Decision-Focused Model-based Reinforcement Learning for Reward Transfer

Abhishek Sharma ABHISHEKSHARMA@G.HARVARD.EDU
SEAS, Harvard University
Allston, MA, USA

Sonali Parbhoo S.PARBHOO@IMPERIAL.AC.UK
Imperial College London
London, UK

Omer Gottesman OMERGOTT@GMAIL.COM
Amazon
New York, NY, USA

Finale Doshi-Velez FINALE@SEAS.HARVARD.EDU
SEAS, Harvard University
Allston, MA, USA

Abstract

Model-based reinforcement learning (MBRL) provides a way to learn a transition model of the environment, which can then be used to plan personalized policies for different patient cohorts and to understand the dynamics involved in the decision-making process. However, standard MBRL algorithms are either sensitive to changes in the reward function or achieve suboptimal performance on the task when the transition model is restricted. Motivated by the need to use simple and interpretable models in critical domains such as healthcare, we propose a novel robust decision-focused (RDF) algorithm that learns a transition model that achieves high returns while being robust to changes in the reward function. We demonstrate our RDF algorithm can be used with several model classes and planning algorithms. We also provide theoretical and empirical evidence, on a variety of simulators and real patient data, that RDF can learn simple yet effective models that can be used to plan personalized policies.[1]

1. Introduction

Reinforcement learning (RL) is the branch of machine learning where an agent learns to make optimal decisions (with respect to a reward) by interacting with an environment. One example is an algorithm (agent) that learns to provide treatments (decisions/actions) after observing how the patient's health evolves (environment), with a reward function that penalizes unhealthy patient states. This evolution of health is determined by the patient's transition dynamics, i.e., how a patient's state will change based on the current state and the chosen treatment. For example, it could predict how a patient's blood pressure will change after they are given a treatment like vasopressor. Model-based RL (MBRL) learns a *model of the transition dynamics* (i.e. a transition model) to plan actions, while model-free

1. The code for our method is available at https://github.com/dtak/robust_decision_focused_rl_public

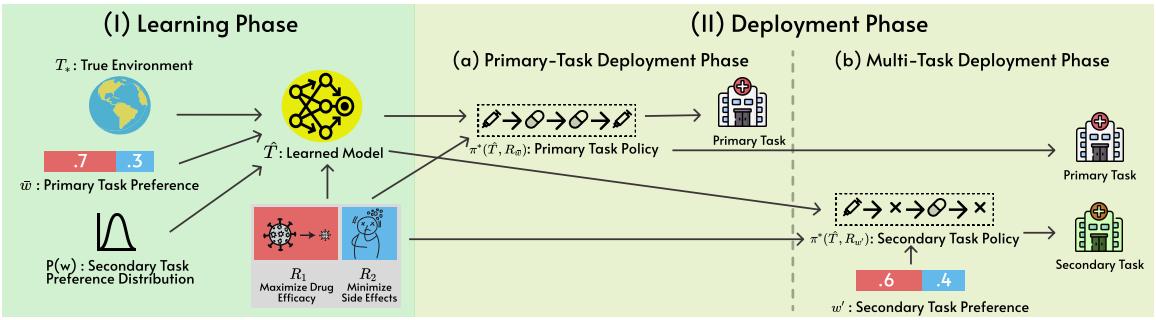

Figure 1: Overview of the setting.

RL learns to take actions directly from experience without learning a transition model. MBRL learns the transition model by predicting the next state given the current state and action. This is also known as maximum likelihood estimation (MLE) because it learns the model that maximizes the likelihood of the observed data.

The reward function is important in RL because it defines the objective of the agent. However, specifying a reward function can be difficult because there are often multiple objectives to consider, the preferences between objectives can change over time or between patients, and the practitioner may only have a vague idea of what the preferences should be. Clinicians often have a sense of key objectives they want a reward function to encapsulate, but may want to support different trade-offs when the objectives are competing. For example, they may want to trade off long- and short-term health-effects or balance the aggressiveness of treatments with other measures of a patient's well-being (Lizotte et al., 2010) In this paper, we consider a similar setting where the reward function can change—between a *learning phase* and a *deployment phase* (Fig 1). As an example in cancer treatment, during the learning phase, we may learn a transition model based on the data from one hospital that prioritizes maximizing drug efficacy over patient side-effects. At deployment, however, the learned model may need to be applied in an alternate scenario, e.g., palliative care where the focus may shift to minimizing side effects over maximizing drug efficacy. Here, the transition model captures the disease dynamics in the body that are expected to be the same across hospitals, although treatment preferences change. Ideally, we want to learn a transition model that is *robust* to changes in the reward preferences at deployment, where robustness refers to the fact that model performs well across different reward preferences during both the learning and deployment phases.

Learning a transition model using MBRL offers several benefits. First, MBRL methods are more data-efficient than model-free methods, as they can leverage the model to simulate patient trajectories and plan actions (Deisenroth, 2011). Second, they allow us to build an understanding of the transition dynamics, which can be useful for understanding how patients respond to treatments. Finally, they allow reusing the learned model to plan treatments for new patients or when the reward function changes for the same patient. However, the choice of model class for modeling transition dynamics is important. While complex model classes (e.g., neural networks) can be expressive, they are difficult to interpret and are prone to overfitting, especially with limited data. Simple model classes (e.g., linear

models) may not model every nuance of the dynamics, but they use less data to train, are more robust to overfitting and can be inspected to understand the learned relationships between variables.

However, using a simple model class introduces errors in the model predictions. The typical model learning method of maximum likelihood estimation (MLE) minimizes the prediction error but the decision policies learned using such models are no longer guaranteed to be good (Joseph et al., 2013). Decision-focused (DF) learning is an alternative to MLE which selects the model that maximizes the eventual return of the decisions, rather than just minimizing prediction error (Grimm et al., 2020). This approach has been shown to have better performance than maximum likelihood estimation (MLE) when the transition model is misspecified, as it focuses on the most important aspects of the environment for decision-making (Wilder et al., 2019; Sharma et al., 2021). For example, if the reward function is designed to avoid hypotension, the decision-focused model will focus on predicting the patient's blood pressure accurately if it has to choose between predicting blood pressure and other variables. However, DF methods are "overfitting" by learning a model that maximizes the return with respect to the current reward function (Wang et al., 2021; Nikishin et al., 2022), which can lead to suboptimal decisions when the reward function changes. This is a problem in healthcare because the reward preferences of a clinician can change over time. Prior work (Futoma et al., 2020) addressed this issue of "overfitting" by combining the decision-focused learning objective with a maximum likelihood estimation (MLE) objective. This was done by maximizing the likelihood of the observed data, while also constraining the model to produce high returns under the (fixed) reward function. Although promising, this approach fails to generalize to new reward functions that are very different from the one used during training.

We address these challenges faced by MBRL for simple model classes by introducing a *robust decision-focused (RDF) learning* objective to learn a simple model that performs well across different reward preferences. We show that there are several possible models consistent with the decision-focused objective (i.e., they are non-identifiable), and RDF leverages this non-identifiability to learn a model that is robust to changes in the reward function. RDF trades off the decision-focused objective (optimized to have high returns with the learning-phase reward function) with the averaged decision-focused objective (over a distribution of reward functions expected at deployment-phase). Our requirements for this reward distribution are minimal: we only require that the practitioner can provide the boundaries of the distribution. We develop a novel algorithm that allows us to perform such optimization efficiently. In addition to characterizing the RDF objective, we provide a theoretical analysis that shows RDF can achieve better performance than MLE and DF methods. We also demonstrate the effectiveness of RDF learning of simple and interpretable transition models for a synthetic simulator, a cancer simulator, and a real-world healthcare dataset for hypotension management.

Generalizable Insights about Machine Learning in the Context of Healthcare

Model-based RL methods show promise in discovering better treatment policies from historical data, but the sensitivity of these methods to changes in rewards at deployment time poses a key challenge to their adoption in high-stakes clinical domains. Our main hypothe-

sis is that we achieve robustness across different reward settings by optimizing for average performance while learning a transition model in RL. Our work provides one of the first solutions to learning transition dynamics models that are robust to these changes and can inform future research on applying MBRL to problems in healthcare. Our contributions are as follows:

- We show that DF learning objective is non-identifiable, i.e, there are multiple models that can achieve the same decision-focused objective.
- We introduce RDF learning objective that leverages this non-identifiability to learn a model that is robust to changes in the reward preferences, while continuing to have high returns under the learning-phase reward function. We provide a theoretical analysis that shows RDF models can achieve better decision quality than MLE and DF models.
- On a suite of synthetic and real-world healthcare datasets, we demonstrate that RDF models outperform MLE and DF models in terms of decision-making performance on new reward functions.

2. Related Work

Decision-Focused Model-based Reinforcement Learning. Several works have developed algorithms for decision-focused (DF) model-based reinforcement learning (Joseph et al., 2013; Farahmand et al., 2017; Wang et al., 2021; Nikishin et al., 2022; Futoma et al., 2020). These works focus on building simple models, where the transition model cannot represent the true transition dynamics, that can be used for planning. In contrast to these, we consider the setting where the learned model must be re-used for different reward functions.

Grimm et al. (2020) and Nikishin et al. (2022) note that DF models can be non-identifiable for a given reward function. Both suggest this can be a good thing: if the only evaluation metric is the return of the RL agent (i.e. the DF objective), it is easier to find an optimal solution when multiple equivalent solutions of the DF objective exist. However, they do not consider the robustness of different DF solutions to changes in reward function. In contrast, we show that the DF objective does not optimize for robustness, and may find a solution that is not robust to changes in reward. Furthermore, we exploit this non-identifiability of models with respect to the performance of the agent to find a solution that generalizes better to the changed reward function.

RL with changing objectives. Multi-Objective RL (MORL) and reward-robust RL aim to train agents to perform well on multiple reward functions (Barrett and Narayanan, 2008; Lizotte et al., 2010; Mossalam et al., 2016; Abels et al., 2019; Husain et al., 2021; Derman et al., 2021). The most common setting addressed in the MORL literature involves rewards functions linearly combined using a weight called *preference* Abels et al. (2019); Mossalam et al. (2016); Barrett and Narayanan (2008); Yamaguchi et al. (2019); Lizotte et al. (2010); Hayes et al. (2022). We make the same assumption. In both MORL and robust RL, the transition dynamics model (and potentially uncertainty about it) is either explicitly or implicitly (via data) provided as input to the algorithm. The algorithm's goal is to then output a policy that is robust, or can be efficiently recomputed if preferences

change. In contrast, our aim here is to *learn a transition model* which will produce robust policies when optimized for different reward functions.

There are a few exceptions—(Wiering et al., 2014; Wan et al., 2021; Yamaguchi et al., 2019) propose MORL algorithms which learn a transition model, but do so (a) by only using $\{(S_t, A_t, S_{t+1})\}$ transitions to learn the model, and (b) by making strong assumptions on the state space (e.g. discrete) or the model dynamics (e.g. SIR model in epidemiology). Importantly, their model is learned *before* it is used for the MORL step (i.e. model learning does not take into account the policy from the MORL step). In contrast, our model is learned *along with* the MORL step: we consider which transition model will result in a good policy during the MORL step.

Transfer Learning across Rewards Under Fixed Dynamics. The setting of fixed MDP transition dynamics but different reward functions has also been addressed in the transfer learning literature. The key difference between the transfer learning literature and the multi-objective learning literature in which our work is centered is that our method is not designed to adapt to a new task with new data, but rather find one model which applies well to multiple tasks at once. Barreto et al. (2020) performs transfer learning in situations where only the reward function differs, by using successor features to decouple a policy's dynamics from expected rewards. Reinke and Alameda-Pineda (2021) relax some of the assumptions that rewards may be decomposed linearly into successor features for knowledge transfer. Unlike both of these, our work makes no assumptions about the form of the reward function. The idea of using successor features to express the reward function is complementary and can also be incorporated into our approach.

3. Preliminaries and Background

3.1. Notation

Markov Decision Process In RL, a Markov Decision Process (MDP) M is defined as $\mathcal{M} = (\mathcal{S}, \mathcal{A}, T_0, R, T_*, \gamma)$. $\mathcal{S}$ is a state space, $\mathcal{A}$ is an action space, T_0 is the starting-state distribution, $R(s, a)$ is a reward function, $T_*(s'|s, a)$ is a transition distribution function, and $\gamma \in [0, 1)$ is a discount factor. We assume a fixed starting state, although having a distribution over the starting states is straightforward.

