its using state-of-the-art algorithms as a subroutine brings algorithm complexity, implementation difficulty, and computational overhead.

In this work, we instead turn our attention to Decision Transformer (DT) (Chen et al., 2021), a new offline RL paradigm that is simple and elegant and has shown promising results extending the sequence model success with transformers (Vaswani et al., 2017) from language models to offline RL. DT bypasses value function estimation and policy gradients in traditional RL altogether. Instead, DT directly predicts actions autoregressively conditioning on desired returns and observed states via sequence modeling with a causally masked transformer architecture. Sequence modeling is pertinent for medical settings: in dynamic treatment regimes treatment decisions are tailored for each individual, as such trajectory modeling is more natural for personalized treatments than transition-based RL methods. Further, DT does not require the Markov property of the underlying dynamics, which is a common and essential assumption for MDP-based RL methods but also an assumption that is not always met in medical settings. Moreover, with sequence modeling DT does not rely on the time homogeneity of transitions, which adds flexibility for medical applications. One important limitation of DT, as recently studied by Brandfonbrener et al. (2022), is that DT relies heavily on the high quality of the behavior policy. Luckily, the assumption that offline data contain trajectories logged by high-quality policies can be met in many medical datasets since doctors are required to act in the best interests of patients. It is worth noting that while the sequence model in the DT framework utilizes the transformer architecture, DT's *training* is in principle agnostic to the model architecture choice and therefore one may use any other deep autoregressive model such as Long Short Term Memory (LSTM) networks (Hochreiter & Schmidhuber, 1997) or Temporal Convolutional Networks (TCNs) (Bai et al., 2018). Concurrent to DT, Trajectory Transformer (Janner et al., 2021) uses a similar sequence modeling concept and the transformer architecture, but also adds model-based planning which further requires discretizing states and actions and predicting future states and returns.

Our Contribution In this paper, we investigate the effectiveness of DT in medical applications, in which the desired return can be an expected mean survival time or a pre-determined biomarker value. Since DT only handles discrete-time scenarios, it does not directly apply to medical applications in which the decisions are made continuously in time. For example, when physicians determine the next clinic visitation time for patients after kidney transplantation, it can be any time ranging from a few days to one year later, depending on the patient's historical clinical measurements and treatments (Hua et al., 2021). To address such challenges, we propose Continuous-Time Decision Transformer (CTDT), a generalization of the Decision Transformer, to handle actions made in continuous time and irregular time intervals between states. In particular, by putting interval times into our new trajectory representation, we are able to learn the interconnections among visit timings, clinical treatments, observed states, and achievable returns; by a careful treatment of the temporal and type embeddings, we allow the hidden states of all the elements in our trajectory representation vary in time in a continuous manner. We highlight some notable features of our proposed CTDT framework:

- Our way of breaking the multi-dimensional control (clinical treatments and interval times) into completely separate tokens to discover their further interdependencies is in line with the autoregressive dynamics model idea presented in Zhang et al. (2021) and Janner et al. (2021). However, unlike those prior works that reach into the model-based RL domain requiring accuracy of the learnt dynamics model, our approach remains model-free.

- Building on the original DT, we enjoy all the benefits of the simplicity of DT. Unlike many other value-based or model-based offline methods (Kidambi et al., 2020; Yu et al., 2020; Fujimoto et al., 2019), we do not require any explicit pessimism design or regularization. Rather, we are able to jointly learn the optimal clinical treatments as well as the treatment timings, purely from re-weighting the behavior policies by the future return distributions (Brandfonbrener et al., 2022).

- Compared to the semi-MDP algorithms (Du et al., 2020; Fatemi et al., 2022), our straight-forward continuous-time treatment brings no significant algorithm complexity, and therefore maintains an ease of implementation.

2 PROBLEM FORMULATION

There are numerous medical applications that involve making decisions continuously in time. In Section 2 and 3, we consider one signature application of care management for patients after kidney transplantation to make our discussion and notations concrete. We note that the proposed framework also readily adapts to other continuous-time medical decision making problems. Kidney transplantation is the most common type of organ transplantation and the primary therapy for patients with end-stage kidney disease (Arshad et al., 2019). To prevent graft rejection, patients are required to have frequent follow-ups at an outpatient center after the transplantation surgery. At each visit, their clinical measurements such as creatinine (an indicator for kidney function) are recorded, then they are administered immunosuppressive drugs (e.g., tacrolimus), to keep their immune systems from attacking and rejecting the new kidney (Kasiske et al., 2010). Based on the patient's clinical measurements and the assigned dosages of immunosuppressive drugs, physicians then determine the patient's next follow-up time, which can be anytime ranging from few days to one year from now. Our goal is to optimize clinical decisions including scheduling a patient's follow-up visitations and prescribing dosages to maximize the patient' health outcome.

2.1 Inference Objective

We aim to improve patients' health conditions by optimizing their clinic visit schedules and treatments at each visit. Technically, each patient has a stream of time-stamped clinic visits $\{(t_i, \mathbf{m}_i, \mathbf{c}_i, s_i)\}_{i=1}^I$: for the i^{th} visit, t_i is its time, $\mathbf{m}_i$ is the list of clinical measurements (e.g., creatinine) collected for that patient, $\mathbf{c}_i$ is the list of treatments (e.g., tacrolimus) the doctor assigns to that patient, and s_i is the future schedule which the doctor plans for that patient. Particularly, s_i is the elapsed time for which the patient has to wait until they come back to the clinic for the next visit: i.e., $t_{i+1} = t_i + s_i$.

Suppose the *instantaneous* health condition of a patient at time t is given by $h(t)$, which possibly depends on the entire treatment history. Then we aim to maximize their *total* health condition $\int_0^\infty h(t)dt$. That is, we aim to learn a policy π from which the right clinical treatments $\mathbf{c}_i$ and future schedule s_i can be chosen at each hospital visit i so as to maximize the expected total health condition which we refer to as the total *return*:

$$\max_\pi \mathbb{E}_\pi \int_0^\infty h(t)dt = \max_\pi \sum_{i=1}^\infty \mathbb{E}_{\mathbf{c}_i, s_i \sim \pi} \int_{t_i}^{t_{i+1}} h(t)dt. \quad (1)$$

It is an *offline* RL problem: medical applications are safety-critical so we can not afford to learn from try-and-trial; instead, we are only given a corpus of historical data to learn from without further interacting with patients.

2.2 Offline RL Via Sequence Modeling

We consider an alternative offline RL paradigm called offline RL via sequence modeling: instead of learning measurement-treatment pair values and deriving a value-guided policy or optimize a policy with policy gradients, we wish to *undeviatingly* map rewards to treatments by discovering the underlying interdependence among rewards, treatments, and measurements. As such, an RL problem in this paradigm gets cast into a supervised learning (more specifically, sequence modeling) problem.

To facilitate easier discussion, we transform our notation system from the previously introduced $\{(t_i, \mathbf{m}_i, \mathbf{c}_i, s_i)\}_{i=1}^I$ to $\{(\mathbf{o}_i, \mathbf{a}_i)\}_{i=1}^I$, where $\mathbf{o}$ is for observables or observations containing $\mathbf{m}$ and t, and $\mathbf{a}$ is for actions containing $\mathbf{c}$ and s. We treat inter-visit rewards $r_i := \int_{t_i}^{t_{i+1}} h(t)dt$, observations, and actions as a simple stream of data: $(r_1, \mathbf{o}_1, \mathbf{a}_1, \ldots, r_i, \mathbf{o}_i, \mathbf{a}_i, \ldots)$, and build a sequence model parameterized by $\boldsymbol{\theta}$ to model such data trajectories; by looking at and conditioning on the rewards r_i, the current observation $\mathbf{o}_i$, and all past history $\mathcal{H}(i-1) := \{(r_j, \mathbf{o}_j, \mathbf{a}_j)\}_{j=1}^{i-1}$, appropriate treatment decisions can be obtained autoregressively with the sequence model. As such, the training objective naturally becomes a density maximization problem:

$$\max_{\boldsymbol{\theta}} \prod_{i=1}^I f_{\boldsymbol{\theta}}(\mathbf{a}_i | r_i, \mathbf{o}_i, \mathcal{H}(i-1)). \quad (2)$$

In the case of discrete actions, the densities in the objective Equation (2) simply gets replaced by probability masses, making this RL paradigm easily transfer between discrete and continuous action regimes. The most notable example of RL via sequence modeling algorithm is Decision Transformer proposed by Chen et al. (2021), where they use the powerful transformer architecture (Vaswani et al., 2017) and utilize its large memory capacity to model the reward-observation-action data trajectory.

2.3 Time Distortion

The shortcoming of directly applying the vanilla Decision Transformer to the continuous-time medical settings that we consider now becomes apparent: in contrast with classical RL tasks such as Mujoco or Atari games where actions happen in a turn-based fashion, i.e., at discrete and evenly-spaced times, in the continuous-time medical settings actions take place at irregularly spaced times, and during those time intervals observations evolve continuously in time. Therefore, by directly treating the timestamps as just another dimension of the observation $\mathbf{o}$ (i.e., concatenating time with $\mathbf{m}_i$ to form $\mathbf{o}_i$), the sequence model only gets to look at the discrete-time decision process *embedded in* the underlying true continuous-time decision process, cutting the direct connection between states/actions and time and turning the continuously-evolving process back into a turn-based process. While this modeling approach may get away when time intervals are roughly similar in size

(making the discrete-time approximation accurate), it can be severely weakened when the differences in time intervals are significant since the irregular time distortion is being ignored. Another approach is to discretize continuous-time into discrete time buckets, but this approach is also not feasible when the task horizon is long and loss of information due to low temporal resolution becomes severe. Thus, a more proper way to handle the time distortion is to establish the interaction between actions/observations and time by allowing the actions and observations to change continuously in time; in terms of the transformer structure, this entails allowing the action and state embeddings and the attentions paid to them evolve continuously in time. In essence, we aim to change the modeling philosophy from "observation $\mathbf{o}_i$ is medical measurement $\mathbf{m}_i$ *and* time t_i" to "observation $\mathbf{o}_i$ is medical measurement $\mathbf{m}(t)$ @ time t_i".