The goal of the agent is to learn a policy $A_t = \pi^*(S_t)$ that maximises the expected return

$$J_{T_*,R}(\pi) = \mathop{\mathbb{E}}_{A_t \sim \pi} \left[\sum_{t=0}^{\infty} \gamma^t R(S_t, A_t) \right] \tag{1}$$

for transition function T_* and reward function R. The policy quality can also be measured by the Bellman optimality error,

$$L_{T_*,R}(\pi) = \sum_{s,a} \left| Q^{\pi}_{T_*,R}(s, a) - BQ^{\pi}_{T_*,R}(s, a) \right|, \tag{2}$$

where $Q^{\pi}_{T_*,R}(s, a) \triangleq \mathbb{E}_{\pi} \left[\sum_{t=0}^{\infty} \gamma^t R(S_t, A_t) | s_0 = s, a_0 = a \right]$ is the action-value function of π: and B is the Bellman optimality operator induced by the transition T_* and reward R on

action-value function Q:

$$BQ(s,a) \triangleq R(s,a) + \gamma \mathop{\mathbb{E}}_{T_*(s'|s,a)} \left[\max_{a'} Q(s',a') \right] \tag{3}$$

The optimal policy π^* can also be derived from $Q^*_{T_*,R}$. $Q^*_{T_*,R}$ is a fixed point of B, and can either be obtained by applying the iteration $Q \leftarrow BQ$ in Eqn 3 until convergence, or by minimizing the Bellman error in Eqn 2. Therefore, maximizing $J_{T_*,R}(\pi)$ is equivalent to minimizing $L_{T_*,R}(\pi)$.

Reward Preferences We assume a reward function that linearly interpolates between K basis reward functions R_1 to R_k according to,

$$R_w(s,a) = \sum_{k=1}^{K} w_k R_k(s,a); \quad \sum_{k=1}^{K} w_k = 1 \tag{4}$$

where w_k denotes a preference for a particular basis, and the R_k's are assumed as given. In practice, these reward bases are competing reward functions that the practitioner considers important. For example, in cancer treatment, they would promote the treatment's efficacy (R_1) and penalize the patient's side-effects (R_2).

Model-based RL Model-based RL (MBRL) algorithms learn a transition model T_θ of the environment and use this model to plan a policy. Here, $\theta \in \Theta$ are the parameters of the model which need to be estimated. MBRL methods allow improved sample-efficiency and generalizability (Deisenroth, 2011). Traditional methods Sutton (1991); Wiering et al. (2014) use maximum-likelihood estimation to estimate θ (MLE-MBRL), which is equivalent to minimizing the KL divergence between the transition model and the true dynamics:

$$\theta_{\text{MLE}} \leftarrow \arg\min_{\theta} \mathbb{KL}\left(T_* || T_\theta\right) \tag{5}$$

Since the MLE objective does not directly optimize for the objective of discounted returns, it can fail to find optimal policies when the model capacity is limited Nikishin et al. (2022); Farahmand (2018); Joseph et al. (2013). For example, an MLE model may "waste" its capacity on modeling an action that the optimal policy will never take, at the expense of differentiating between two near-optimal actions because it does not take into account the policies learned with the model.

Decision-focused Model-based RL (DF-MBRL). DF-MBRL considers the full computational graph of how the transition model T_θ affects the performance of the policy $\pi^*(\theta, R)$:

$$\theta \rightarrow T_\theta \rightarrow Q^*_{T_\theta,R} \rightarrow \pi^*(\theta,R) \rightarrow J_{T_*,R}(\pi^*(\theta,R)) \tag{6}$$

where $J_{T_*,R}$ expected return but we can also use the Bellman optimality error $L_{T_*,R}$. The policy $\pi^*(\theta,R)$ is the optimal policy for the transition model T_θ and reward function R, and is computed by maximizing $J_{T_\theta,R}(\pi)$. $J_{T_*,R}(\pi^*(\theta,R))$ depends on θ through its dependence on $\pi^*(\theta,R)$. We use the notation $J_{T_*,R}(\theta)$ or $J_{T_*,R}(\pi^*)$ to show dependence on θ or π^* respectively. DF-MBRL directly optimizes for the performance of the policy on the true transition T_*:

$$\theta_{\text{DF}} \leftarrow \arg\max_{\theta} J_{T_*,R}(\theta) \tag{7}$$

In settings where the model class of T_θ cannot represent T_*, the decision-focused model can outperform the maximum-likelihood model (Joseph et al., 2013; Farahmand et al., 2017). However, there can be several model parameters θ whose policy has high performance on the true environment (Grimm et al., 2020; Nikishin et al., 2022), i.e. the DF model is non-identifiable.

The main limitation of DF-MBRL is that the model is optimized for *one specific reward function*, R. While this is not an issue in settings where the reward function is not expected to change, this can be problematic in settings where the reward function is expected to change—or simply not known precisely at training time.

4. Problem Setting

The goal of our proposed approach is to learn a *simple decision-focused model* of the MDP dynamics that produces high-performing policies for rewards encountered during *both* learning and deployment phases. For example, a model for treating patients with a condition in the ICU may need to be deployed in a setting where resources may be more constrained and clinicians must prioritise patient health *as well as minimising costs*. Alternatively, in cancer treatment, different patients during learning and deployment phases may be more susceptible to developing adverse side effects from the treatment and would therefore need lower dosages (Lizotte et al., 2010).A robust model would perform well across a range of changes in these preferences.

Learning phase. In the *learning phase*, we are given access to the true transition function T_* (in the form of a simulator) and the reward function $R_{\bar{w}}$, where $\bar{w}$ is the learning-phase reward preference. We are also provided a *deployment-phase* reward preference distribution $P(w)$, where w is the deployment-phase reward preference. During the learning phase, we can simulate trajectories from the simulator without any restrictions. At the end of this phase, we must build a model of the simulator that is simple enough, but also allows us to plan high-performing policies for different reward preferences.

Deployment phase. During the *deployment phase*, we are no longer given access to the true transition function. Note that while the deployment-phase reward preference w is unknown during the learning phase, we know the distribution $P(w)$ from which w comes from. In addition to doing well on $R_{\bar{w}}$, we should be able to re-plan high-quality policies on the deployment-phase reward functions.

$\bar{w}$ encodes the preferences for a known patient population, and $P(w)$ would correspond to the set of preferences the doctor wants to support.

Why simple models? In healthcare applications, the true environment dynamics are very complex and expensive to access. For example, organs-on-chips models of a lung alveolus can faithfully simulate lung cancer dynamics but are expensive (Ingber, 2022; Hassell et al., 2017). For this reason, simple models are learned to understand disease dynamics and to simulate data for reasoning and policy learning. While complex computational models can be used to simulate highly accurate data, they can be expensive to run and their access can be siloed. Simple models are desirable for reduced computational complexity, improved interpretability (Doshi-Velez and Kim, 2017), and to *only* capture the dynamics relevant for the problem.

Choosing $P(w)$. Although knowledge of $P(w)$ might seem to be a strong assumption, in real-world settings it is often possible to define a $P(w)$ using reasonable boundary conditions. For example, in many healthcare applications, patients can express their preferred trade-off between the aggressiveness of the treatment and its side-effects. A domain expert will both know what kinds of trade-offs are common in their domain, as well as what is the common span of preferences along these trade-offs. Access to this type of knowledge motivates our use of a uniform distribution over a given range of w, encoding the idea that domain experts can often easily provide reasonable ranges of preferences, but not probabilities over the preferences. Our goal then becomes to be robust over the entire range of reasonable preferences.

5. Proposed Framework: Robust Decision-Focused Model-based RL

While DF-MBRL can learn a model that performs well on the true environment, it is tied to a specific reward function that was used during training. To alleviate this drawback of DF-MBRL, we leverage the non-identifiability of DF solutions—among the many DF solutions on the learning-phase reward function, we choose the one that would perform well on multiple possible deployment-phase reward functions.

We formalize this notion in the following Robust Decision-Focused (RDF) MBRL objective:

$$\begin{aligned} \theta_{\text{RDF}} \leftarrow \arg\min_{\theta} & \mathop{\mathbb{E}}_{P(w)} \left[J_{T_*, R_w}(\theta) \right] \\ \text{s.t. } & J_{T_*, R_{\bar{w}}}(\pi^*(\theta, R_{\bar{w}})) \geq \delta \end{aligned} \tag{8}$$

which optimizes the model parameters θ to have high performance on the (yet unknown) deployment reward function R_w, while simultaneously achieving high performance on the learning-phase reward function $R_{\bar{w}}$. Below we first provide a theoretical analysis of this objective and then describe our optimization approach.

5.1. Theoretical Analysis

We theoretically characterize the quality of policies achieved by RDF algorithm, and show that our RDF objective can achieve better policies than DF-MBRL in the deployment phase. Hence, given the same representational capacity, RDF will learn a model that better approximates the optimal Q function across the range of possible deployment rewards. We provide the proof in the supplement.

Theorem 1 *Let $R_{\bar{w}}$ be the learning-phase reward function with preference $\bar{w}$, and R_w be the reward function with an arbitrary preference w. Let Q^*_w be the optimal action-value function for the true MDP for reward function R_w. Let $B_w, \hat{B}^{DF}_w, \hat{B}^{RDF}_w$ denote the Bellman optimality operators under the true dynamics, DF model, and RDF model respectively.*

Assume $\hat{Q}^{DF}_w$ and $\hat{Q}^{RDF}_w$ are fixed points under $\hat{B}^{DF}_w$ and $\hat{B}^{RDF}_w$ respectively. Further assume that the reward function is bounded, $R_w(s,a) \in [0, r_{max}] \forall s, a, w$.

DF Case. *Consider a DF model trained on $R_{\bar{w}}$. If the Bellman operator induced by the DF model achieves the error* $\sup_{s,a} |B_{\bar{w}}\hat{Q}^{DF}_{\bar{w}}(s,a) - \hat{B}^{DF}_{\bar{w}}\hat{Q}^{DF}_{\bar{w}}(s,a)| = \epsilon^{DF}_{\bar{w}}$, *then*

$$Q^*_w(s,a) - \hat{Q}^{DF}_w(s,a) \leq \frac{\epsilon^{DF}_w}{(1-\gamma)} \qquad \text{for } w = \bar{w} \tag{9}$$

$$Q^*_w(s,a) - \hat{Q}^{DF}_w(s,a) \leq \gamma \frac{r_{max}}{(1-\gamma)^2} \qquad \forall w \neq \bar{w} \tag{10}$$

RDF Case. *Consider the RDF model trained with learning-phase preference $\bar{w}$ and deployment-phase reward preference distribution $P(w)$. For a $w \in P(w)$, if the Bellman operator $\hat{B}^{RDF}_w$ induced by the RDF model achieves the error* $\sup_{s,a} |B_w\hat{Q}^{RDF}_w(s,a) - \hat{B}^{RDF}_w\hat{Q}^{RDF}_w(s,a)| = \epsilon^{RDF}_w$, *then*

$$Q^*_w(s,a) - \hat{Q}^{RDF}_w(s,a) \leq \frac{\epsilon^{RDF}_w}{(1-\gamma)} \tag{11}$$

For, $w \neq \bar{w}$, the RDF bound is tighter since we explicitly optimize ϵ^{RDF}_w whereas $\gamma \frac{r_{max}}{(1-\gamma)^2}$ is constant.

Empirical validation of Theorem. We empirically validate our theorem in the left panel of Figure 5.1 using a simulated MDP with twenty states and two actions (we describe the MDP and provide the simulation code in the supplement). We plot the suboptimality gaps $\max_{(s,a)} |Q^*_w(s,a) - \hat{Q}^{DF}_w(s,a)|$ and $\max_{(s,a)} |Q^*_w(s,a) - \hat{Q}^{RDF}_w(s,a)|$ for learning-phase ($\bar{w}$) and deployment-phase (w) preferences. We observe that the RDF bound (red dashed line) is tighter than the DF bound (blue dashed line) for $w \neq \bar{w}$. *More importantly*, there exist w values for which the RDF bound is tighter than the observed DF suboptimality gap (blue solid line is above the red dashed line in Figure 5.1 (Left)). For these w values, our bound guarantees that RDF model's suboptimality gap will be lower than the DF model's gap. This is possible because the RDF objective included these w values.

5.2. Optimizing of RDF objective

Since optimizing the constrained form of RDF objective in Eqn 8 can be challenging, we rewrite it using a Lagrange multiplier $\lambda \geq 0$:

$$J^{\text{RDF}}_{T_*}(\theta, \lambda) = \mathop{\mathbb{E}}_{P(w)} [J_{T_*,R_w}(\theta)] + \lambda(J_{T_*,R_{\bar{w}}}(\theta) - \delta) \tag{12}$$

The limit $\lambda \to \infty$ recovers the DF objective, while setting $\lambda = 0$ ignores performance on the learning-phase reward and focuses on overall robustness with respect to deployment-phase reward distribution, $P(w)$.

Evaluating the objective. As discussed in Sec 4, a uniform distribution is a reasonable assumption for $P(w)$ in real-world settings. We approximate the expectation in Eqn 12 using the trapezoidal rule Abramowitz and Stegun (1968). Specifically, we construct $\mathcal{W}$, a set of uniformly-spaced w values in the support of P and use it to approximate the expectation:

$$\mathop{\mathbb{E}}_{w \sim P(w)} [J_{T_*,R_w}(\theta)] \approx \frac{1}{|\mathcal{W}|} \sum_{w \in \mathcal{W}} J_{T_*,R_w}(\theta) \tag{13}$$

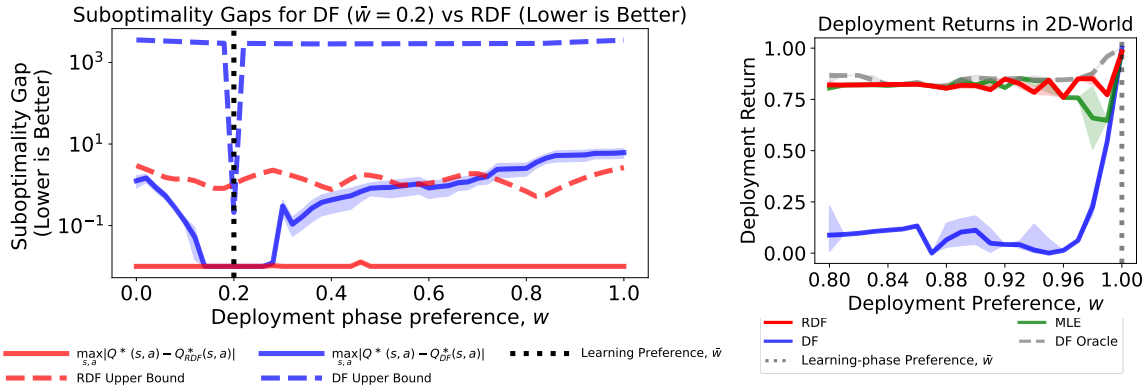

Figure 2: **Left:** Theoretical bounds on the suboptimality gap of RDF models (red dashed line) confirm that RDF models can achieve a better (i.e. lower) suboptimality gap than DF models (blue solid line) away from the learning-phase reward preference $\bar{w}$. Here, the RDF bound guarantees better-than-DF performance for $w > 0.8$. Learning-phase preference $\bar{w}$ is 0.2 and $P(w)$ is uniform on $[0, 1]$. **Right:** RDF outperforms DF and MLE models by achieving high returns across the deployment preferences in the 2D-World Environment.

When w is high dimensional, we can use a more appropriate sampling method, and if $P(w)$ is known, we can use importance weights to approximate the expectation more accurately. The choice of the sampling method is orthogonal to the RDF objective, and we proceed with a uniform grid for simplicity.

Computing the gradient of the RDF objective. We employ implicit differentiation to compute gradients through the policy learning step (Nikishin et al., 2022), and use the chain rule to compute the gradient of the RDF objective:

$$\frac{\partial J_{T_*,R}(\theta)}{\partial \theta} = -\underbrace{\frac{\partial J_{T_*,R}(\pi^*)}{\partial \pi}}_{\text{Grad Bellman}} \cdot \underbrace{\left[\frac{\partial^2 J_{T_\theta,R}(\pi^*)}{\partial \pi^2}\right]^{-1} \frac{\partial^2 J_{T_\theta,R}(\pi^*)}{\partial \pi \partial \theta}}_{\text{Implicit Grad of } \pi^* \text{ w.r.t } \theta} \tag{14}$$

We can then estimate the gradient of the RDF objective as:

$$\frac{\partial J^{\text{RDF}}_{T_*}(\theta, \lambda)}{\partial \theta} \approx \frac{1}{|\mathcal{W}|} \sum_{w \in \mathcal{W}} \frac{\partial J_{T_*,R_w}(\theta)}{\partial \theta} + \lambda \frac{\partial J_{T_*,R_{\bar{w}}}(\theta)}{\partial \theta} \tag{15}$$

Choosing a policy planner. Depending on the model parameterization, we can use an appropriate planning algorithm to learn the policy. The choice of the algorithm is orthogonal to the RDF objective, with the only requirement being that the algorithm can compute the gradient of the policy $\pi^*(\theta)$ with respect to the model parameters θ. However, specific simple model classes support fast planning methods that can be used, e.g. Value Iteration (VI) for tabular transition matrix with discrete states, and Linear Quadratic Regulator (LQR) for linear transition dynamics with continuous states. For example, in the cancer simulator we describe in Sec 7.1, an LQR planner returned an optimal policy in less than a

second because we used a linear transition model, while a Deep Q Network (DQN) (Mnih et al., 2015) takes hours to train.

Algorithm 1: General Robust Decision-Focused RL Algorithm

Input: Initial model parameters θ, learning reward preference $\bar{w}$, range for $P(w)$, reward basis functions $R_1, \ldots, R_k$

Create uniform grid $\mathcal{W}$ on the range of $P(w)$
Initialize Q-function parameters $\phi_{\bar{w}}$, $\{\phi_w : w \in \mathcal{W}\}$
repeat
  foreach $w \in \{\bar{w}\} \cup \mathcal{W}$ **do**
    Using model T_θ and reward function R_w, update action-value function $Q^*_{T_\theta, R_w}(\phi_w)$
    Compute policy $\pi^*(\theta, R_w)$ using the action-value function
    Compute return on true model, $J_{T_*, R_w}(\pi^*(\theta, R_w))$
  end
  Update θ using the gradients computed using Eqn 15
until *the Bellman error converges*

6. Experiments

We demonstrate the use of the RDF algorithm on a wide variety of simple model classes, such as tabular representation and linear dynamics. We first evaluate our RDF algorithm on a synthetic environments, where we can control the complexity of the environment and the reward preferences. Next, we evaluate our algorithm on a cancer simulator, and finally on a real dataset of hypotensive patients in the ICU. When assuming access to the simulator, we can simulate from the true simulators only during the learning phase and must learn a *simple* model that can be used to plan policies during the deployment phase.

Baselines. We compare the performance of our RDF approach to methods that are relevant to our setting. That is, (a) the baseline must output a simple model of transition dynamics, and (b) the model learned using the baseline must transfer to multiple reward preferences. As identified in the Related Works (Sec 2), existing methods satisfy one of these requirements, but not both. We do not compare with model-based MORL methods (Wiering et al., 2014; Wan et al., 2021; Yamaguchi et al., 2019) since they do not fit our setting for two reasons: (a) they only perform a single planning step and only *evaluate* the planned policy on different preferences, and (b) they do not consider the learning of restricted models informed by the MORL step (they just do MLE/Bayesian model learning). To consider the best one can do with the constraint of a restricted model class, we train a DF-Oracle model that has access to the true transition dynamics during the deployment phase. Therefore, we compare RDF against DF and MLE (trained once on the training phase setting), as well as DF-Oracle (trained for each deployment phase setting).

Metrics. We evaluate the performance of RDF, DF, MLE, and DF-Oracle on the average deployment-phase return, denoted as J_{avg}, and the learning-phase return, denoted as $J_{\bar{w}}$.

We scale all returns to $[0, 1]$ by using the maximum and minimum observed returns for a given reward preference in the case of the simulators, and normalize the returns by the behavior policy's return for the hypotension dataset. This ensures that different weight preferences (with different return ranges) are comparable to each other, and J_{avg} is not dominated by a single w value. We report the means and standard errors across 10 random seeds.

6.1. Results on Synthetic Experiment

6.2. 2D-World Environment

We construct a navigational task with two-dimensional continuous states $\in R^2_+$ and two-dimensional actions in $\{[0, 1]^\top, [1, 0]^\top\}$. The agent starts at state $[0, 0]^\top$ and the episode ends if any of the states crosses value 25. The transition model is given by $\mathbf{s}' \leftarrow \mathbf{s} + \theta \odot \mathbf{a}$, where the true model's parameters are $\theta = \theta^* = [1.5, 5]^\top$. For a physical interpretation, the θ parameters can be thought of as slippage coefficients for the two state dimensions. If θ is higher, the agent will slip more for an action it takes in that dimension.

Reward function. The first reward basis function R_1 only depends on the first state dimension S_1. The second reward basis function R_2 is only a function of S_2.

$$R_1(S) = \begin{cases} 5, & \text{if } S_1 \leq 2.5 \\ -1, & \text{o/w.} \end{cases}; \qquad R_2(S) = \begin{cases} \frac{20}{17} S_2^2 - 1, & \text{if } S_2 \leq 13 \\ -201, & \text{o/w.} \end{cases}$$

The first reward function incentivizes the agent to stay in the region $S_1 \leq 2.5$, so knowing the correct value of S_1 is important to learn a good policy. The second reward's value increases with S_2 until $S_2 = 13$ (a cliff), after which the agent reaches a region of large negative rewards. Without knowing the correct value of S_2 it will end up in, the agent would find it hard to learn a good policy.

Transition model learning. We restrict the model class to $\theta = \theta_c[1, 1]^\top$, where θ_c is a scalar parameter, which corresponds to a subspace of the full parameter space not containing the true model parameters θ^*. The environment is challenging because the model forces the agent to either be too aggressive or too conservative in at least one of the state dimensions. For example, if it believes the slippage in the second dimension is smaller than it actually is (i.e. $\theta_c < \theta^*_2$), it will take aggressive steps in the second dimension and will fall off the cliff at $S_2 = 13$.

Since the learning-phase reward only depends on the first state dimension S_1, the DF model θ_{DF} learns a value close to θ^*_1 and can ignore the second state dimension S_2 without losing performance on the learning-phase reward function. The RDF model θ_{RDF} should reasonably estimate slippage in both the first and second state dimensions to achieve high performance. The MLE model θ_{MLE} learns the value $(\theta^*_1 + \theta^*_2)/2$ for a uniform random policy.

We set the learning-phase reward preference to $\bar{w} = 1$ (i.e. $R_w = R_1$), and the deployment-phase reward preferences range to $w \in [0.8, 1]$. We create a grid of 50 θ values in the range $[0, 6]$ and compute the optimal policy's return for θ. We use Fitted Q-Iteration (Ernst et al., 2005) to learn a deterministic policy for any θ.

6.3. Conclusions from synthetic domain

RDF models are more robust to reward function changes. Right panel of Fig 5.1 demonstrate the robustness of the RDF models when subjected to reward preferences away from their training settings. Unlike the DF and MLE solutions, RDF achieves near-optimal performance for the model class in both domains, with almost no degradation in learning-phase return. Performance of the DF model degrades significantly away from the learning-phase reward preference that it was trained on. MLE tends to transfer better than DF in both cases—but not as well as our RDF approach.

RDF offers significant advantages in optimizing trade-offs between objectives. The RDF agents consistently reached the goal faster than DF and MLE agents, even as fuel costs increased, all while maintaining reduced acceleration levels. This suggests that RDF transition models capture the effect of all actions much better than DF and MLE models. When transferring to a preference characterized by higher fuel costs, the DF models failed to even reach the goal.

7. Healthcare simulators

Now we apply our RDF approach to two more complex environments relevant to our intended application: healthcare simulators.

7.1. Healthcare Simulator: Cancer Treatment

The focus of this cancer simulator is on optimizing dosing strategies to reduce mean tumor diameters (MTDs) for patients undergoing chemotherapy for the drug temozolomide (TMZ) (Yauney and Shah, 2018). The domain utilizes a tumor growth inhibition (TGI) model that captures the growth kinetics of diffuse low-grade gliomas (LGG) during and after chemo- and radiotherapy (CRT) of patients (Ribba et al., 2012). The 5-dimensional states $S_t = (M_t^1, M_t^2, M_t^3, C_t, t)$, consisting of the patient's mean tumor diameters, drug concentration, and current time-step to ensure the Markovian assumption is met. The actions are discrete, in $\{0, 1\}$, and correspond to whether the drug is administered or not.

Reward function The paper introducing this simulator (Yauney and Shah, 2018) included a reward function that can be interpreted as having two components: one that promotes a reduction in overall tumor size, and another that penalizes side-effects from using high concentrations of drugs. That is,

$$R_1(M_t, A_t, M_{t+1}) = \begin{cases} c_1(M_t - M_{t+1}) + (M_0 - M_T), & \text{if } t = T - 1. \\ c_1(M_t - M_{t+1}), & \text{otherwise.} \end{cases}$$
$$R_1(S_t, A_t, S_{t+1}) = -c_2 C_{t+1} \tag{16}$$

where $M_t = M_t^1 + M_t^2 + M_t^3$ is the total mean tumor diameter at time t. The parameters c_1 and c_2 are constants set to .1 and .5 in the original paper, after observing that these values led to sufficient MTD reduction in the patient population. Notably, these parameters do not take into account the range of preferences that a clinician may want to balance in terms of MTD reduction and side-effects. This is an important consideration in the real-world settings (Lizotte et al., 2010).

Transition model learning The environment dynamics follow TGI model, which is a system of ordinary differential equations (ODEs) that describe the evolution of the tumor size over time. We learn a linear model class to map (S_t, A_t) to S_{t+1} and use the learned model to plan policies for different reward preferences. The linear model class and linear reward function allow us to use a Linear Quadratic Regulator (LQR) to plan policies.

We set the learning-phase preference $\bar{w}$ to be 0.75, and the deployment phase preference range to $w \in [0.5, 1]$. We employed a Linear Quadratic Regulator (LQR), which leverages the advantages of the linear model class and linear reward function to efficiently plan policies. We set constraint values δ in the RDF objective (Eq 1) to $\delta \in \{0.95, 0.98, 1\}$ times the DF return. We did not try lower δ values because the unconstrained RDF model gives a high enough learning-phase return of 0.95 times the DF return.