3 METHODOLOGY

In this section, we generalize Decision Transformer into a Continuous-Time Decision Transformer (CTDT), which elegantly handles temporal information and flexibly captures the continuous-time process. As a result, our CTDT is a natural fit for learning clinical policies from the offline electronic health record (EHR) data. In the following subsections, we will first sketch how a continuous-time Transformer learns a return-conditioned policy from offline data (Section 3.1), then elaborate the details of the model architecture (Section 3.2), and finally give the training objective and some key implementation choices (Section 3.3).

3.1 CTDT For Medical Sequences

We begin introducing our method by recalling the continuous-time offline medical RL objective Equation (1): at each clinic visitation i, the goal is to find a policy π that maximizes the subsequent tail return/ cumulative health condition, which following standard text for discrete-time RL we also refer to as the return-to-go $\hat{R}_i^\pi$:

$$\sum_{j=i}^{\infty} \mathop{\mathbb{E}}_{\mathbf{c}_i, s_i \sim \pi} \int_{t_j}^{t_{j+1}} h(t)dt =: \sum_{j=i}^{\infty} \mathop{\mathbb{E}}_{\mathbf{c}_i, s_i \sim \pi} r_j =: \hat{R}_i^\pi.$$

In the discrete-time setting with even time spaces, rather than directly seeking a policy $\arg\max_\pi \hat{R}_i^\pi$, the original decision transformer aims to discover the the interconnections among the return-to-go's $\hat{R}_i$, observations $\mathbf{m}_i$, and actions $\mathbf{c}_i$ by modeling them as a sequence of data with trajectory representation $\tau = \left(\hat{R}_1, \mathbf{m}_1, \mathbf{c}_1, \ldots, \hat{R}_I, \mathbf{m}_I, \mathbf{c}_I\right)$. Conditioning on an initial desired return $\hat{R}_{\text{input}}$, the induced policy at each stage i is the output $\hat{\mathbf{c}}_i$ obtained by autoregressing on the trajectory data starting from $\left(\hat{R}_{\text{input}}, \mathbf{m}_1\right)$.

To extend the DT paradigm to the continuous-time setting, we need to address two main questions: 1) how would we undertake the learning task for the interval times s_i such that the interval times $\hat{s}_i$ output by CTDT indeed lead to achieving the desired $\hat{R}_{\text{input}}$? 2) since interval times are irregular and the position indices no longer accurately reflect the temporal distances among elements in the data trajectory, how should we adjust the attention mechanism of DT such that the attention of an element paid to previous elements can vary in accordance with the actual temporal distances?

To tackle the first question, we propose to model the interval times s_i as not only another dimension of the clinical decision, but also give them their separate tokens so that the complex interdependency among clinical measurements, treatments, cumulative health conditions and interval times can be better captured by the sequence model. In the signature kidney transplantation survival problem, doctors typically schedule the next visit based on the clinical measurements and dosages from the current visit, therefore, we choose to model s_i having a dependence on the current treatment decision $\mathbf{c}_i$. To this end, it is natural that we consider a trajectory representation composed of (return-to-go, measurement, decision, interval time) tuples of each visit:

$$\tau_{\text{CTDT}} = \left(\hat{R}_{t_1}, \mathbf{m}_{t_1}, \mathbf{c}_{t_1}, s_{t_1}, \ldots, \hat{R}_{t_I}, \mathbf{m}_{t_I}, \mathbf{c}_{t_I}, s_{t_I}\right). \quad (3)$$

In accordance with the above proposed trajectory representation, we factorize the training objective Equation (2) as $\prod_{i=1}^{I} f_{\boldsymbol{\theta}}(\boldsymbol{c}_i | r_i, \mathbf{o}_i, \mathcal{H}(i-1)) \times f_{\boldsymbol{\theta}}(s_i | \mathbf{c}_i, r_i, \mathbf{o}_i, \mathcal{H}(i-1))$. Note that such trajectory representation is by choice and specific to our application; the dependence of treatment decisions on interval times can be easily modeled by simply swapping the order of $\mathbf{c}_i$ and s_i if desired.

Now, to address the second question, we make necessary and intuitive modifications to the decision transformer architecture, which we present with details in the next subsection.

3.2 CTDT Model Architecture

To account for the irregular temporal distances between clinic visits, we adopt a temporal position embedding for the decision transformer that is able to vary continuously in time. Denoting each element in our trajectory τ_{CTDT} Equation (3) of total length $4I$ (4 elements at each visit i) having value y (could be scalar, e.g., dosage amount; or vector valued, e.g., multidimensional health states) at time t and type $e \in \{\hat{R}, \mathbf{m}, \mathbf{c}, s\}$ by $y_e(t)$. For each $y_{e_j}(t_j)$ with $j \in \{1, ..., 4I\}$, we tokenize by separately embedding its temporal information $\boldsymbol{t}_j$, value $\boldsymbol{y}_j$, and type $\boldsymbol{e}_j$. Following Zuo et al. (2020) and Yang et al. (2022), we use the sinusoidal temporal embedding $\boldsymbol{t}$ such that t_k is

$$\sin\left(\frac{t}{C^{\frac{k}{d_{\text{time}}}}}\right) \text{ for even } k \text{ or } \cos\left(\frac{t}{C^{\frac{k-1}{d_{\text{time}}}}}\right) \text{ for odd } k$$

where $C = 10000$ and $k \in \{1, \ldots, d_{\text{time}}\}$ is the kth temporal embedding dimension. We obtain both the initial value embeddings $\boldsymbol{y}_j$ and type embeddings $\boldsymbol{e}_j$ by linear transformations. Rather than adding up, we concatenate the temporal embeddings, type embeddings, and value embeddings to form the base layer input tokens

$\text{Emb}_j^{(0)} := [\boldsymbol{y}_j; \boldsymbol{t}_j; \boldsymbol{e}_j]$, allowing more direct access to the temporal information. Subsequently, at each attention layer l, the key, query, and value are obtained by $\boldsymbol{\omega}_j^{(l)} = \Omega^{(l)}(\text{Emb}_j^{(l-1)})$, with $\boldsymbol{\omega} \in \{\boldsymbol{k}, \boldsymbol{q}, \boldsymbol{v}\}$ and their corresponding linear transformations $\Omega \in \{K, Q, V\}$. We apply a causal mask to our transformer such that each element $y_{e_j}(t_j) \in \tau_{\text{CTDT}}$ with order index j only attends to elements that precede it in order and itself $\{y_{e_n}(t_n)\}_{n=1}^{j}$, with (unnormalized) attention weight paid to element $y_{e_n}(t_n)$ given by $\alpha(y_{e_j}(t_j), y_{e_n}(t_n))^{(l)} := \boldsymbol{q}_j^{(l)} \cdot \boldsymbol{k}_n^{(l)}$. Finally, to get the layer-level embeddings $\text{Emb}_j^{(l)}$, we compute $\sum_{n=1}^{j} \texttt{softmax}\big(\{\alpha(y_{e_j}(t_j), y_{e_{n'}}(t_{n'}))^{(l)}\}_{n'=1}^{j}\big)_n \cdot \boldsymbol{v}_n^{(l)}$ followed by layer normalization (Ba et al., 2016), feed-forward connection, and residual connection.

3.3 Training Objective And Implementation Choices

At the training stage, we compute each $\hat{R}_i = \sum_{k=i}^{I} r_k$ and $s_i = t_{i+1} - t_i$ from the offline data and prepare trajectories in the form of Equation (3) to feed into our continuous-time transformer. In this work, we choose to deterministically predict both the treatments and interval times, and as such, we directly output point estimates: following the last layer of the transformer, we apply additional layers of fully connected networks to the last layer embeddings of $\mathbf{c}_i$ and s_i to either a) directly output both the treatments $\hat{\mathbf{c}}_i$ and interval times $\hat{s}_i$ when treatments are continuous (e.g., drug dosage); or b) output the treatment weights and apply `argmax` for discrete treatments (e.g., drug combinations). Consequently, the likelihood training objective also gets replaced by deterministic supervised learning losses: we train our continuous-time RL sequence model with mean squared error $L_{\text{train}} = \frac{1}{I}\sum_{i=1}^{I}(\hat{\mathbf{c}}_i - \mathbf{c}_i)^2$ for continuous treatments and cross-entropy loss $L_{\text{train}} = \sum_{i=1}^{I}\sum_{k=1}^{\text{treatment }\#} -\log \mathbb{P}(\hat{c}_i = k)\mathbf{1}_{c_i=k}$ for discrete treatments, and in both cases the continuous interval times are trained with mean squared error. Rather than predicting deterministically, the treatment decisions and interval times can also be sampled stochastically by e.g. a marked temporal point process, and we would instead learn the mark distribution and temporal intensity. We leave the exploration in that direction to future work.

At the evaluation step, the initial conditioning return-to-go $\hat{R}_1$ is a user-specified desired return. Similar to Chen et al. (2021), we find the largest return in the offline dataset to be a reasonably good starting place. Each subsequent conditioning return is obtained by $\hat{R}_i = \hat{R}_{i-1} - r_{i-1}$, the previous conditioning return subtracting the actual obtained reward at each visit. The treatment is obtained autoregressively conditioning on all the trajectory history, together with the conditioning return-to-go and the measurement at the current visit, and the interval time is further conditioned on the current assigned treatment.

We base our transformer implementation on the GPT model (Radford et al., 2018), and refer the readers to the original paper for more details on the GPT architecture.