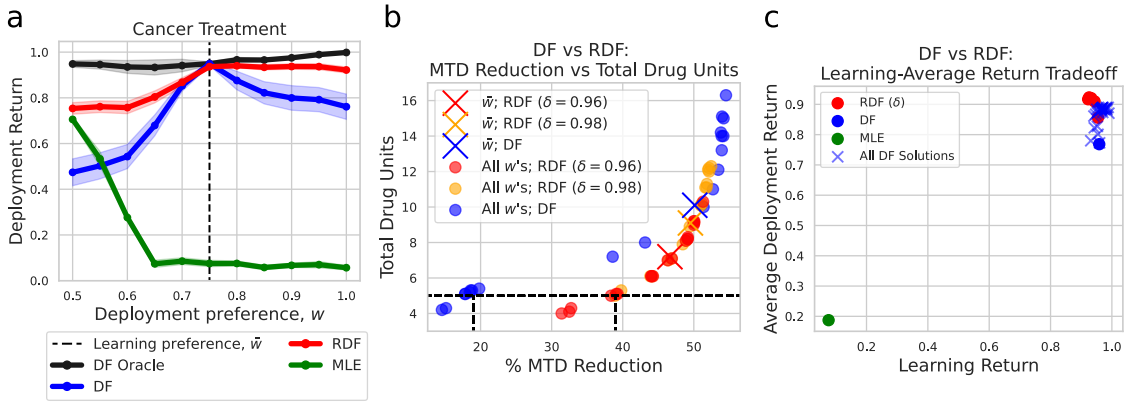

Figure 3: Results for Cancer Simulator: (a): RDF achieves superior average-case performance and covers DF's performance across preferences for both domains. (b): The trade-off between Cancer environment's objectives illustrates how RDF models (red and orange) achieve reductions in tumor diameter, even in low-drug-dose scenarios (they are further right of DF models (blue)). A model achieving a perfect trade-off would be in the bottom-right corner of the plot. (c): The learning-vs-average return plot shows how RDF models balance learning and average performance: RDF loses very little in learning return, and gains a lot in average return.

7.2. Conclusions from Cancer Simulator

DF models can be non-identifiable. Fig 3(c) shows that multiple DF solutions (marked by 'x' markers) can have very different performance on deployment-time preferences.. This provides evidence that transfer in DF can be problematic without additional specifications, which RDF provides in form of the average performance objective.

RDF models consistently learn a higher-return policy when evaluated on deployment phase rewards compared to DF and MLE (Fig 3(a)). Note that the

constraint on the learning return (δ) enables RDF to trade-off learning-phase and average return (Fig 3(c)). Typically, the lower the constraint δ, the higher the average-case performance we can achieve. However, even for the same learning-phase return as DF, RDF model achieves a higher average return.

RDF offers significant advantages in optimizing trade-offs between clinical objectives (Fig 3(b)) These trade-offs can be particularly relevant when considering individual patient preferences and tolerances. For patients with low tolerance to side-effects (i.e. treatment constrained to 4-6 doses), RDF demonstrates MTD reduction of approximately 35%, well above the 20% reduction by DF. Even for patients with high tolerance to side-effects, RDF remains competitive, achieving an MTD reduction similar to DF, at 50%. These findings underscore the versatility and effectiveness of RDF in optimizing dosing regimens based on patient-specific needs and preferences. We provide detailed trajectories generated by both RDF and DF models in the supplement.

RDF models allow learning dynamics using interpretable model classes while still giving good policies Choosing a linear model class for transition dynamics allows us to inspect the $(S_t, A_t) \rightarrow S_{t+1}$ relationships that the model has learned. Table 1(a) presents the linear model's coefficients for predicting the next state's concentration using the current state and action. We see that the MLE has correctly learned the relationship of concentration (C_{t+1}) with the previous concentration (C_t) and the action of administering a drug ($A_t = 1$). It has also learned that the next C_{t+1} is independent of the tumor dimensions (M_t^1, M_t^2, M_t^2) and the time-step t. On the other hand, both DF and RDF models have learned non-zero effects of tumor dimensions and time-step on C_{t+1}. Notably, for the RDF model, the effect of C_t and t on C_{t+1} is *more* than what the data suggests, meaning that the RDF model is biased to over-estimate the concentration when the existing drug concentration is high or the treatment is in the later stages. Therefore the optimal policy under the RDF model will hold off giving the drug if there is a penalty for having a high drug concentration, which is a desirable outcome for smaller w values. The relationship that the RDF model has learned, while not accurate, enables us to find good policy across a range of reward preferences (w) between tumor reduction and drug penalty. The DF model learns the opposite effects and thus tends to be more aggressive in its drug dosage. Note that there is still always a trade-off when the model class is restricted: the MLE model minimized its prediction error at the cost of having a poor-quality policy (Fig 3), and DF/RDF models maximized the quality of their policies at the cost of having higher prediction error (Table 1(b)).

8. Real ICU Data - MIMIC IV

8.1. Data

Cohort Selection We use EHR data from the MIMIC-IV care database, which contains deidentified clinical data of patients admitted to the Beth Israel Deaconess Medical Center ICU unit (Johnson et al., 2023) We filter the data to include only patients who were admitted to the ICU for a total of at least 24 hours, were aged 18-80 years, and had at least 7 mean arterial pressure (MAP) measurements of less than 65 mmHg within the first 72 hours of their ICU stay. We set the 72-hour limit because the care for hypotension during later

(a) Linear Model Coefficients for Concentration (C_{t+1}) as Outcome

State/Action	MLE	DF	RDF
Intercept	-0.00	-0.01	-0.01
Concentration (C_t)	**0.76**	**0.64**	**0.83**
Proliferative Tissue (M_t^1)	0.00	-0.05	-0.06
Non-proliferative Quiescent Tissue (M_t^2)	0.00	0.02	-0.03
Damaged Quiescent Cells (M_t^3)	0.00	-0.10	-0.01
Time Step (t)	**0.00**	**-0.10**	**0.13**
Action (A_t)	0.76	0.60	0.69

(b) RMSE for all Outcomes

Outcome	MLE	DF	RDF
C_t	0.142	3.936	1.699
M_t^1	0.873	3.476	2.358
M_t^2	6.804	7.527	7.412
M_t^3	5.193	5.917	5.298

Table 1: Cancer Simulator: (a) Coefficients for current state and actions for the linear model predicting the Concentration of the next state. MLE learns a zero effect of time step on concentration, whereas DF and RDF models learn nonzero and *opposite* effects. (b) RMSE of predicting the next state given the current state and action.

periods of ICU stay may be quite different. After filtering, we were left with 5939 stay IDs. We randomly split this dataset $\mathcal{D}$ into a training set $\mathcal{D}_{\text{train}}$ of 2000 stay IDs, a validation set $\mathcal{D}_{\text{val}}$ of 1500 stay IDs, and a test set $\mathcal{D}_{\text{test}}$ of 2439 stay IDs.

Clinical Variables Given the hypotension-related cohort selection, we select the following variables for our model: mean arterial pressure (MAP), Glasgow Coma Scale (GCS), PaO2/FiO2 ratio, creatinine, vasopressor use, and fluid bolus use. Among these, vasopressor use, and fluid bolus use are our action variables, and the rest inform the patient state. We impute missing values by first forward filling, then backward filling, and finally filling with the median value of the variable.

8.2. MDP Details

State- and action-space construction We discretize the continuous variables into bins that correspond to severity levels. The state space corresponds to all possible combinations of these severity levels. The discretization is done as follows:

$$\text{O2} = \begin{cases} 0 & \text{if } \frac{\text{PaO2}}{\text{FiO2}} \geq 200 \\ 1 & \text{if } \frac{\text{PaO2}}{\text{FiO2}} < 200 \end{cases} \quad \text{MAP} = \begin{cases} 0 & \text{if MAP} \geq 70 \\ 1 & \text{if MAP} < 70 \end{cases}$$

$$\text{GCS} = \begin{cases} 0 & \text{if GCS} \geq 12 \\ 1 & \text{if GCS} < 12 \end{cases} \quad \text{Creat} = \begin{cases} 0 & \text{if Creatinine} \leq 1.9 \\ 1 & \text{if } 1.9 < \text{Creatinine} \leq 4.9 \\ 2 & \text{if Creatinine} > 4.9 \end{cases}$$

The state S_t is then defined as the tuple $(\text{O2}_t, \text{MAP}_t, \text{GCS}_t, \text{Creat}_t)$. The action space corresponds to all possible combinations of vasopressor and fluid bolus use, each of which can be either 0 or 1. There are thus four possible actions: $(0,0), (0,1), (1,0), (1,1)$.

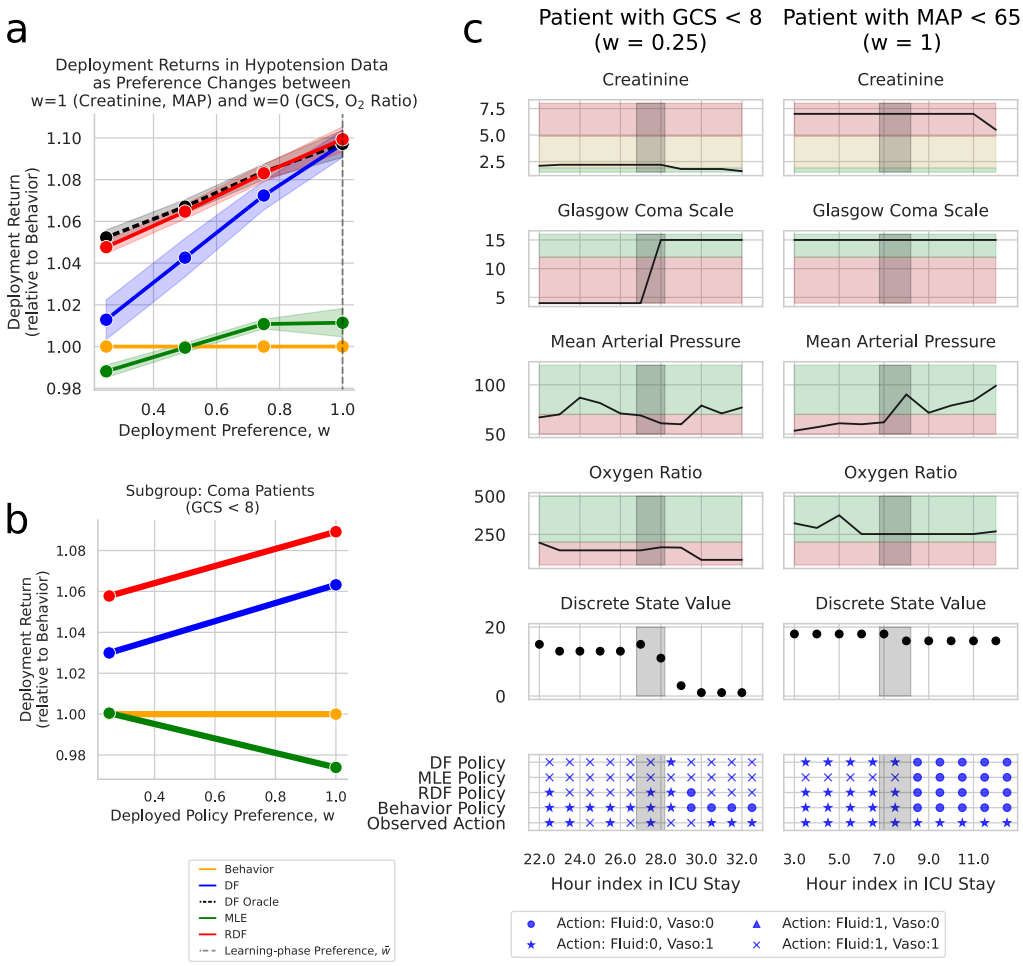

Figure 4: Performance comparison of RDF vs baselines on MIMIC Hypotension data. (a) RDF is better able to achieve higher deployment returns across a range of different deployment preferences across hypotensive patients. (b) RDF consistently outperforms baselines in terms of deployment returns for both acutely hypotensive and comatose patients. For acutely hypotensive patients, DF and RDF baselines prioritise MAP to achieve higher returns than MLE. For comatose patients, both these solutions prioritise improving GCS to achieve higher returns than MLE. RDF manages performance over varying preferences more efficiently than DF. (c) For comatose patients, the RDF policy matches the behaviour policy of the clinician that suggests using only vasopressors which increases the GCS score in the window of interest. For acutely hypotensive patients, DF and RDF policies both suggest using only vasopressors which improves the mean arterial pressure in the window of interest.