4 EXPERIMENTAL RESULTS

We evaluate the proposed CTDT on three datasets, including one simple synthetic dataset and two medical datasets on kidney transplantation and HIV, respectively. On each dataset, we train our method and evaluate it by an in-domain simulator; we hope the trained agent could help improve the simulated patients' health outcomes and prolong their survival. For online evaluation, we use the same custom simulator used for data generation for our synthetic data study; for the kidney transplantation dataset, we implement a custom python simulator inspired by the stochastic longitudinal model proposed in Hua et al. (2021); for the HIV dataset, we use the simulator provided by Du et al. (2020). The latter two simulators are meaningful since their structures were designed by medical domain experts and their parameters were learned from real-world data. The in-depth description of the three envrionments is given in Appendix A, and the model hyperparameters and training details of all algorithms used in the experiments are given in Appendix B.

4.1 Synthetic Data Study

We design a simple survival experiment that serves a two-fold purpose: 1) we aim to use it as a proof-of-concept for applying CTDT to continuous-time medical settings, and as a clear evaluation tool to compare against the discrete-time counterpart original DT and other baselines; 2) as a simplified version of the signature kidney transplantation application, this survival experiment shares many of its key features and prepares us before moving on to the main experiment. To this end, we design three groups of experiments hoping to answer the following questions. How would CTDT perform when the offline dataset was logged with a single policy vs. a mixture of policies? Instead of the optimal policy, how would suboptimal policies and policies with bias in treatment decisions affect CTDT's performance? How does a conservative visit schedule affect the treatment outcome?

Environment Description This simple yet illustrative environment is motivated by medical applications with survival outcomes. We assume that each patient has an unobservable underlying health condition $h(t)$ which linearly decreases with time t if no medical interventions are provided. If the health condition drops to a threshold the patient dies and the episode is terminated. At each visitation, a medical agent takes a measurement $m(t)$ for the patient which is based on the underlying $h(t)$ with added noise. Then, based on the measurement $m(t)$ the agent assigns a dosage $c(m(t))$. For each $h(t)$, there exists an optimal dosage $c_{\text{optimal},t}$, and the dosage effect is a piece-wise linear function: the health benefit is proportional to the dosage amount, peaks at the optimal dosage $c_{\text{optimal},t}$, then drops down for larger dosage, and eventually the overly large dosage becomes harmful. At the ith visit, the agent then decides the next visitation

timing t_{i+1}. As our goal is to prolong the patient's survival time, we choose the death indicator $\mathbf{1}_{\text{alive at } t}$ as the reward rate function. As such, the episodic return is the total survival time. Due to limited medical resources, we assume that there is a cap L for the number of visitations. Thus, to achieve high total returns, the agent should schedule the next visitation time that is not too far away in case the patient dies before the next visitation time, or not too close to waste medical resources. We choose the limit for the number of visitations to be $L = 20$, and the theoretical maximum achievable return is 500. We defer the details of the environment to Appendix A.1.

Baselines We compare with DT, which we adapt to this continuous-time setting by putting logged data timestamps into discretized time buckets (days), and at test time DT assigns interval times s in days. We also consider behavior cloning (BC) as a baseline, which utilizes a simple MLP architecture and performs likelihood based imitation learning. Similar to CTDT, we let BC assign both dosages and interval times continuously, and train the two-dimensional policy with a supervised loss. A detailed description of BC can be found in Chen et al. (2021). We also compare to two state-of-the-art model-free offline RL methods called Batch-Constrained deep Q-learning (BCQ) (Fujimoto et al., 2019) and Conservative Q-Learning (CQL) (Kumar et al., 2020), which use Deep Deterministic Policy Gradient (Lillicrap et al., 2015) and Soft Actor-Critic (Haarnoja et al., 2018) , respectively, as a subroutine for continuous action settings. For BCQ and CQL, we set the action space to be two-dimensional composed of dosage and interval times.

Single Behavioral Policy We begin by considering logged datasets generated by a single behavioral policy. We consider low dosage, optimal dosage, and high dosage settings with dosage $c(t)$ at each visit sampled independently from $\mathcal{N}(\mu_{(\cdot)}, 1^2)$, with $\mu_{\text{low}} = c_{\text{optimal},t} - 2, \mu_{\text{opt}} = c_{\text{optimal},t}$, and $\mu_{\text{high}} = c_{\text{optimal},t} + 2$. At each visit, given $c(t)$ the interval time is sampled *dependently* from $\mathcal{N}(s_{\text{optimal},t}, 5^2)$, with the optimal interval time $s_{\text{optimal},t}$ being a function of the assigned dosage $c(t)$ as well as the noisy measurement $m(t)$ (see Appendix A.1). For each of the three settings, we sample 10000 trajectories as logged data. Average returns over 100 episodes of online environment interactions are reported in Table 1.

Single Policy With Biased Timing We now consider logged datasets each containing 10000 trajectories generated by a single policy, but the visitation timings are systematically biased to be conservative. We take three dosage sampling policies identical to the previous set of experiments, and sample the interval times dependently given the dosages as $\mathcal{N}(s_{\text{biased},t}, 5^2))$, with mean interval times biased towards lower values compared to the optimal interval times given by $s_{\text{biased},t} = s_{\text{optimal},t} - 3$. The average returns over 100 episodes are summarized in Table 2.

Table 1: Single Behavior Policy

	Low dosage	**Optimal dosage**	**High dosage**
CTDT	**358.4 ± 20.3**	**435.2± 9.1**	**358.9± 4.8**
DT	345.3± 61.3	397.9± 40.2	188.0±114.5
BC	20.0± 0.7	23.9±0.6	304.0±15.6
BCQ	135.4± 6.7	192.8± 6.5	263.2±79.8
CQL	164.6 ± 12.6	154.3 ± 22.7	151.7 ± 12.3

Table 2: Single Behavior Policy With Biased Timing

	Low dosage	**Optimal dosage**	**High dosage**
CTDT	**356.7 ± 1.3**	379.4± 0.1	**328.0± 11.7**
DT	331.4 ± 42.5	**382.5±38.5**	325.1±5.8
BC	27.1± 11.2	375.5±67.8	318.8±32.3
BCQ	96.1± 6.0	182.2± 5.7	239.9±89.4
CQL	159.7± 24.5	160.2± 11.5	155.1±24.4

Suboptimal Policy And Mixture Of Policies Logged data collected with suboptimal policies that are performing reasonably well but not optimal or mixtures of high- and low-performance policies are prevalent in medical data. In this setting, we consider six scenarios. First, we sample a logged dataset of 10000 trajectories with a suboptimal behavioral policy, where the dosage is sampled from $\mathcal{N}(c_{\text{sub},t}, 1^2)$ with $c_{\text{sub},t} = c_{\text{optimal},t} - 0.5$, and interval times are sampled dependently from $\mathcal{N}(s_{\text{optimal},t}, 5^2)$. Second, using the same suboptimal policy to generate another 5000 trajectories, we mix with 5000 trajectories sampled with a low performance policy where dosage is sampled with μ_{low} as defined in scenario 1. Third, we sample 5000 trajectories with the same low dosage policy mixed with 5000 trajectories sampled with the optimal dosage policy. Finally, for the last three experiments, we consider again data logged with mixture policies, but this time with the conservatively biased timing described in scenario 2 – we consider mixtures of suboptimal dosage policy and low dosage policy, optimal and low dosage policy, and high and low dosage policy, respectively, and in each setting the interval timing has mean $s_{\text{bias}}(c(t))$ for both behavior policies in the mixture. Evaluation results over 100 episodes are reported in Table 3 for the first three experiments, and in Table 4 for the last three experiments.

Performance And Generalization

In all 12 experiments, CTDT performs at least comparably with or better than all three baseline algorithms in terms of averaged return over 100 episodes. In many cases, CTDT signifcantly outperforms the baselines algorithms. Notably, in the mixture settings where the overall logged data quality is dragged down by a low performance behavior policy,

Table 3: Single Suboptimal Policy and Mixture Policies

	Suboptimal dosage	**Sub-low dos. mixture**	**Opt-low dos. mixture**
CTDT	**416.9 ± 1.2**	**427.0± 3.1**	**410.3± 0.6**
DT	393.7± 25.1	361.9± 102.9	302.4±104.8
BC	23.3± 0.4	20.6±0.6	20.2±0.5
BCQ	166.1± 6.5	155.1± 3.4	195.1±8.6
CQL	158.6 ± 20.4	164.2 ± 11.4	159.3 ± 16.9

Table 4: Mixture policies With Biased Timing

	Sub-low dos. mixture	**Opt-low dos. mixture**	**High-low dos. mixture**
CTDT	**383.0 ± 3.9**	**383.7± 1.1**	332.3± 0.7
DT	361.9± 53.9	355.2± 6.5	317.5±15.9
BC	191.9± 43.4	59.6±30.3	**348.1±80.7**
BCQ	167.2± 11.5	185.9± 23.4	157.6±2.7
CQL	162.9± 32.7	159.8± 12.0	151.3±9.0

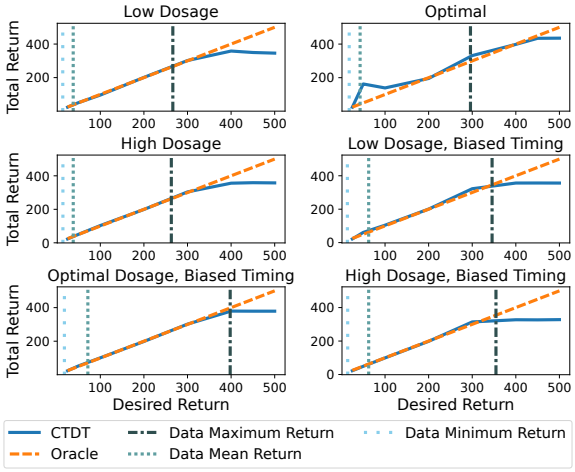

Figure 1: Obtained Return vs. Desired Return

CTDT remains a high performance that is comparable with learning from a single optimal behavior policy. Furthermore, CTDT not only attains high mean returns across all 12 experiments, it does so while achieving much lower variances compared to the baselines, which showcases the high precision in performance afforded by the continuous-time sequence modeling approach. In many cases out of the suite of experiments, BC and BCQ have inferior performances compared to the two return-conditioned sequence modeling methods: whereas BCQ suffers from poor value function approximation and distributional shift resulting from the stochastic behavior policies and environment, BC performance is impeded by the fact that it always simply mimics the ensemble average policy present in the logged data that is not optimal in this survival setting. In most cases, the original DT comes as a close second and outperforms the other two baselines. However, since DT is subject to temporal rounding off errors (due to temporal discretization) and lacks the ability to capture the stage-wise dependence of interval times on clinical treatments, it still lags behind CTDT considerably in overall performance.

Chen et al. (2021) found that by varying the conditional desired returns over a range there was a high correlation between desired target returns and DT's obtained returns on discrete-time offline RL benchmarks. We perform a similar investigation on all 12 experiments presented above, with the first 6 exhibited in Figure 1 and the rest in Appendix C.1. We find that on all of the experiments the survival outcomes are even more strongly correlated with the desired survival time at least up through the maximum return present in the offline training data; in many cases, the correlation trend remains well above the maximum return. This illustrates CTDT's ability to not only accurately map returns to appropriate sequential visitation schedules as well as dosages at visitations, but also generalizes beyond the behavior policy present in the logged data.

4.2 Application On Kidney Transplantation

Problem Description We now turn our attention to the signature kidney transplantation application mentioned in Section 2. Large-scale kidney transplantation databases, such as French computerized and validated data in transplantation (DIVAT), provide us opportunities to determine the optimal follow-up schedules and tacrolimus dosages. DIVAT is a database which monitors medical records for kidney transplantations in several French hospitals (e.g., Nantes, Paris Necker). Data are collected from the date of transplantation until the graft failure. At each scheduled follow-up visit, patients' creatinine levels, an important biomarker for measuring kidney function, and tacrolimus levels are collected longitudinally to investigate therapeutic strategies and determine the next follow-up time by clinicians.

Due to the data privacy of DIVAT, we use a simulated dataset that closely mimics the DIVAT data as our training data (Hua et al., 2021). The dataset composes of longitudinal creatinine measurements, follow-up schedules, tacrolimus dosages, and survival time for $N = 4000$ patients, with 2000 patients treated with a low-variance behavioral policy and 2000 with a policy whose assigned dosage is slightly biased and has a higher variance. Three baseline covariates are considered, including donor age, delayed graft function, and body mass index. Patients' longitudinal creatinine levels are generated from a linear-mixed effects model, in which the fixed and random effects include the time since transplantation, tacrolimus dosage, and baseline covariates. Patients' survival times are generated from a Cox proportional hazards model with a Weibull baseline hazard, which can be affected by patients' creatinine levels, dosage effects,

and times since transplantation. For ease of interpretation, we do not consider censoring times for this experiment. Finally, patients' follow-up schedules and tacrolimus dosages in the behavior policies are generated with linear models that take the baseline covariates, the creatinine measurements, and times since transplantation into account, such that patients with a higher hazard rate receives increased dosages and more frequent clinical visits. We defer more details of this environment and the behavioral policies description to Appendix A.2 and Appendix C.2, respectively.

Table 5: Kidney Transplantation Application Performance

	Median survival time	**Median visit number**
CTDT	**2645.8 ± 2595.1**	11.0±9.0
DT	1792.5±1774.7	6.0±4.0
BC	1267.1 ± 1263.3	15.75± 13.75
BCQ	376.5± 376.3	2.0 ± 0

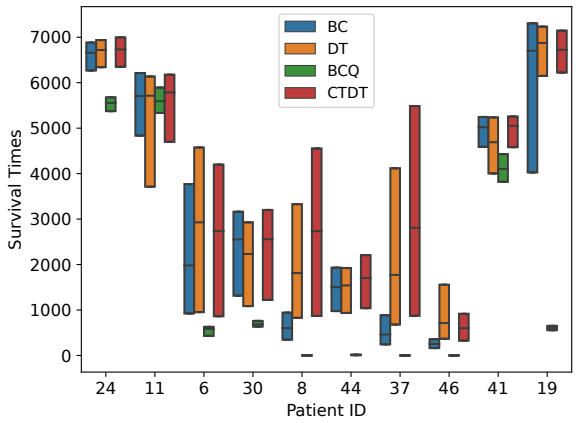

Figure 2: Median Survival Times For 10 Random Patients

Results We consider BC, BCQ (with necessary modification described in Appendix B.1), and DT (with time discretized into days similar to Section 4.1) as baselines for this experiment, and drop CQL due to its constant low performance in the previous synthetic data study. For all algorithms the states include the creatinine measurement and the three baselines. We train CTDT and the three baselines on the kidney dataset logged with the mixture behavior policy and evaluate them using 50 patients with new baseline covariates and random effects. Due to the high variance nature of the survival model, we use median survival time as the evaluation metric for this experiment. For each patient, we use 100 Monte Carlo online evaluations to estimate the median survival time. The overall performance taken as the median over all 50 test patients is summarized in Table 5. An accompanying box plot for the survival times from all 100 evaluations for 10 randomly selected patients is given in Figure 2, and the box plot for all 50 patients are given in Appendix C.2. In this experiment, CTDT greatly outperforms all three baselines, in the sense that compared with the baselines patients can achieve longer expected survival times following the treatment decisions assigned by CTDT. Besides a high metric score, CTDT also assigns clinically meaningful decisions: CTDT achieves longer survival times without much more frequent clinical visits. In fact, akin to the behavior policies, when the patients have stable creatinine levels CTDT tends to assign longer interval times than the baselines (BC in particular, as already indicated by having even more number of visits in spite of the shorter median survival times).

We now zoom into the trajectory-level performance to closely examine the 4 algorithms. In Figure 3, we plot the creatinine level measurement and assigned dosage amount at each visit for a random Monte Carlo trajectory of a randomly sampled patient. Among all 4 algorithms BC clearly assigns the most frequent clinical visits. This is not surprising, as the simple behavior cloning algorithm does not fully capture the time-dependent interval time schedule of the behavioral policies. In contrast, CTDT inherits the more clinically meaningful visit schedule from the behavioral policies: when the creatinine level is stable at the early stages, CTDT gives long interval times; towards the end when creatinine measurements rise up, CTDT assigns more frequent schedules. In terms of visit timing, the baseline DT behaves similarly to CTDT. It is also worth noting that the distributional shift and value function misfit issues of BCQ are apparent in Figure 3: due to the high-variance nature of this survival experiment, as soon as BCQ encounters a less seen measurement value (at the second visit), the assigned dosage, guided by the poorly estimated Q function, quickly becomes unreasonably large; as a result, BCQ performance is considerably inferior compared to the other 3 likelihood-based algorithms in this experiment.

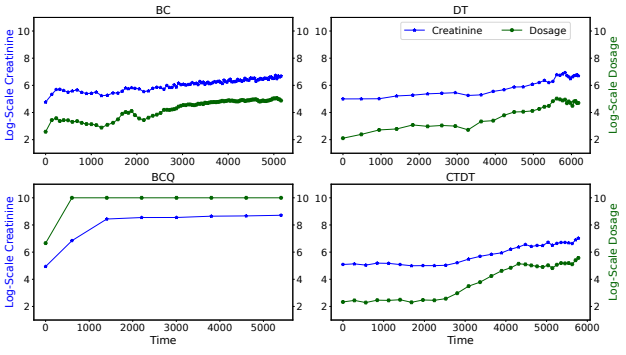

Figure 3: Clinical Measurement And Dosage At Each Visit

4.3 Application On HIV

Problem Description We further consider applying CTDT to adjusting HIV drug combinations in continuous time. How human immunodeficiency virus (HIV) interacts with

the body immune system and drugs was mathematically modeled in Adams et al. (2004). Ernst et al. (2006) first approached optimizing HIV drug combinations in an offline RL setting under this dynamics system where the clinical visitation schedule was fixed. In their setting, the treatment decision was made based on six-dimensional observable states corresponding to 6 clinical measurements: uninfected type i cells T_i, $i \in \{1, 2\}$, infected type i cells T_i^*, free virus V, and immune effectors E, and their dynamics were described by a system of ordinary differential equations. There were 4 discrete actions, corresponding to the on-off combination of two HIV drugs. In Du et al. (2020), the HIV RL problem was generalized to an irregularly spaced visit schedule setting, where patients would pay less frequent clinical visits (though still in integer days) once their immune systems became more stable, with the maximum interval being 14 days. Here, we too consider irregularly spaced visits, except that for our algorithm we allow visits to be scheduled continuously in time.

From visit j to the next, the obtained reward takes a different form and scale from the previous two survival settings:

$$r_j = \int_{t_j}^{t_{j+1}} (-0.1V(t) - f(c_j) + 1000E(t))\, dt,$$

where $f(c_j)$ is a value that depends on the drug combination. The HIV environment is detailed in Appendix A.3.

Baselines And Logged Data For this problem, we compare to 1) the discrete version of BCQ, which builds on Deep Q-Network (Mnih et al., 2015); 2) BC, which assigns interval times in continuous time; and 3) due to the weaker dependence of the interval times on the treatments in the logged dataset and the maximum interval being only 14 days, CTDT has a stronger resemblance to DT in this setting than the previous survival studies, thus we replace DT with an online Neural-ODE based RL (Du et al., 2020) that has been previously applied to the HIV setting, and we adapt it to the offline RL setting by training their dynamics model from only the logged data and giving them the oracle reward function. The ODE RL has a built-in interval time selection function to choose from discrete visit times in days. We use the ODE RL's online pre-trained dynamics model to train an RL agent to convergence, and use the trained agent as a high performance behavioral policy to sample 200 trajectories. Due to the simpler discrete action setting, we lower the high performance trajectories ratio and mix the good trajectories with 800 trajectories sampled with uniformly random drug assignments. Each episode finishes running after a total observation period of 1000 days. A detailed description of the logged data is provided in Appendix C.3.