Reward function We define the reward functions as follows:

$$R_1(S_t) = 60 - 10\,(\mathrm{MAP}_t + \mathrm{Creat}_t)$$
$$R_2(S_t) = 60 - 10\,(\mathrm{O2}_t + \mathrm{GCS}_t)$$
$$R_w(S_t) = wR_1(S_t) + (1-w)R_2(S_t)$$

The reward function R_1 incentivizes MAP and Creatinine levels within the desired range, and are proxied as indicators of hypotension and kidney function. The reward function R_2 incentivizes O2 and GCS levels within the desired range, and are proxied as indicators of consciousness and sedation during the ICU stay. A reward preference of $w = 0$ chooses R_2 and $w = 1$ chooses R_1. We visualize these reward functions for different preferences w in Fig 12 in the supplement.

Transition model learning Since the state and action spaces are discrete, we learn a tabular transition model. This amounts to learning the transition probabilities $P(S_{t+1}|S_t, A_t)$ for all possible state-action pairs. For the MLE model, we estimate these probabilities by counting the number of times each transition occurs in the training data and normalizing by the total number of transitions from state S_t under action A_t (we also add a pseudocount of 0.01 to avoid zero probabilities).

Policy Learning The tabular transition model allows us to use Value Iteration (by applying Equation 3) to learn the optimal policy. To ensure that the learned policies don't take unsafe actions (i.e. actions not observed in the training data), we force the policies to have zero probability on actions that have less than 3% probability in the training data. For the DF and RDF models, we use consistent weighted per-decision importance sampling (CWPDIS) (Metelli et al., 2020) to estimate the return of a policy $\pi(\theta)$ using the training data. The CWPDIS estimator is given by:

$$\hat{J}^{\text{CWPDIS}}(\pi) = \sum_{t=1}^{T} \gamma^t \frac{\sum_{n \in \mathcal{D}} r_{nt} \rho_{nt}(\pi)}{\sum_{n \in \mathcal{D}} \rho_{nt}(\pi)}, \qquad \rho_{nt}(\pi) = \prod_{k=1}^{t} \frac{\pi(a_{nk}|s_{nk})}{\mu(a_{nk}|s_{nk})},$$

where r_{nt} is the reward at time t in trajectory n, $\rho_{nt}(\pi)$ is the called the importance weight, $\pi(\theta)$ is the policy being evaluated, and μ is the behavior policy (estimated from the data).

8.3. Conclusions from ICU Application

Learning a tabular transition matrix is useful for interpreting what the model learned about the dynamics Fig 13 shows the transition dynamics learned by the RDF model (DF and MLE models in the appendix). Choosing a tabular class of transition model helps the practitioner inspect the transition dynamics learned by the model. Such an inspection is useful as it allows model critique and revision if necessary. Complex model classes (e.g. NNs) can achieve good predictive performance and decision quality, but their black box nature precludes this ability to inspect the model. The RDF approach allowed us to use the simpler model classes, while still achieving good decision quality.

RDF model achieves the best transfer to different reward functions As seen in the earlier results, the RDF model achieves superior performance both on and away from the learning-phase reward preferences, as demonstrated by Fig 4(a). There is a significant drop in the performance of the DF transition model, providing more evidence that DF models are prone to severe overfitting to the reward function they were trained with ($w = 1$ in this case). This suggests that DF model does worse for patients who have a GCS less than 12 or O2 ratio less than 200. The MLE model is less sensitive to the reward preference but also has low performance overall.

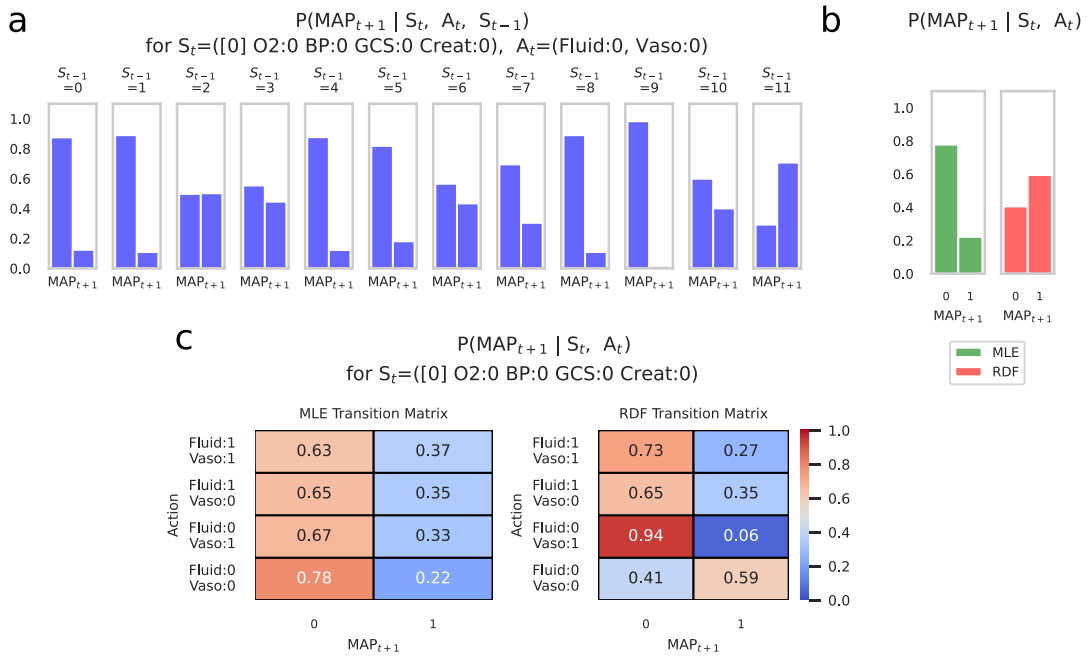

Figure 5: Non-Markovian Transition Dynamics: (a) The dataset suggests that knowing the present state $S_t = 0$ and action $A_t = 0$ are not sufficient to determine the next state's MAP_{t+1} distribution (i.e. transitions are non-Markovian) since this distribution changes as S_{t-1} is varied. (b) MLE and RDF learn different transition dynamics under the constraint that the transition model is Markovian. (c) Learned MLE and RDF transition probabilities for the transition $\text{MAP}_{t+1}|S_t = 0, A_t$ for all actions.

RDF can adjust for non-Markovian dynamics by learning Markovian transition dynamics useful for decision-making Markovian assumption means that the future outcome of a process is independent of the past given the present. In the context of the MIMIC dataset, this would mean that knowledge of the present present state S_t and action A_t is sufficient to predict the next state S_{t+1}, and that the past states and actions $S_{1:t-1}$ and $A_{1:t-1}$ are not necessary for this prediction. This is clearly a simplifying assumption that does not hold in general. For example, in our dataset, the future MAP severity (MAP_{t+1}) of a patient is likely to depend on the past MAP severity ($\text{MAP}_{1:t}$) of the patient. Indeed, we observe that the distribution of MAP_{t+1} given $S_t = 0$ (healthy state) and $A_t = 0$ (no action) depends on whether the patient was in a healthy state ($S_{t-1} = 0$) or a very sick state ($S_{t-1} = 0$) in the previous time step (Fig 5). Nevertheless, the Markovian assumption on states is a common simplification in reinforcement learning (Futoma et al., 2020) and is often used in practice to reduce the complexity of the learning problem. Knowing that we are working with a simplification, we would hope to learn a model which can still learn to make good decisions. The RDF model learns the most useful dynamics for getting a high-return policy (Fig 5(b and c). This means it chooses to learn transitions which are opposite of what the average (i.e. MLE) would suggest (Fig 5(b and c)).

RDF can allow high-quality personalized policies for different patient cohorts A consequence of the RDF model learning is that for different patient cohorts, we can learn personalized policies that are tailored to the requirements of that cohort. Consider, for example, the patient cohort with coma (GCS value less than 8). For this cohort, the reward preference of $w = 0.25$ is more appropriate than the learning-phase reward preference of $w = 1.0$ since it incentivizes GCS values above 12. Fig 4(b) shows the superior transfer performance of the RDF model for this cohort. In the left panel of Fig 4(c), we inspect the trajectory of a patient in the coma cohort. We see that the RDF policy (for $w = 0.25$) chooses actions that match the expert policy (behavior) at the point of transition to a healthy GCS state. We see the same behavior in the right panel of Fig 4(c) for a patient in the non-coma cohort (but high-creatinine cohort).

9. Discussion and Limitations

In this paper, we introduced the Robust Decision-focused Model-based RL framework (RDF) as a novel approach for learning transition models in settings with varying reward preferences. Our RDF framework enabled us to learn transition models in settings with varying reward preferences.

We assumed the existence of a model class that is both interpretable and useful for learning a good decision policy. However, this assumption is often valid in clinical settings: the elements important for making decisions frequently end up being simple. Another potential risk is that we may learn incorrect relationships in the data if there are missing features or rewards. However, avoiding this limitation is possible with the help of collaboration with clinicians, and we expect our method to be especially useful in settings where the clinician can provide the relevant features and the range of rewards relevant to the decision-making task. We prioritize inspectability of the model precisely to enable a clinical expert to identify associations that do not make sense due to missing features.

From a technical perspective, an important aspect was addressing the time and space complexity requirements of the RDF framework. We demonstrated that our RDF framework can use a number of different policy optimizers. Settings like the MIMIC Hypotension dataset were small enough such that we could apply Value Iteration to learn the optimal action-value function at each planning step. In other settings, such as the cancer simulator, we combined our RDF framework with a Linear Quadratic Regulator (LQR) planner to make that planning step more efficient. This is an example of a situation in which using a restricted model class (linear) created computational advantages in addition to being interpretable.

To scale the RDF framework to more reward bases and more complex environments, we can use more sophisticated policy planners, such as a Deep Q-Network (DQN) or a Proximal Policy Optimisation (PPO) planner. For these methods, learning an optimal policy at each planning step is computationally prohibitive. However, we can leverage past work that has shown that these interleaving $Q-$ and $\theta-$ learning steps makes it possible to achieve convergence in decision-focused learning with neural-network-based policy optimizers (Nikishin et al., 2022). To mitigate the computational and space complexity of storing the $Q-$values for each reward preference, we can use a single network.

Our work assumed access to the reward preferences at the learning and deployment stages. While the RDF framework is robust to errors in these preferences by design, future work could explore how to learn these preferences from data. We also note that decision-focused learning can be used with complex transition model classes, such as neural networks, and that our RDF framework can be readily extended to these settings. We limited our work to simple transition models due to our focus on healthcare applications which often require interpretable models. Finally, we note that RDF learning proceeds by learning a separate policy for each reward preference. Future work could explore how to learn a single policy that can adapt to different reward preferences at deployment time. Combining RDF with multi-objective reinforcement learning, which has focused on the development of (mostly) model-free techniques to learn such policies, could be a promising direction for this work.

9.1. Clinical Applicability

Our approach could help in cases where multiple objectives are important from a clinical perspective (Zhang et al., 2022), by learning a simple transition model in MDP that focuses on its downstream use of providing policies for multiple rewards (MAP, Mortality probability, and Final Survival in the case of Zhang et al. (2022)). Our approach can also be beneficial in scenarios where there is legitimate uncertainty about which reward to use. For instance, in the selection of cancer dosing regimens (Yauney and Shah, 2018), the reward function specification required setting several coefficients. We adapted this environment in Section 7 by proposing a reward function that is a weighted combination of (1) the reduction in tumor size and (2) drug concentration.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. IIS-1750358, Grant No. IIS-2007076 and by NIH award R01MH123804. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the official views of the National Science Foundation or the National Institutes of Health.