Results We summarize the online evaluation total returns averaged over 100 episodes in Table 6. Due to the deterministic nature of both the environment and all algorithms except the Neural-ODE RL which softens their policy with $\epsilon = 0.05$, only the ODE RL has variations in the total returns. We outperform all baselines in this mixture behavioral policy experiment by a large margin. Notably, BCQ fails to learn in this setting, but the purely imitation-based BC performs surprisingly well. Besides total achieved returns, we also compare the total number of visits over the 1000 days to check if the temporal relationship can be correctly picked up by the agents. Using Neural-ODE RL's interval time selection function, it very often greedily selects the highest possible interval time (14 days) to obtain maximum per visit reward. In comparison, CTDT achieves the best balance between total returns and the number of visits.

Table 6: HIV Mean Performance

	Mean return	**Mean visit number**
CTDT	**5.5E10**	202
Neural-ODE	(2.1±0.7)E10	**122.1±26.8**
BC	5.1E10	168
BCQ	4.5E7	316

5 DISCUSSION AND CONCLUSION

We extend Decision Transformer (DT) to Continuous-Time Decision Transformer (CTDT) in the context of offline reinforcement learning to estimate and optimize sequential decisions in continuous time, and show its effectiveness in the healthcare/medical domain. Through extensive numerical studies, we demonstrate that the proposed CTDT is able to achieve high returns by correctly estimating the optimal medical decisions in continuous time, and outperform competitors. The analyses of the HIV and kidney datasets yield clinically meaningful and interpretable results, showing that the proposed method has the clinical utility to assist physicians' decisions and improve patients' health outcomes.

The proposed framework has several possible extensions. First, we only consider one type of treatment (e.g., tacrolimus in the kidney example) in this paper. However, in clinical practice, patients may receive other treatments such as mycophenolate mofetil after kidney transplantation, causing potential interaction effects with tacrolimus when used in combination. The proposed CTDT can be extended to allow for multiple treatments and model potential drug-drug interactions. Second, our framework can be easily extended to incorporate stochastic sampling for interval times, which may be more appropriate than the deterministic counterpart in situations (e.g. the future visitation timings are subject to compliance issues). Lastly, both the logged data and the online evaluation environments considered in this work are based on simulations inspired by the real-world medical applications, but they are in no way an accurate surrogate for the actual medical problems; we can benefit from more sophisticated models, while also looking into off-policy evaluations which avoid the need for online evaluation entirely.

Acknowledgements

This work was partially supported by NSF 1940107, NSF 1918854, and NIH R01MH128085 to Dr. Xu, and Adobe Research Gift to Dr. Mei. We thank the anonymous referees for their constructive comments.

References

Aalen, O., Borgan, O., and Gjessing, H. *Survival and event history analysis: a process point of view*. Springer Science & Business Media, 2008.

Adams, B. M., Banks, H. T., Kwon, H.-D., and Tran, H. T. Dynamic multidrug therapies for hiv: Optimal and sti control approaches. *Mathematical Biosciences & Engineering*, 1(2):223, 2004.

Arshad, A., Anderson, B., and Sharif, A. Comparison of organ donation and transplantation rates between opt-out and opt-in systems. *Kidney International*, 95(6):1453–1460, 2019.

Ba, J. L., Kiros, J. R., and Hinton, G. E. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Bai, S., Kolter, J. Z., and Koltun, V. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

Brandfonbrener, D., Bietti, A., Buckman, J., Laroche, R., and Bruna, J. When does return-conditioned supervised learning work for offline reinforcement learning? *arXiv preprint arXiv:2206.01079*, 2022.

Carpenter, S. M., Menictas, M., Nahum-Shani, I., Wetter, D. W., and Murphy, S. A. Developments in mobile health just-in-time adaptive interventions for addiction science. *Current Addiction Reports*, pp. 1–11, 2020.

Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., Abbeel, P., Srinivas, A., and Mordatch, I. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34, 2021.

Chen, R. T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Clifton, J. and Laber, E. Q-learning: Theory and applications. *Annual Review of Statistics and its Application*, 7: 279–301, 2020.

D'Souza, G., Golub, E. T., and Gange, S. J. The changing science of hiv epidemiology in the us. *American Journal of Epidemiology*, 2019.

Du, J., Futoma, J., and Doshi-Velez, F. Model-based reinforcement learning for semi-markov decision processes with neural odes. *Advances in Neural Information Processing Systems*, 33:19805–19816, 2020.

Ernst, D., Stan, G.-B., Goncalves, J., and Wehenkel, L. Clinical data based optimal sti strategies for hiv: a reinforcement learning approach. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pp. 667–672. IEEE, 2006.

Fatemi, M., Killian, T. W., Subramanian, J., and Ghassemi, M. Medical dead-ends and learning to identify high-risk states and treatments. *Advances in Neural Information Processing Systems*, 34:4856–4870, 2021.

Fatemi, M., Wu, M., Petch, J., Nelson, W., Connolly, S. J., Benz, A., Carnicelli, A., and Ghassemi, M. Semi-markov offline reinforcement learning for healthcare. In *Conference on Health, Inference, and Learning*, pp. 119–137. PMLR, 2022.

Fujimoto, S., Meger, D., and Precup, D. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pp. 2052–2062. PMLR, 2019.

Futoma, J., Hughes, M. C., and Doshi-Velez, F. Popcorn: Partially observed prediction constrained reinforcement learning. *arXiv preprint arXiv:2001.04032*, 2020.

Gottesman, O., Johansson, F., Komorowski, M., Faisal, A., Sontag, D., Doshi-Velez, F., and Celi, L. A. Guidelines for reinforcement learning in healthcare. *Nature medicine*, 25(1):16–18, 2019.

Ha, D. and Schmidhuber, J. World models. *arXiv preprint arXiv:1803.10122*, 2018.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. PMLR, 2018.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Hua, W., Mei, H., Zohar, S., Giral, M., and Xu, Y. Personalized dynamic treatment regimes in continuous time: A Bayesian joint model for optimizing clinical decisions with timing. *Bayesian Analysis*, 2021.

Israni, A., Dean, C. E., Salkowski, N., Li, S., Ratner, L. E., Rabb, H., Powe, N. R., and Kim, S. J. Variation in structure and delivery of care between kidney transplant centers in the united states. *Transplantation*, 98(5):520, 2014.

Janner, M., Li, Q., and Levine, S. Offline reinforcement learning as one big sequence modeling problem. *Advances in neural information processing systems*, 34, 2021.

Kasiske, B. L., Zeier, M. G., Chapman, J. R., Craig, J. C., Ekberg, H., Garvey, C. A., Green, M. D., Jha, V., Josephson, M. A., Kiberd, B. A., Kreis, H. A., McDonald, R. A., Newmann, J. M., Obrador, G. T., Vincenti, F. G., Cheung, M., Earley, A., Raman, G., Abariga, S., Wagner, M., and

Balk, E. M. KDIGO clinical practice guideline for the care of kidney transplant recipients: A summary. *Kidney International*, 77(4):299–311, 2010.

Kidambi, R., Rajeswaran, A., Netrapalli, P., and Joachims, T. Morel: Model-based offline reinforcement learning. *Advances in neural information processing systems*, 33: 21810–21823, 2020.

Killian, T. W., Daulton, S., Konidaris, G., and Doshi-Velez, F. Robust and efficient transfer learning with hidden parameter markov decision processes. *Advances in neural information processing systems*, 30, 2017.

Komorowski, M., Celi, L. A., Badawi, O., Gordon, A. C., and Faisal, A. A. The artificial intelligence clinician learns optimal treatment strategies for sepsis in intensive care. *Nature medicine*, 24(11):1716–1720, 2018.

Kumar, A., Zhou, A., Tucker, G., and Levine, S. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33: 1179–1191, 2020.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *nature*, 518(7540): 529–533, 2015.

Orellana, L., Rotnitzky, A., and Robins, J. M. Dynamic regime marginal structural mean models for estimation of optimal dynamic treatment regimes, part i: main content. *The international journal of biostatistics*, 6(2), 2010.

Parbhoo, S., Bogojeska, J., Zazzi, M., Roth, V., and Doshi-Velez, F. Combining kernel and model based learning for hiv therapy selection. *AMIA Summits on Translational Science Proceedings*, 2017:239, 2017.

Peng, X., Ding, Y., Wihl, D., Gottesman, O., Komorowski, M., Li-wei, H. L., Ross, A., Faisal, A., and Doshi-Velez, F. Improving sepsis treatment strategies by combining deep and kernel-based reinforcement learning. In *AMIA Annual Symposium Proceedings*, volume 2018, pp. 887. American Medical Informatics Association, 2018.

Pennings, P. S. Hiv drug resistance: problems and perspectives. *Infectious disease reports*, 5(S1):21–25, 2013.

Popel, M. and Bojar, O. Training tips for the transformer model. *arXiv preprint arXiv:1804.00247*, 2018.

Prasad, N., Cheng, L.-F., Chivers, C., Draugelis, M., and Engelhardt, B. E. A reinforcement learning approach to weaning of mechanical ventilation in intensive care units. *arXiv preprint arXiv:1704.06300*, 2017.

Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. Improving language understanding by generative pre-training. 2018.

Raghu, A., Komorowski, M., Celi, L. A., Szolovits, P., and Ghassemi, M. Continuous state-space models for optimal sepsis treatment: a deep reinforcement learning approach. In *Machine Learning for Healthcare Conference*, pp. 147–163. PMLR, 2017.

Shortreed, S. M., Laber, E., Lizotte, D. J., Stroup, T. S., Pineau, J., and Murphy, S. A. Informing sequential clinical decision-making through reinforcement learning: an empirical study. *Machine learning*, 84(1):109–136, 2011.

Upadhyay, U., De, A., and Gomez Rodriguez, M. Deep reinforcement learning of marked temporal point processes. *Advances in Neural Information Processing Systems*, 31, 2018.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Yang, C., Mei, H., and Eisner, J. Transformer embeddings of irregularly spaced events and their participants. In *ICLR*, 2022.

Yildiz, C., Heinonen, M., and Lähdesmäki, H. Continuous-time model-based reinforcement learning. In *International Conference on Machine Learning*, pp. 12009–12018. PMLR, 2021.

Yu, C., Dong, Y., Liu, J., and Ren, G. Incorporating causal factors into reinforcement learning for dynamic treatment regimes in hiv. *BMC medical informatics and decision making*, 19(2):19–29, 2019.

Yu, C., Liu, J., Nemati, S., and Yin, G. Reinforcement learning in healthcare: A survey. *ACM Computing Surveys (CSUR)*, 55(1):1–36, 2021.

Yu, T., Thomas, G., Yu, L., Ermon, S., Zou, J. Y., Levine, S., Finn, C., and Ma, T. Mopo: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, 33:14129–14142, 2020.

Zhang, M. R., Paine, T. L., Nachum, O., Paduraru, C., Tucker, G., Wang, Z., and Norouzi, M. Autoregressive dynamics models for offline policy evaluation and optimization. *arXiv preprint arXiv:2104.13877*, 2021.

Zhao, Y.-Q., Zeng, D., Laber, E. B., Song, R., Yuan, M., and Kosorok, M. R. Doubly robust learning for estimating individualized treatment with censored data. *Biometrika*, 102(1):151–168, 2015.

Zuo, S., Jiang, H., Li, Z., Zhao, T., and Zha, H. Transformer hawkes process. In *International conference on machine learning*, pp. 11692–11702. PMLR, 2020.

A ENVIRONMENTS DETAILS

In this section, we detail the three environments used in the experiments in Section 4.

A.1 Synthetic Data Study

For the survival experiment in Section 4.1, We assume that each patient's underlying health condition is $h(t)$, which linearly decreases with time t if no medical interventions are provided: $h(t) = h_0 - d * t$, where h_0 is the initial health condition, and d is a constant decline rate. Each patient's health condition is assumed to be upper bounded by h_{max} and the patient dies at time t if the health condition $h(t)$ drops to 0. We denote $m(t)$ to be a noisy clinical measurement of the health condition: $m(t) \sim \mathcal{N}(h(t), \sigma_m^2)$.

At each visitation, a medical agent assigns a dosage $c(m(t))$ based on the clinical measurement $m(t)$. We suppress the measurement dependence in $c(m(t))$ and use $c(t)$ instead for notation simplicity. The assigned dosage gives an instant surge to the patient's health condition by $f(h(t), c(t))$, which is a piece-wise linear function given by

$$f(h(t), c(t)) = \begin{cases} c(t) & \text{if } c(t) \leq c_{\text{optimal},t}, \\ 2c_{\text{optimal},t} - c(t) & \text{if } c(t) > c_{\text{optimal},t}, \end{cases}$$

where $c_{\text{optimal},t}$ is the optimal dosage for patient to achieve the state with the maximum health condition h_{max}. Such a design makes the dosage effect on health condition increase first when the dosage increases from 0, peak at the dosage of $c_{\text{optimal},t}$, then decrease when the dosage continues to increase, and eventually become harmful (negative value) when the dosage is higher than $2c_{\text{optimal},t}$. We choose $c_{\text{optimal},t}$ to be a simple health condition dependent function that is unknown to the agent but known to the behavior policy: $c_{\text{optimal},t} := h_{\text{max}} - h(t)$.

Denoting the current visit index by i, the agent then decides the next visitation timing t_{i+1}. As our goal is to prolong the patient's survival time, we choose the death indicator function $\mathbf{1}_{\text{alive at } t}$ as the reward rate function. As such, the reward from visit i to visit $i+1$ is $t_{i+1} - t_i$ if the patient survives until t_{i+1}; if the patient dies at time $g \in (t_i, t_{i+1})$, the reward is $g - t_i$. Due to limited medical resources, we assume that there is a cap L for the number of visitations. Thus, to achieve high total returns, the agent should schedule the next visitation time that is not too far away in case the patient dies before the next visitation time, or not too close to waste medical resources. Given dosage $c(t)$ and noisy measurement $m(t)$, the optimal interval time based on the noisy observation $m(t)$ is computed as

$$s_{\text{optimal},t} = \begin{cases} \left(\frac{h_{\text{max}} - m(t)}{c_{\text{optimal},t}} \cdot c(t) + m(t) \right) / d & \text{if } c(t) \leq c_{\text{optimal},t}, \\ \left(\frac{h_{\text{max}} - m(t)}{c_{\text{optimal},t}} \cdot (2c_{\text{optimal},t} - c(t)) + m(t) \right) / d & \text{if } c(t) > c_{\text{optimal},t}. \end{cases}$$

The theoretical maximum achievable return is bounded by $L \cdot h_{\text{max}}/d$. We choose the limit for the number of visitations to be $L = 20$, death rate $d = 0.5$, maximum health condition $h_{\text{max}} = 12.5$, the initial health condition $h_0 = 0.5$, measurement variance $\sigma_m^2 = 0.5^2$. The maximum achievable return for this episodic problem is therefore upper bounded by 500.

A.2 Application On Kidney Transplantation

Environment Description Considered in Section 4.2, the French computerized and validated data in transplantation (= Données Informatisées et VAlidées en Transplantation, or DIVAT) database (`www.divat.fr`) provides patients medical records collected from the date of kidney transplantation until their death, graft failure, or being censored. Hua et al. (2021) designed a probabilistic model with structures suggested by medical domain experts to generate data that closely mimic the DIVAT data. They jointly modeled the clinical decisions (dosages and treatment timings) and observations (clinical measurements and patient survival). For our experiment in Section 4.2, we modify and implement their observations submodel in `Python` as our online evaluation environment.

In the observations submodel, each patient has three baseline covariates: donor age (AgeD), body mass index (BMI), and delayed graft function (DGF), i.e., $\mathbf{x} = (\text{AgeD}, \text{BMI}, \text{DGF})$. The logarithm of creatinine clinical measurement is modeled by a linear mixed effects model:

$$m(t) \sim \mathcal{N}(m^*(t), \sigma_l^2), \tag{4}$$

with the unobserved true creatinine process $m^*(t) = \mathbf{z}(t)\boldsymbol{\beta}_\mathbf{l} + \mathbf{r}(t)\mathbf{b}$, where $\boldsymbol{\beta}_\mathbf{l}$ is the fixed effect, and the random effect $\mathbf{b} \sim \mathcal{N}(0, \boldsymbol{\Sigma}_\mathbf{b})$. The covariate vectors $\mathbf{z}(t)$ and $\mathbf{r}(t)$ corresponding to fixed and random effects respectively are given by

$$\mathbf{z}(t) = (1, c(t), \mathbf{x}, t, t^2) \text{ and } \mathbf{r}(t) = (1, c(t), t),$$

with the interval dosage defined as $c(t) = c_j$ for $t \in [t_j, t_{j+1})$. The dependence of $\mathbf{z}(t)$ and $\mathbf{r}(t)$ on dosage $c(t)$ reflects the tacrolimus dosage effect on the creatinine level exhibited in the real DIVAT data. Next, the graft failure/death event time is

modeled by a time-to-event model. Specifically, a Weibull proportional hazards model that depends on clinical decisions and the true creatinine process $m^*(t)$ is used, with the hazard function $\lambda(t)$ given by

$$\lambda(t) = \exp\Big(-(\beta_{s_1} m^*(t)^2 + \beta_{s_2} c(t) + \lambda_0)\Big)\omega t^{\omega-1}, \tag{5}$$

where in the above $\beta_{s_1} m^*(t)^2$ is the longitudinal creatinine effect, $\beta_{s_2} c(t)$ the instantaneous dosage effect, λ_0 the baseline hazard, and ω the shape parameter. With such a model, the dosage effect is initially beneficial, but for overly large amount the dosage effect then becomes harmful.

The reward function is the same survival reward described in the previous synthetic data study, namely,

$$r_j = \int_{t_j}^{t_{j+1}} \mathbf{1}_{\{\text{the patient is alive at time } t\}} dt.$$

Parameter Specification The model parameters are chosen such that the model fits well to the real DIVAT data. For each patient, AgeD and BMI are sampled from $\mathcal{N}(52.5, 15.8^2)$ and $\mathcal{N}(24.3, 4.5^2)$, respectively, and then standardized, and DGF $\sim$ Bernoulli(0.4).For the linear mixed effects model Equation (4), we set $\boldsymbol{\beta_1} = (4, 0.5, 0.3, 0.4, 0.25, -1 \times 10^{-4}, 3 \times 10^{-8})^T$, $\sigma_l^2 = 0.1^2$, and $\mathbf{\Sigma_b}$ is a diagonal matrix with diagonal entries $(0.04, 0.0049, 10^{-8})$. For the parameters in the hazard function Equation (5), we set $\beta_{s_1} = -0.67$, $\beta_{s_2} = 4.2$,$h_0 = 18.5$, and $\omega = 1.25$. Each patient's initial logarithm of creatinine level $m(0) \sim \mathcal{N}(5, 0.1^2)$.