References

Axel Abels, Diederik Roijers, Tom Lenaerts, Ann Nowé, and Denis Steckelmacher. Dynamic Weights in Multi-Objective Deep Reinforcement Learning. In *Proceedings of the 36th International Conference on Machine Learning*, pages 11–20. PMLR, May 2019. URL https://proceedings.mlr.press/v97/abels19a.html. ISSN: 2640-3498. 4

Milton Abramowitz and Irene A Stegun. *Handbook of mathematical functions with formulas, graphs, and mathematical tables*, volume 55. US Government printing office, 1968. 9

André Barreto, Shaobo Hou, Diana Borsa, David Silver, and Doina Precup. Fast reinforcement learning with generalized policy updates. *Proceedings of the National Academy of Sciences*, 117(48):30079–30087, Dec 2020. doi: 10.1073/pnas.1907370117. 5

Leon Barrett and Srini Narayanan. Learning All Optimal Policies with Multiple Criteria. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 41–47, New York, NY, USA, 2008. Association for Computing Machinery. ISBN 978-1-60558-205-4. doi: 10.1145/1390156.1390162. URL https://doi.org/10.1145/1390156.1390162. event-place: Helsinki, Finland. 4

Marc Peter Deisenroth. A survey on policy search for robotics. *Foundations and Trends in Robotics*, 2(1–2):1–142, 2011. ISSN 1935-8253, 1935-8261. doi: 10.1561/2300000021. 2, 6

Esther Derman, Matthieu Geist, and Shie Mannor. Twice regularized mdps and the equivalence between robustness and regularization. *Advances in Neural Information Processing Systems*, 34:22274–22287, 2021. 4

Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017. 7

Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6, 2005. 12

Amir-massoud Farahmand. Iterative Value-Aware Model Learning. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://papers.nips.cc/paper/2018/hash/7a2347d96752880e3d58d72e9813cc14-Abstract.html. 6

Amir-Massoud Farahmand, Andre Barreto, and Daniel Nikovski. Value-Aware Loss Function for Model-based Reinforcement Learning. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, pages 1486–1494. PMLR, April 2017. URL https://proceedings.mlr.press/v54/farahmand17a.html. ISSN: 2640-3498. 4, 7

Joseph Futoma, Michael Hughes, and Finale Doshi-Velez. POPCORN: Partially Observed Prediction Constrained Reinforcement Learning. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, pages 3578–3588. PMLR, June 2020. URL https://proceedings.mlr.press/v108/futoma20a.html. ISSN: 2640-3498. 3, 4, 19

Christopher Grimm, Andre Barreto, Satinder Singh, and David Silver. The value equivalence principle for model-based reinforcement learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 5541–5552. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/3bb585ea00014b0e3ebe4c6dd165a358-Paper.pdf. 3, 4, 7

Bryan A. Hassell, Girija Goyal, Esak Lee, Alexandra Sontheimer-Phelps, Oren Levy, Christopher S. Chen, and Donald E. Ingber. Human organ chip models recapitulate orthotopic lung cancer growth, therapeutic responses, and tumor dormancy in vitro. *Cell Reports*, 21(2):508–516, Oct 2017. ISSN 2211-1247. doi: 10.1016/j.celrep.2017.09.043. 7

Conor F. Hayes, Roxana Rădulescu, Eugenio Bargiacchi, Johan Källström, Matthew Macfarlane, Mathieu Reymond, Timothy Verstraeten, Luisa M. Zintgraf, Richard Dazeley, Fredrik Heintz, Enda Howley, Athirai A. Irissappane, Patrick Mannion, Ann Nowé, Gabriel Ramos, Marcello Restelli, Peter Vamplew, and Diederik M. Roijers. A practical guide to multi-objective reinforcement learning and planning. *Autonomous Agents and Multi-Agent Systems*, 36(1):26, April 2022. ISSN 1573-7454. doi: 10.1007/s10458-022-09552-y. URL https://doi.org/10.1007/s10458-022-09552-y. 4

Hisham Husain, Kamil Ciosek, and Ryota Tomioka. Regularized policies are reward robust. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, page 64–72. PMLR, Mar 2021. URL https://proceedings.mlr.press/v130/husain21a.html. 4

Donald E. Ingber. Human organs-on-chips for disease modelling, drug development and personalized medicine. *Nature Reviews Genetics*, 23(88):467–491, Aug 2022. ISSN 1471-0064. doi: 10.1038/s41576-022-00466-9. 7

Alistair EW Johnson, Lucas Bulgarelli, Lu Shen, Alvin Gayles, Ayad Shammout, Steven Horng, Tom J Pollard, Sicheng Hao, Benjamin Moody, Brian Gow, et al. Mimic-iv, a freely accessible electronic health record dataset. *Scientific data*, 10(1):1, 2023. 15

Joshua Joseph, Alborz Geramifard, John W. Roberts, Jonathan P. How, and Nicholas Roy. Reinforcement learning with misspecified model classes. In *2013 IEEE International Conference on Robotics and Automation*, pages 939–946, May 2013. doi: 10.1109/ICRA.2013.6630686. ISSN: 1050-4729. 3, 4, 6, 7

Daniel J. Lizotte, Michael Bowling, and Susan A. Murphy. Efficient Reinforcement Learning with Multiple Reward Functions for Randomized Controlled Trial Analysis. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, pages 695–702, Madison, WI, USA, 2010. Omnipress. ISBN 978-1-60558-907-7. event-place: Haifa, Israel. 2, 4, 7, 13

Alberto Maria Metelli, Matteo Papini, Nico Montali, and Marcello Restelli. Importance sampling techniques for policy optimization. *Journal of Machine Learning Research*, 21 (141):1–75, 2020. 18

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(75407540):529–533, Feb 2015. ISSN 1476-4687. doi: 10.1038/nature14236. 11

Hossam Mossalam, Yannis M. Assael, Diederik M. Roijers, and Shimon Whiteson. Multi-Objective Deep Reinforcement Learning, October 2016. URL http://arxiv.org/abs/1610.02707. arXiv:1610.02707 [cs]. 4

Evgenii Nikishin, Romina Abachi, Rishabh Agarwal, and Pierre-Luc Bacon. Control-Oriented Model-Based Reinforcement Learning with Implicit Differentiation. *Proceedings*

of the AAAI Conference on Artificial Intelligence, 36(7):7886–7894, June 2022. ISSN 2374-3468. doi: 10.1609/aaai.v36i7.20758. URL https://ojs.aaai.org/index.php/AAAI/article/view/20758. Number: 7. 3, 4, 6, 7, 10, 20

Chris Reinke and Xavier Alameda-Pineda. Xi-learning: Successor feature transfer learning for general reward functions. *arXiv preprint arXiv:2110.15701*, 2021. 5

Benjamin Ribba, Gentian Kaloshi, Mathieu Peyre, Damien Ricard, Vincent Calvez, Michel Tod, Branka Čajavec Bernard, Ahmed Idbaih, Dimitri Psimaras, Linda Dainese, Johan Pallud, Stéphanie Cartalat-Carel, Jean-Yves Delattre, Jérôme Honnorat, Emmanuel Grenier, and François Ducray. A Tumor Growth Inhibition Model for Low-Grade Glioma Treated with Chemotherapy or Radiotherapy. *Clinical Cancer Research*, 18(18):5071–5080, September 2012. ISSN 1078-0432. doi: 10.1158/1078-0432.CCR-12-0084. URL https://doi.org/10.1158/1078-0432.CCR-12-0084. 13

Abhishek Sharma, Catherine Zeng, Sanjana Narayanan, Sonali Parbhoo, and Finale Doshi-Velez. On learning prediction-focused mixtures. *arXiv preprint arXiv:2110.13221*, 2021. 3

Richard S. Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *ACM SIGART Bulletin*, 2(4):160–163, July 1991. ISSN 0163-5719. doi: 10.1145/122344.122377. 6

Runzhe Wan, Xinyu Zhang, and Rui Song. Multi-Objective Model-based Reinforcement Learning for Infectious Disease Control. KDD '21, pages 1634–1644, New York, NY, USA, August 2021. Association for Computing Machinery. ISBN 978-1-4503-8332-5. doi: 10.1145/3447548.3467303. URL https://doi.org/10.1145/3447548.3467303. 5, 11

Kai Wang, Sanket Shah, Haipeng Chen, Andrew Perrault, Finale Doshi-Velez, and Milind Tambe. Learning MDPs from Features: Predict-Then-Optimize for Sequential Decision Making by Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 34, pages 8795–8806. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/hash/49e863b146f3b5470ee222ee84669b1c-Abstract.html. 3, 4

Marco A. Wiering, Maikel Withagen, and Mădălina M Drugan. Model-based multi-objective reinforcement learning. In *2014 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, pages 1–6, December 2014. doi: 10.1109/ADPRL.2014.7010622. ISSN: 2325-1867. 5, 6, 11

Bryan Wilder, Eric Ewing, Bistra Dilkina, and Milind Tambe. End to end learning and optimization on graphs. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/hash/8bd39eae38511daad6152e84545e504d-Abstract.html. 3

Tomohiro Yamaguchi, Shota Nagahama, Yoshihiro Ichikawa, and Keiki Takadama. Model-Based Multi-objective Reinforcement Learning with Unknown Weights. In Sakae Yamamoto and Hirohiko Mori, editors, *Human Interface and the Management of Information. Information in Intelligent Systems*, Lecture Notes in Computer Science, pages

311–321, Cham, 2019. Springer International Publishing. ISBN 978-3-030-22649-7. doi: 10.1007/978-3-030-22649-7_25. 4, 5, 11

Gregory Yauney and Pratik Shah. Reinforcement Learning with Action-Derived Rewards for Chemotherapy and Clinical Trial Dosing Regimen Selection. In *Proceedings of the 3rd Machine Learning for Healthcare Conference*, pages 161–226. PMLR, November 2018. URL https://proceedings.mlr.press/v85/yauney18a.html. ISSN: 2640-3498. 13, 21

Kristine Zhang, Henry Wang, Jianzhun Du, Brian Chu, Aldo Robles Arévalo, Ryan Kindle, Leo Anthony Celi, and Finale Doshi-Velez. An interpretable rl framework for pre-deployment modeling in icu hypotension management. *npj Digital Medicine*, 5(1):173, 2022. 21

Appendix A. Alternate RDF Objective Formulation

The RDF objective can also be formulated in the integral terms

$$\theta_{\text{RDF}} \leftarrow \arg\min_{\theta} \int_{w \in \mathcal{U}} L_{T_*,R_w}(\theta) dw$$
$$\text{s.t. } J_{T_*,R_{\bar{w}}}(\pi^*(\theta, R_{\bar{w}})) \geq \delta \tag{17}$$

where $\mathcal{U}$ specifies the region over which we wish to be robust over.

While this objective is equivalent to the objective in Eqn 8 under the assumption that $P(w)$ is a uniform measure on the domain of $\mathcal{U}$, it has an intuitive interpretation: we wish to maximize the volume of the return achieved by our model θ. This intuition also motivates why we choose a uniform measure for $P(w)$.

Appendix B. Proof for Theorem 1

Theorem 2 *Let $R_{\bar{w}}$ be the learning-phase reward function with preference $\bar{w}$, and R_w be the reward function with an arbitrary preference w. Let Q^*_w be the optimal action-value function for the true MDP for reward function R_w. Let $B_w, \hat{B}^{DF}_w, \hat{B}^{RDF}_w$ denote the Bellman optimality operators under the true dynamics, DF model and RDF model respectively.*

Assume $\hat{Q}^{DF}_w$ and $\hat{Q}^{RDF}_w$ are fixed points under $\hat{B}^{DF}_w$ and $\hat{B}^{RDF}_w$ respectively. Further assume that the reward function is bounded, $R_w(s,a) \in [0, r_{max}] \forall s, a, w$.

DF Case *Consider a DF model trained on $R_{\bar{w}}$. If the Bellman operator induced by the DF model achieves the error*

$$\sup_{s,a} |B_{\bar{w}} \hat{Q}^{DF}_{\bar{w}}(s,a) - \hat{B}^{DF}_{\bar{w}} \hat{Q}^{DF}_{\bar{w}}(s,a)| = \epsilon^{DF}_{\bar{w}},$$

then

$$Q^*_w(s,a) - \hat{Q}^{DF}_w(s,a) \leq \frac{\epsilon^{DF}_w}{(1-\gamma)} \qquad \text{for } w = \bar{w} \tag{18}$$

$$Q^*_w(s,a) - \hat{Q}^{DF}_w(s,a) \leq \gamma \frac{r_{max}}{(1-\gamma)^2} \qquad \forall w \neq \bar{w} \tag{19}$$

RDF Case *Consider the RDF model trained with learning-phase preference $\bar{w}$ and deployment-phase reward preference distribution $P(w)$. For a $w \in P(w)$, if the Bellman operator $\hat{B}^{RDF}_w$ induced by the RDF model achieves the error*

$$\sup_{s,a} |B_w \hat{Q}^{RDF}_w(s,a) - \hat{B}^{RDF}_w \hat{Q}^{RDF}_w(s,a)| = \epsilon^{RDF}_w,$$

then

$$Q^*_w(s,a) - \hat{Q}^{RDF}_w(s,a) \leq \frac{\epsilon^{RDF}_w}{(1-\gamma)} \tag{20}$$

For, $w \neq \bar{w}$, the RDF bound is tighter since we explicitly optimize ϵ^{RDF}_w whereas $\gamma \frac{r_{max}}{(1-\gamma)^2}$ is constant.

Proof First, we show the bound for the RDF and DF($w = \bar{w}$) case. Then we show the DF($w \neq \bar{w}$) bound.