A.3 Application On HIV

Environment Description For the experiments in Section 4.3, we use the ordinary differential equations (ODEs) representation of the human immunodeficiency virus (HIV) infection dynamics described in Adams et al. (2004), which is a well-studied RL benchmark (Ernst et al., 2006; Killian et al., 2017; Yu et al., 2019; Du et al., 2020). The `Python` implementation is available at `github.com/dtak/mbrl-smdp-ode`. At each visit, 6 measurements are available: uninfected type i cells T_i, $i \in \{1, 2\}$, infected type i cells T_i^*, free virus V, and immune effectors E. The system of ODEs describing their dynamics is:

$$\begin{aligned}
\dot{T_1} &= \lambda_1 - d_1 T_1 - (1-\epsilon_1) k_1 V T_1, \\
\dot{T_2} &= \lambda_2 - d_2 T_2 - (1-f\epsilon_1) k_2 V T_2, \\
\dot{T_1^*} &= (1-\epsilon_1) k_1 V T_1 - \delta T_1^* - m_1 E T_1^*, \\
\dot{T_2^*} &= (1-f\epsilon_1) k_2 V T_2 - \delta T_2^* - m_2 E T_2^*, \\
\dot{V} &= (1-\epsilon_2) N_T \delta (T_1^* + T_2^*) - cV - ((1-\epsilon_1)\rho_1 k_1 T_1 + (1-f\epsilon_1)\rho_2 k_2 T_2) V, \\
\dot{E} &= \lambda_E + \frac{b_E (T_1^* + T_2^*)}{(T_1^* + T_2^*) + K_b} E - \frac{d_E (T_1^* + T_2^*)}{(T_1^* + T_2^*) + K_d} E - \delta_E E,
\end{aligned}$$

where $\epsilon_1 = 0.7$ (resp., 0) if type 1 drug is on (resp., off), and $\epsilon_2 = 0.3$ (resp., 0) if type 2 drug is on (resp., off). We modify the reward definition to be suitable for our continuous-time generalization: we turn the discrete reward into a reward rate function, and integrate over interval times to obtain the interval reward:

$$r_j = \int_{t_j}^{t_{j+1}} (-0.1V(t) - 20000\epsilon_1^2(t) - 2000\epsilon_2^2(t) + 1000E(t)) dt.$$

Since we do not assume access to the intermediate states, we linearly approximate the integration above with the trapezoid rule using the two end-point states. The observation period for each patient ends after 1000 days.

Parameter specification The initial condition of each patient at $t = 0$ is set to be a non-healthy state: $(T_1, T_2, T_1^*, T_2^*, V, E)(0) = (163573, 5, 11945, 46, 63919, 24)$. The dynamics model parameters in the ODEs are chosen according to Table 1 in Adams et al. (2004).

B EXPERIMENTAL DETAILS

In this section, we first present the necessary modifications to the baseline algorithms that we make to adapt them to the continuous-time offline RL setting. Then, we report the training details, hyperparameters, and computing information for all algorithms used in Section 4.

B.1 Baseline Modifications

To adapt the baseline algorithms to our experimental settings, we perform the following necessary modifications.

DT We implement DT based on `github.com/kzl/decision-transformer`. For both experiments in Section 4.1 and Section 4.2, we put the logged data timestamps into discretized time buckets (days), and at test time DT assigns interval times s in days.

BCQ We implement BCQ based on `github.com/sfujim/BCQ`. For the continuous-action settings in Section 4.1 and Section 4.2, we treat the dosage assignment and the interval time as a two-dimensional action. Due to the mismatched scales of dosages and interval times (with interval times typically much larger than dosages) in the Section 4.2 DIVAT kidney experiment, we perform a standardization transformation on the logged actions for BCQ training, and reverse transform the output actions for evaluation. Doing so introduces an extra tunable hyperparameter $a_{\max}$, the maximum action value (in terms of standard deviations), which we set to 3 for Section 4.2. For the discrete-action setting in Section 4.3, the drug combination and interval time are encoded into one single action, with the total number of possible actions now being $4 \times 14 = 56$ (4 drug combinations, integer interval days 1 through 14).

CQL We implement CQL based on `github.com/BY571/CQL`. Similar to BCQ, for CQL we treat the dosage assignment and the interval time as a two-dimensional action. Since the action range in Section 4.1 is no longer bounded in $[-1, 1]$, we remove the final $tanh(\cdot)$ operation for the actor, and change the actor update rule accordingly.

Neural-ODE MBRL We implement the MBRL algorithm based on `github.com/dtak/mbrl-smdp-ode`. The MBRL algorithm is an online algorithm, and the discrete-action version of the algorithm learns world models (Ha & Schmidhuber, 2018) to output learnt policies after having trained a dynamics model via online interactions. Therefore, we adapt the MBRL algorithm to the offline learning setting by forbidding online interactions at the dynamics model training stage and instead training solely with logged data; afterwards, we allow access to the oracle reward function at the world model learning stage to complete policy training, which is slightly advantageous compared to the full offline training procedure (without access to the oracle reward function) of the other algorithms. A further modification is that, since the MBRL only anticipates the interval time distributions but does not actively set the interval times, we use the output of the interval time optimization algorithm (Algorithm 1 in Du et al. (2020)) as the interval time assignment, with again access to the oracle reward function during both policy training and evaluation.

B.2 Training Details

Early Stopping We perform early stopping for training of all algorithms in Section 4 to prevent overfitting: after at most every 10% of the total number of training steps, each algorithm is validated over 100 online episodes, and the validation scores are obtained by averaging the total returns of all the evaluation episodes.

Model Hyperparameters And Optimizer For CTDT, we use the AdamW optimizer (Loshchilov & Hutter, 2017). The first 10% training steps are warmup steps (Popel & Bojar, 2018) with much smaller learning rates. The CTDT model hyperparameters are summarized in Table 7.

Table 7: CTDT Hyperparameters

	Synthetic Data	**Kidney**	**HIV**
Training Steps	1E5	1E5	7500
Batch Size	64	64	4
Value Embedding $\mathbf{y}$ Dimension	128	128	96
Type embedding $\mathbf{e}$ Dimension	64	64	32
Temporal Embedding $\mathbf{t}$ Dimension	64	64	32
Number Of Layers	3	3	4
Number Of Attention Heads	1	1	4
Learning Rate	0.0001	0.0001	0.006
Attention Window Size	20	20	unlimited

BC and DT both have the same number of layers, hidden dimensions, and number of attention heads (DT) as CTDT for each experiment. For BCQ, CQL, and Neural-ODE MBRL, we use the respective codebase's default/recommended hyperparameters.

Computing Information All algorithms used in the experiments in Section 4 are trained on a single A100 GPU. With the

hyperparameters stated in Table 7, the training time for CTDT is less than an hour for the synthetic data experiments, less than an hour for the DIVAT kidney experiment, and less than 1.5 hours for the HIV experiment.

C ADDITIONAL EXPERIMANTAL RESULTS

In this section, we detail the behavior policies for the experiments in Section 4.2 and Section 4.3, and give the logged data statistics for all three experiments in Section 4. We also provide supplementary results for Section 4.1 and Section 4.2.

C.1 Synthetic Data Study

Chen et al. (2021) found that by varying the input desired returns over a range there was a high correlation between desired target returns and DT's obtained returns on discrete-time offline RL benchmarks. We perform a similar investigation on all 12 experiments presented in Section 4.1, varying $\hat{R}_{\text{input}} \in [25, 50, 100, 200, 300, 400, 450, 500]$, ranging from the minimum return in the dataset to the upper bound of achievable returns. The obtained return as a function of the desired return for all the 12 settings is plotted in Figure 4. In all the settings, the survival outcome obtained by the trained CTDT is strongly correlated with the desired survival time at least up through the maximum return present in the offline training data (the logged data statistics for the 12 offline datasets are summarized in Table 8); in many cases, the correlation trend remains well above the maximum return, showcasing CTDT's ability to generalize beyond the best performing trajectories for this survival experiment. For the definitions of e.g. the low dosage policy and biased timing in Figure 4 and Table 8, please refer to Section 4.1.

Table 8: Synthetic Data Study Logged Data Statistics

	Min Return	**Mean Return**	**Max Return**	**Std.**
Low Dosage	13.8	37.8	266.3	23.7
Optimal Dosage	17.8	42.8	296.0	27.0
High Dosage	14.2	37.9	262.9	23.9
Low Dosage, Biased Timing	13.6	62.4	345.5	48.3
Optimal dosage, Biased Timing	17.6	71.4	397.9	55.7
High dosage, Biased Timing	14.3	62.4	354.4	48.1
Suboptimal Dosage	16.8	42.4	292.2	26.8
Mixture Of Optimal And Low Dosage Policies	13.8	40.4	296.0	25.7
Mixture Of Suboptimal And Low Dosage Policies	13.8	40.2	292.2	25.6
Mixture Of Suboptimal And Low Dosages, Biased Timing	13.6	66.5	392.7	51.8
Mixture Of Optimal And Low Dosages, Biased Timing	13.6	66.9	396.9	52.3
Mixture Of High And Low Dosages, Biased Timing	13.6	62.5	354.4	48.2