RDF and DF($w = \bar{w}$) case $\forall s, a$ our Q approximation can be written as follows,

$$\left|Q_w^*(s,a) - \hat{Q}_w^{RDF}(s,a)\right| \tag{21}$$

$$= \left|B_w Q_w^*(s,a) - \hat{B}_w^{RDF}\hat{Q}_w^{RDF}(s,a)\right| \tag{22}$$

$$\leq \left|B_w \hat{Q}_w^{RDF}(s,a) - \hat{B}_w^{RDF}\hat{Q}_w^{RDF}(s,a)\right| + \left|B_w Q_w^*(s,a) - B_w \hat{Q}_w^{RDF}(s,a)\right| \tag{23}$$

$$= \epsilon_w^{RDF} + \left|r_w(s,a) + \gamma \mathbb{E}_{T^*(s,a)}\left[\max_{a'} Q_w^*(s',a')\right] - r_w(s,a) - \gamma \mathbb{E}_{T^*(s,a)}\left[\max_{a'} \hat{Q}_w^{RDF}(s',a')\right]\right| \tag{24}$$

$$= \epsilon_w^{RDF} + \gamma \left|\mathbb{E}_{T^*(s,a)}\left[\max_{a'} Q_w^*(s',a') - \max_{a'} \hat{Q}_w^{RDF}(s',a')\right]\right| \tag{25}$$

$$\leq \epsilon_w^{RDF} + \gamma \|T^*(s,a)\|_1 \| \max_{a'} Q_w^*(\cdot,a') - \max_{a'} \hat{Q}_w^{RDF}(\cdot,a')\|_\infty \tag{26}$$

$$\leq \epsilon_w^{RDF} + \gamma \max_{s',a'} \left|Q_w^*(s',a') - \hat{Q}_w^{RDF}(s',a')\right| \tag{27}$$

Taking a max over s, a yields

$$\max_{s,a} \left|Q_w^*(s,a) - \hat{Q}_w^{RDF}(s,a)\right| \leq \epsilon_w^{RDF} + \gamma \max_{s,a} \left|Q_w^*(s,a) - \hat{Q}_w^{RDF}(s,a)\right| \tag{28}$$

$$\implies \max_{s,a} \left|Q_w^*(s,a) - \hat{Q}_w^{RDF}(s,a)\right| \leq \frac{\epsilon_w^{RDF}}{(1-\gamma)} \tag{29}$$

For the DF($w = \bar{w}$) case, we get an analogous bound:

$$\max_{s,a} \left|Q_{\bar{w}}^*(s,a) - \hat{Q}_{\bar{w}}^{DF}(s,a)\right| \leq \frac{\epsilon_{\bar{w}}^{DF}}{(1-\gamma)} \tag{30}$$

DF($w \neq \bar{w}$) case First, we show that for any $Q(s,a)$ and $w \neq \bar{w}$,

$$\left|B_w Q(s,a) - \hat{B}_w^{DF} Q(s,a)\right| \tag{31}$$

$$= \gamma \left|\mathbb{E}_{T^*(s,a) - T^{DF}(s,a)}\left[\max_{a'} Q(s',a')\right]\right| \tag{32}$$

$$\leq \gamma \left|\mathbb{E}_{T^*(s,a) - T^{DF}(s,a)}\left[\max_{a'} Q(s',a')\right] - \frac{r_{max}}{2(1-\gamma)}\right| \tag{33}$$

$$\leq \gamma \|T^*(s,a) - T^{DF}(s,a)\|_1 \| \max_{a'} Q(\cdot,a') - \frac{r_{max}}{2(1-\gamma)}\mathbf{1}\|_\infty \tag{34}$$

$$\leq \gamma \|T^*(s,a) - T^{DF}(s,a)\|_1 \frac{r_{max}}{2(1-\gamma)} \tag{35}$$

$$\leq \gamma \frac{r_{max}}{(1-\gamma)} \tag{36}$$

Now, for a (s,a) and $w \neq \bar{w}$

$$\left|Q_w^*(s,a) - \hat{Q}_w^{DF}(s,a)\right| \tag{37}$$

$$= \left|B_w Q_w^*(s,a) - \hat{B}_w^{DF}\hat{Q}_w^{DF}(s,a)\right| \tag{38}$$

$$\leq \left|B_w \hat{Q}_w^{DF}(s,a) - \hat{B}_w^{DF}\hat{Q}_w^{DF}(s,a)\right| + \left|B_w Q_w^*(s,a) - B_w \hat{Q}_w^{DF}(s,a)\right| \tag{39}$$

$$\leq \left|B_w \hat{Q}_w^{DF}(s,a) - \hat{B}_w^{DF}\hat{Q}_w^{DF}(s,a)\right| + \gamma \max_{s',a'} \left|Q_w^*(s',a') - \hat{Q}_w^{DF}(s',a')\right| \tag{40}$$

$$\leq \gamma \frac{r_{max}}{(1-\gamma)} + \gamma \max_{s',a'} \left|Q_w^*(s',a') - \hat{Q}_w^{DF}(s',a')\right| \tag{41}$$

which yields

$$\max_{s,a} \left|Q_w^*(s,a) - \hat{Q}_w^{DF}(s,a)\right| \leq \gamma \frac{r_{max}}{(1-\gamma)^2} \tag{42}$$

■

Appendix C. Environment Details

C.1. Synthetic MDP

The transition matrix values were sampled from a standard normal distribution before being normalized using a softmax transformation. There are two reward matrices, and their values were uniformly sampled in $[-60, 40]$ before being clipped to $[0, 40]$.

We share the code for generating the synthetic MDP for validating the Theorem 1 in the main paper. The code is written in Python and uses PyTorch.

```
data_seed = 0
torch.manual_seed(data_seed)
num_actions, num_states = 2, 20
dtype_float = torch.float64

# Transition matrix
# Shape: [num_actions, num_states, num_states]
true_transition = torch.randn(num_actions, num_states,
    num_states, dtype=dtype_float)
true_transition = torch.softmax(true_transition, dim=-1)
true_transition[:,-1] = 0
true_transition[:,-1,-1] = 1

# Reward matrix
# Shape: [num_states, num_actions]
true_reward_1 = (torch.rand(num_states, num_actions, dtype=
    dtype_float))
true_reward_2 = (torch.rand(num_states, num_actions, dtype=
    dtype_float))
```

```
true_reward_1 = true_reward_1.clamp(0.6,) - 0.6
true_reward_2 = true_reward_2.clamp(0.6,) - 0.6
true_reward_1 *= 100
true_reward_2 *= 100
true_reward_1[-1,:] = 0
true_reward_2[-1,:] = 0
get_true_reward = lambda w: true_reward_1 * w + true_reward_2
    * (1 - w)

gamma = 0.9
temperature = 0.01
```

Appendix D. Additional Plots

D.1. Cancer Treatment

D.1.1. Sensitivity analysis for λ and δ parameters

The optimization problem in Eqn 8 is solved by the saddle-point algorithm, where the Lagrange multiplier λ is updated depending on whether the constraint is satisfied or not. However, we can also solve the optimization problem by fixing λ and running the optimization on a grid of such λ values.

In figureD.1.1, we show the implicit constraint value δ achieved by the optimization problem for different values of λ.

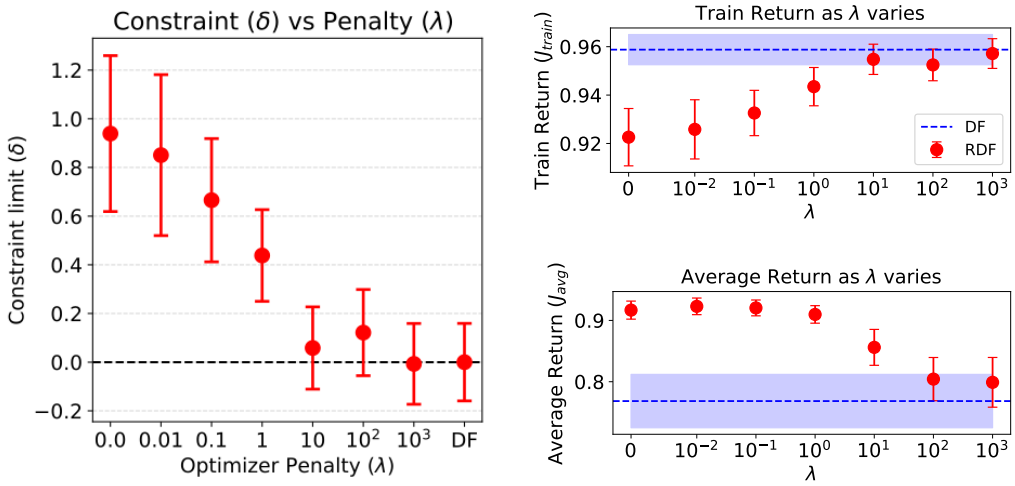

Figure 6: Sensitivity analysis of the λ hyperparameter. **Left:** λ vs δ for the cancer treatment problem. **Right:** λ vs learning-phase (i.e. train) and average return for the cancer treatment problem.

D.1.2. Learned model coefficients for MLE, DF, and RDF models for the Cancer Treatment domain

Outcome	State/Action	MLE Coefficients	DF Coefficients	RDF Coefficients
Concentration	Intercept	-0.00	-0.01	-0.01
	Concentration	0.76	0.64	0.83
	Proliferative Tissue	-0.00	-0.05	-0.06
	Non-proliferative Quiescent Tissue	-0.00	0.02	-0.03
	Damaged Quiescent Cells	-0.00	-0.10	-0.01
	Time Step	0.00	-0.10	0.13
	Action	0.76	0.60	0.69
Proliferative Tissue	Intercept	0.95	1.03	1.03
	Concentration	-0.47	-0.52	-0.39
	Proliferative Tissue	0.93	0.92	0.96
	Non-proliferative Quiescent Tissue	-0.01	0.00	-0.08
	Damaged Quiescent Cells	-0.02	-0.11	0.04
	Time Step	0.01	-0.07	0.07
	Action	-0.62	-0.58	-0.62
Non-proliferative Quiescent Tissue	Intercept	3.02	3.00	2.51
	Concentration	-1.76	-1.78	-1.75
	Proliferative Tissue	-0.35	-0.43	-0.50
	Non-proliferative Quiescent Tissue	0.97	0.83	0.84
	Damaged Quiescent Cells	-0.08	-0.01	0.13
	Time Step	0.07	0.05	0.08
	Action	-2.45	-2.32	-2.45
Damaged Quiescent Cells	Intercept	-2.66	-2.66	-2.17
	Concentration	1.36	1.24	1.33
	Proliferative Tissue	0.30	0.26	0.36
	Non-proliferative Quiescent Tissue	0.04	0.21	-0.02
	Damaged Quiescent Cells	1.06	1.13	0.93
	Time Step	-0.05	-0.21	0.01
	Action	1.89	1.95	2.00

D.1.3. Mean Tumor Diameter Trajectories for different reward preferences - RDF

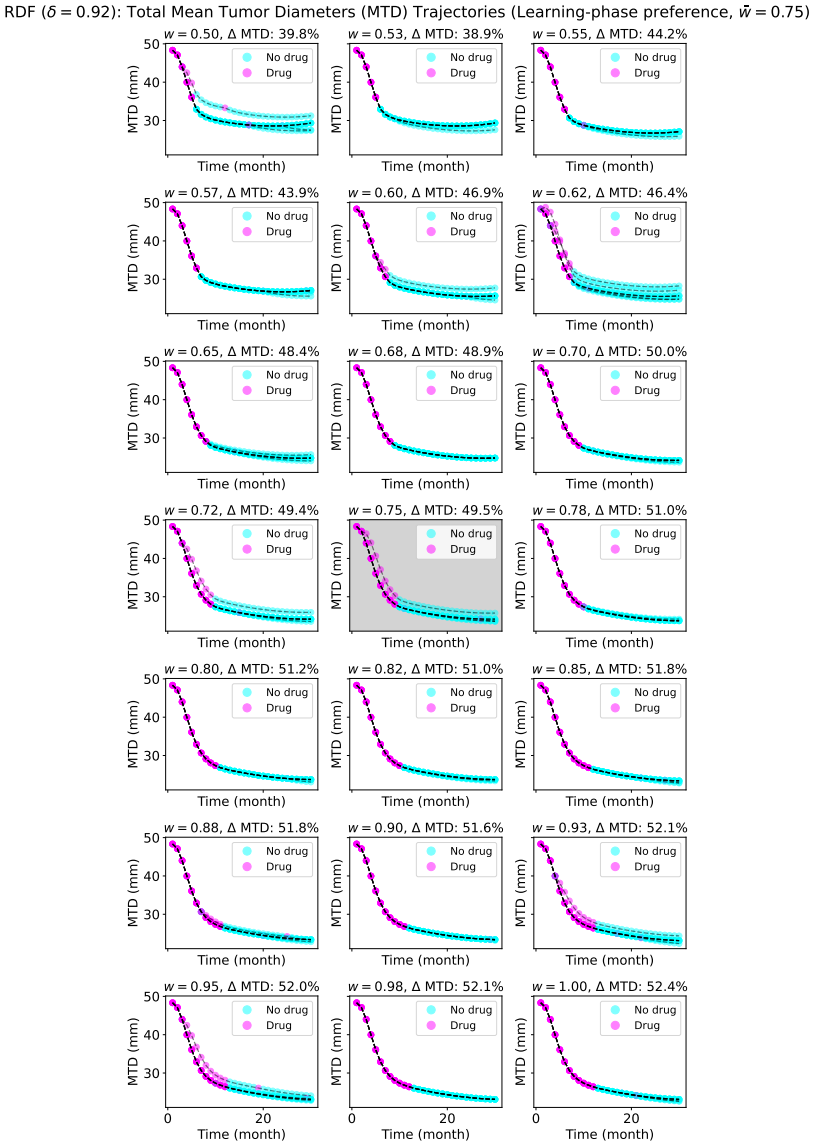

Figure 7: Mean tumor diameter trajectories for different reward preferences with the RDF model. The learning-phase reward preference is $\bar{w} = 0.75$ and the deployment-phase reward preference is $w \in [0.5, 1.0]$.