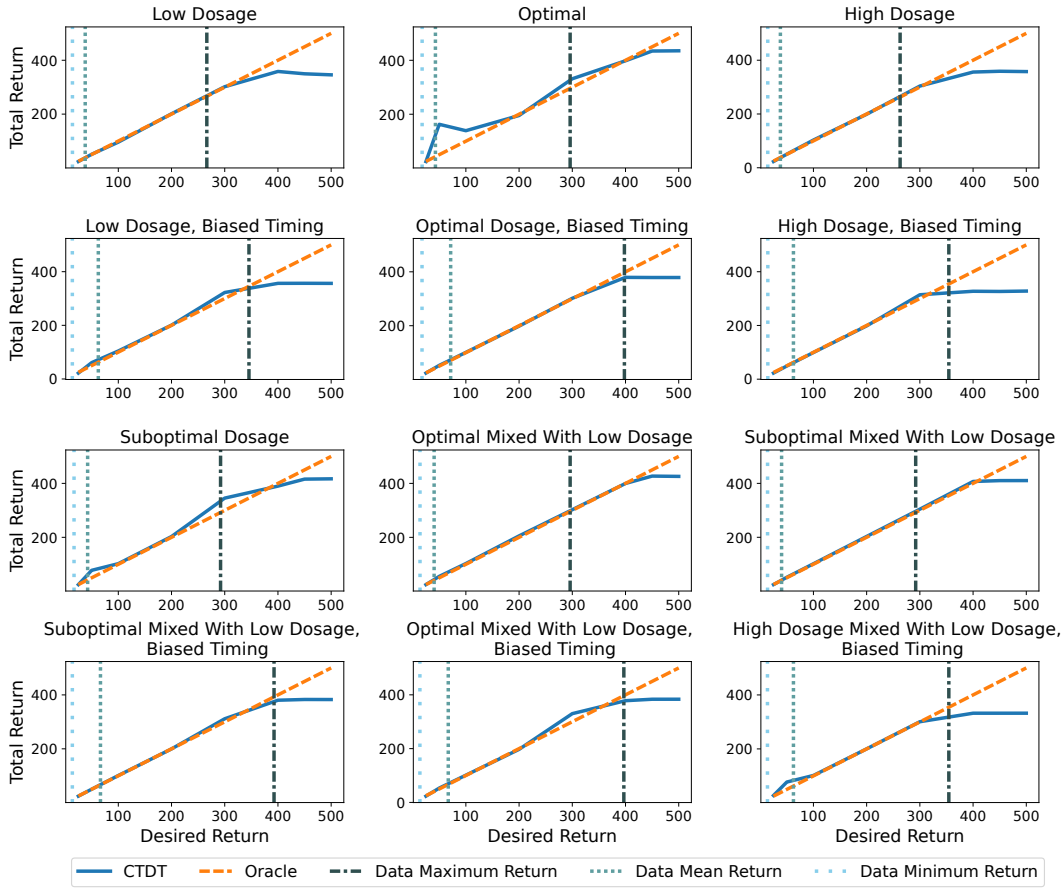

Figure 4: Obtained Return vs. Desired Return

C.2 Application On Kidney Transplantation

In this subsection, we begin by detailing the mixture of behavioral policies used to generate the offline dataset for the kidney transplantation experiment in Section 4.2. Then, we compare the performance of the 4 algorithms in Section 4.2 to the median performance in the logged data, and present the evaluation statistics on all 50 test patients for the 4 algorithms.

The logged dataset contains trajectories for 4000 simulated patients, with 2000 patients treated with a low-variance behavioral policy and 2000 with a policy whose assigned dosage is slightly biased and has a higher variance. The two behavioral policies use linear models to sample the dosage amount and the interval time at each visit. For the low-variance behavioral policy, the dosages are sampled from $\mathcal{N}(c_{\text{mean}}(t), 0.01^2)$, with $c_{\text{mean}} = \mathbf{d(t)}\boldsymbol{\beta_c}$. The covariate vector $\mathbf{d}(t) = (1, m(t), \mathbf{x})$ consists of the current measurement and the patient's baseline covariates, and we set $\boldsymbol{\beta_c} = (-3, 1.2, 0.15, 0.2, 0.15)$. Given the sampled dosage $c(t)$, the interval time is sampled from $\mathcal{N}(800 - \boldsymbol{\tau(t)}\boldsymbol{\beta_s}, 5^2)$, with $\boldsymbol{\tau(t)} = (c(t), t, m(t))$, and $\boldsymbol{\beta_s} = (60, 0.04, 30)$. For the high-variance policy, the dosages are sampled from $\mathcal{N}(c_{\text{bias}}(t), 1^2)$, with $c_{\text{bias}}(t) = c_{\text{mean}}(t) - 1$, and the interval times are obtained in the same way as the low-variance behavioral policy. We clip the samples such that the sampled dosages are in the range $[1, 10]$, and the interval times are in $[30, 800]$. [1]

The first quartile survival time, median survival time, third quartile survival time, and median absolute deviation of the logged dataset are given in Table 9. While BCQ's population-level evaluation score (377) is much lower than the median and BC's (1267) is close to the median, DT (1793) is able to surpass the median performance of the logged dataset. Owing to the extra temporal precision and dosage-dependent interval time assignment, CTDT (2646) is able to further improve beyond DT's performance. In Section 4.2 we used a box plot to show the survival times of all 100 Monte Carlo trajectories

[1] For the online policy evaluations we perform the same clipping for the interval times, and $[0.5, 10]$ for the dosages.

for 10 randomly sampled patients. Here, for completeness we plot the survival times obtained by the 4 algorithms for all 50 test patients in groups of 10 in Figure 5. As can be readily seen, CTDT also achieves best or close to best survival outcomes at the individual patient level.

Table 9: Kidney Transplantation Logged Data Statistics

Q1 Survival Time	Median Survival Time	Q3 Survival Time	Mad.
117.3	1289.1	5388.69	1277.1

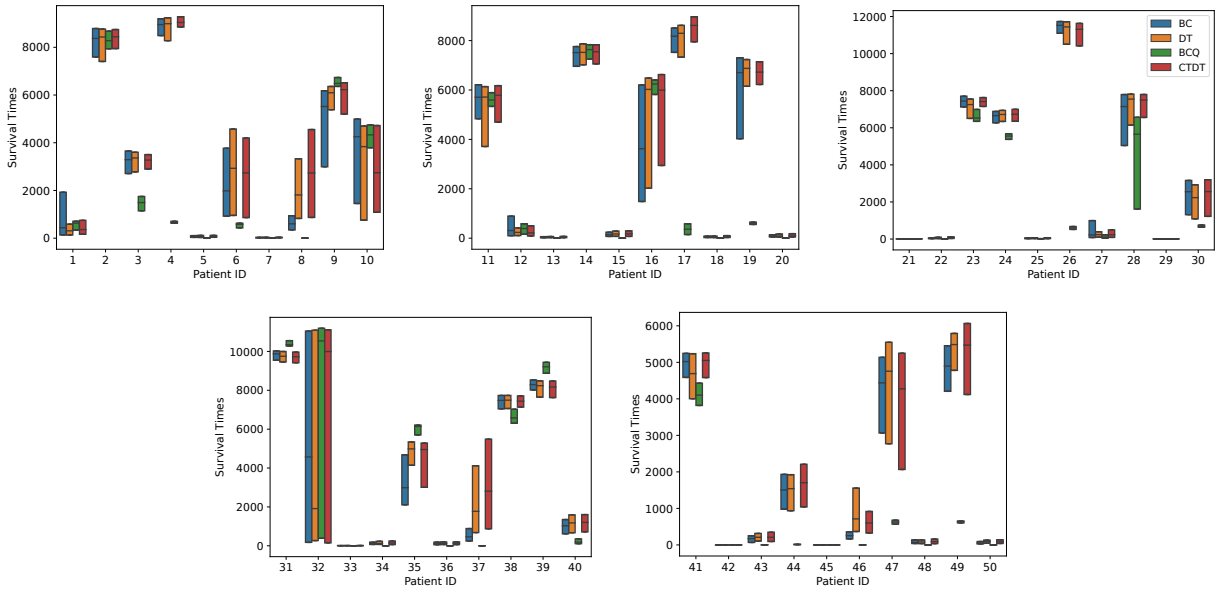

Figure 5: Survival Times For ALL 50 Test Patients

C.3 Application On HIV

We now describe the generation of the logged dataset in Section 4.3. The logged dataset contains a total of 1000 trajectories, with 200 trajectories sampled by a high-performance behavioral policy mixed with 800 trajectories sampled by a low-performance policy. Using a pre-trained dynamics model for the HIV environment provided by Du et al. (2020) at `github.com/dtak/mbrl-smdp-ode`, we train a Neural-ODE MBRL policy to obtain the high-performance policy. With $\mathcal{U}\{a, b\}$ denoting the discrete uniform distribution over support $\{a, a+1, ..., b-1, b\}$, the interval times s are sampled according to

$$s \sim \begin{cases} \mathcal{U}\{11, 14\} & \text{if } (V \leq 10^4 \text{ and } E > 8 \cdot 10^4) \text{ and (no treatment or drug 1)}, \\ \mathcal{U}\{3, 7\} & \text{if } (V \leq 10^4 \text{ and } E > 8 \cdot 10^4) \text{ and (drug 2 or both drugs)}, \\ \mathcal{U}\{3, 7\} & \text{if } (V \leq 10^4 \text{ and } E \leq 8 \cdot 10^4) \text{ and (any or no treatment)}, \\ \mathcal{U}\{3, 7\} & \text{if } (10^4 < V \leq 10^5) \text{ and (no treatment)}, \\ \mathcal{U}\{3, 5\} & \text{if } (10^4 < V \leq 10^5) \text{ and (drug 1 or drug 2)}, \\ 3 & \text{if } (10^4 < V \leq 10^5) \text{ and (both drugs)}, \\ 3 & \text{if } (V > 10^5) \text{ and (no treatment)}, \\ \mathcal{U}\{1, 2\} & \text{if } (V > 10^5) \text{ and (any treatment)}. \end{cases} \tag{6}$$

For the low-performance policy, we sample drug combinations uniformly on the 4 combinations, and use the same interval time schedule given in Equation (6). The dataset statistics are given in Table 10. Since the uniform sampling is a high-variance behavioral policy, the logged dataset has a high standard deviation in total returns.

In Section 4.3 we already showed that CTDT outperforms all baseline algorithms in this experiment. Now, compared to the highly noisy logged dataset, we see that the total return achieved by CTDT (5.5E10) is also very close to the maximum return seen in the logged dataset, in spite of the prevalence of low-performance trajectories.

Table 10: HIV Logged Data Statistics

Min Return	**Mean Return**	**Max Return**	**Std.**
2.6E7	8.1E9	5.7E10	1.7E10