D.1.4. Mean Tumor Diameter Trajectories for different reward preferences - DF

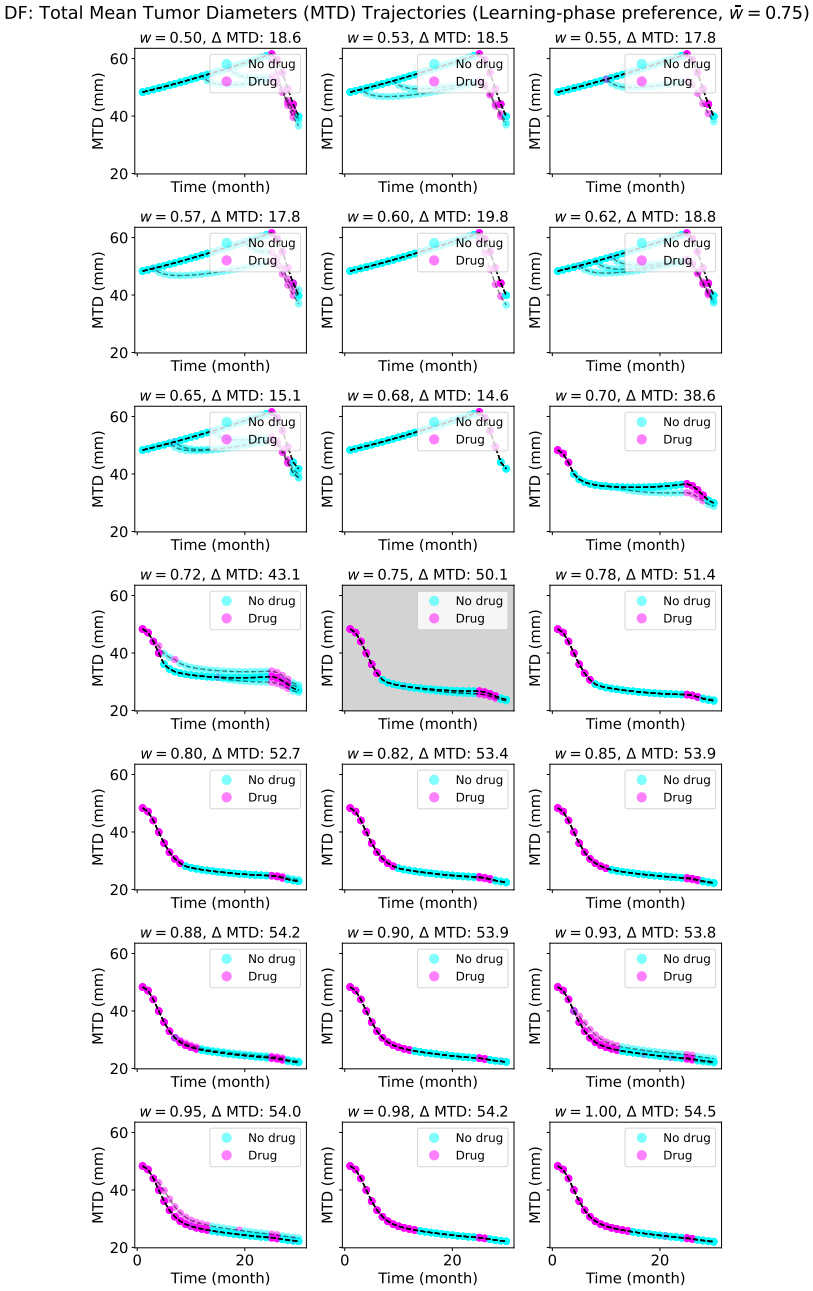

Figure 8: Mean tumor diameter trajectories for different reward preferences with the DF model. The learning-phase reward preference is $\bar{w} = 0.75$ and the deployment-phase reward preference is $w \in [0.5, 1.0]$.

D.1.5. Drug Concentration Trajectories for different reward preferences - RDF

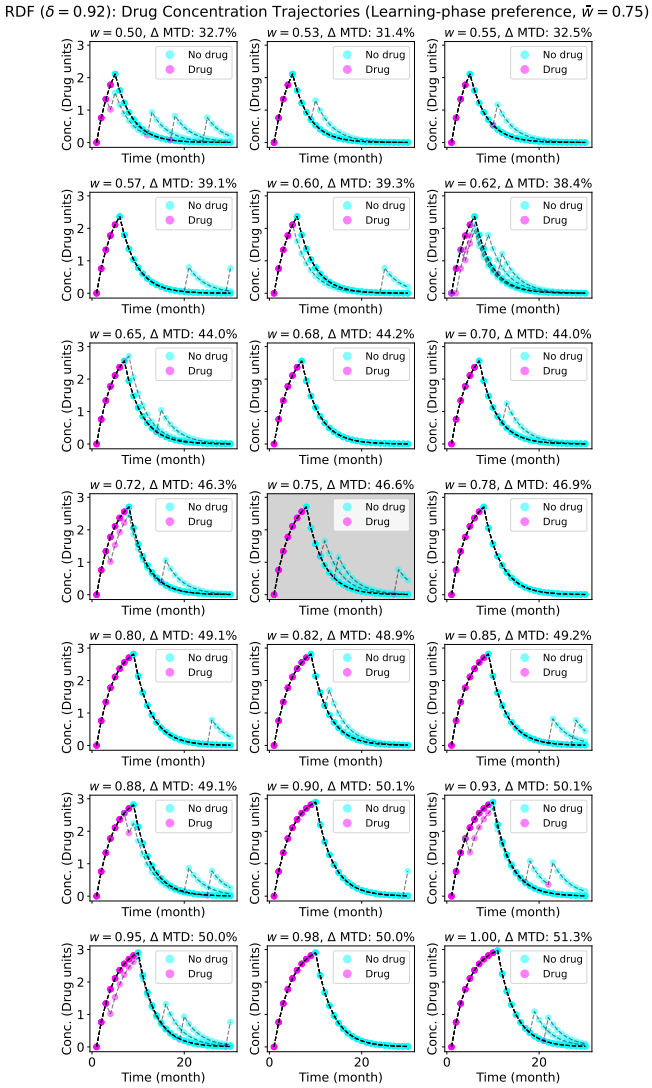

Figure 9: Mean tumor diameter trajectories for different reward preferences with the RDF model. The learning-phase reward preference is $\bar{w} = 0.75$ and the deployment-phase reward preference is $w \in [0.5, 1.0]$.

D.1.6. Drug Concentration Trajectories for different reward preferences - DF

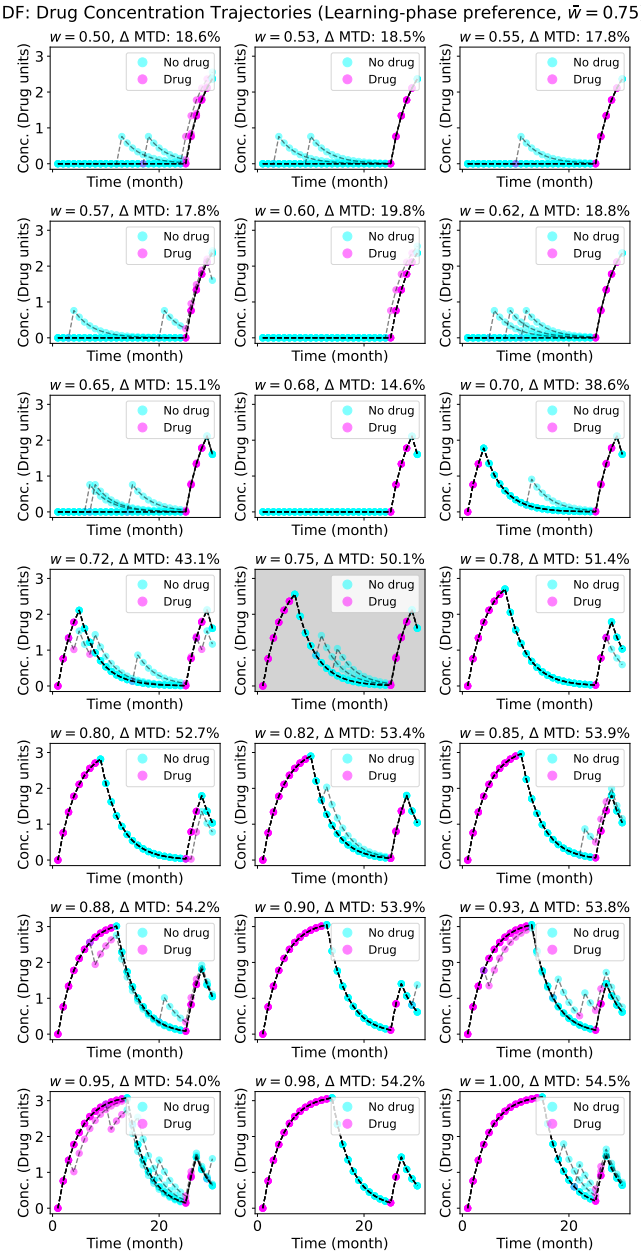

Figure 10: Mean tumor diameter trajectories for different reward preferences with the DF model. The learning-phase reward preference is $\bar{w} = 0.75$ and the deployment-phase reward preference is $w \in [0.5, 1.0]$.

D.2. MIMIC-IV Acute Hypotension dataset

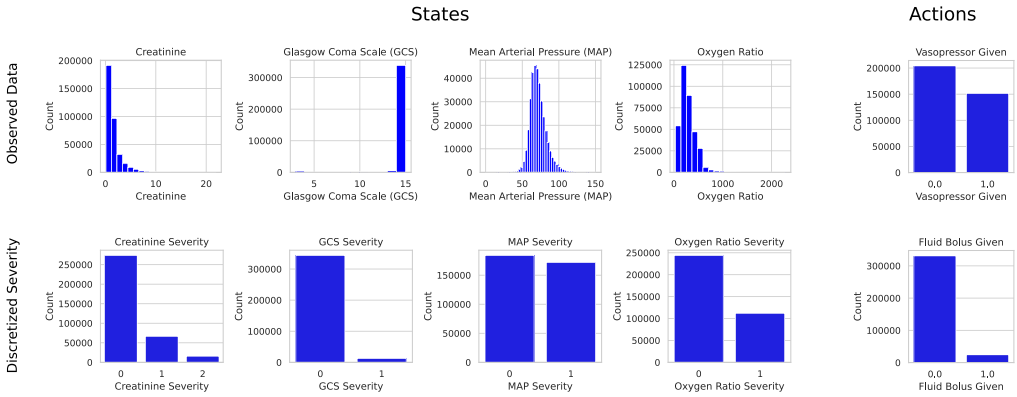

Figure 11: Distributions of states and actions in MIMIC Hypotension dataset

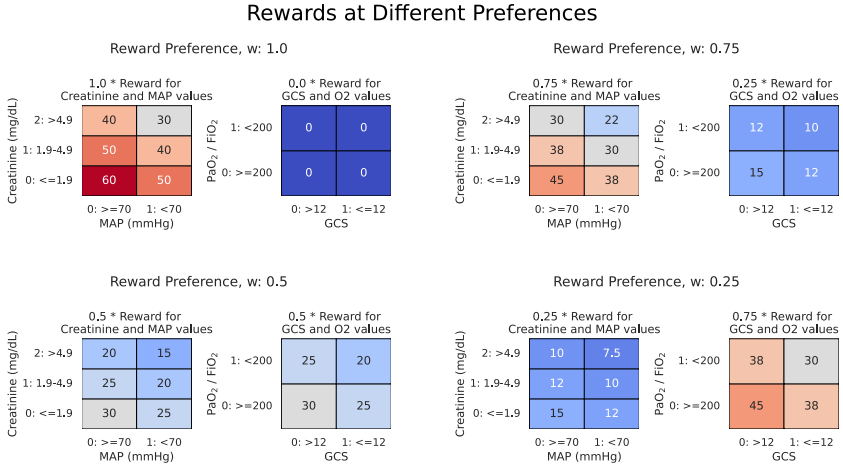

Figure 12: Rewards at different preferences in the MIMIC Hypotension dataset

D.2.1. Table describing the discretization for the state space

Abbreviation	Clinical Variable	Threshold	Bin Value
O2	Partial Pressure of Oxygen / Fraction Inspired Oxygen	≥ 200	0
O2	Partial Pressure of Oxygen / Fraction Inspired Oxygen	< 200	1
BP	Mean Blood Pressure	≥ 70 mmHg	0
BP	Mean Blood Pressure	< 70 mmHg	1
GCS	Glasgow Coma Scale (GCS)	≤ 12	0
GCS	Glasgow Coma Scale (GCS)	> 12	1
Crea	Creatinine	≤ 1.9 mg/dL	0
Crea	Creatinine	> 1.9 and ≤ 4.9 mg/dL	1
Crea	Creatinine	> 4.9 mg/dL	2

Table 2: Discretization of Clinical Variables into Bins

D.2.2. RDF Transition Dynamics for MIMIC Hypotension data

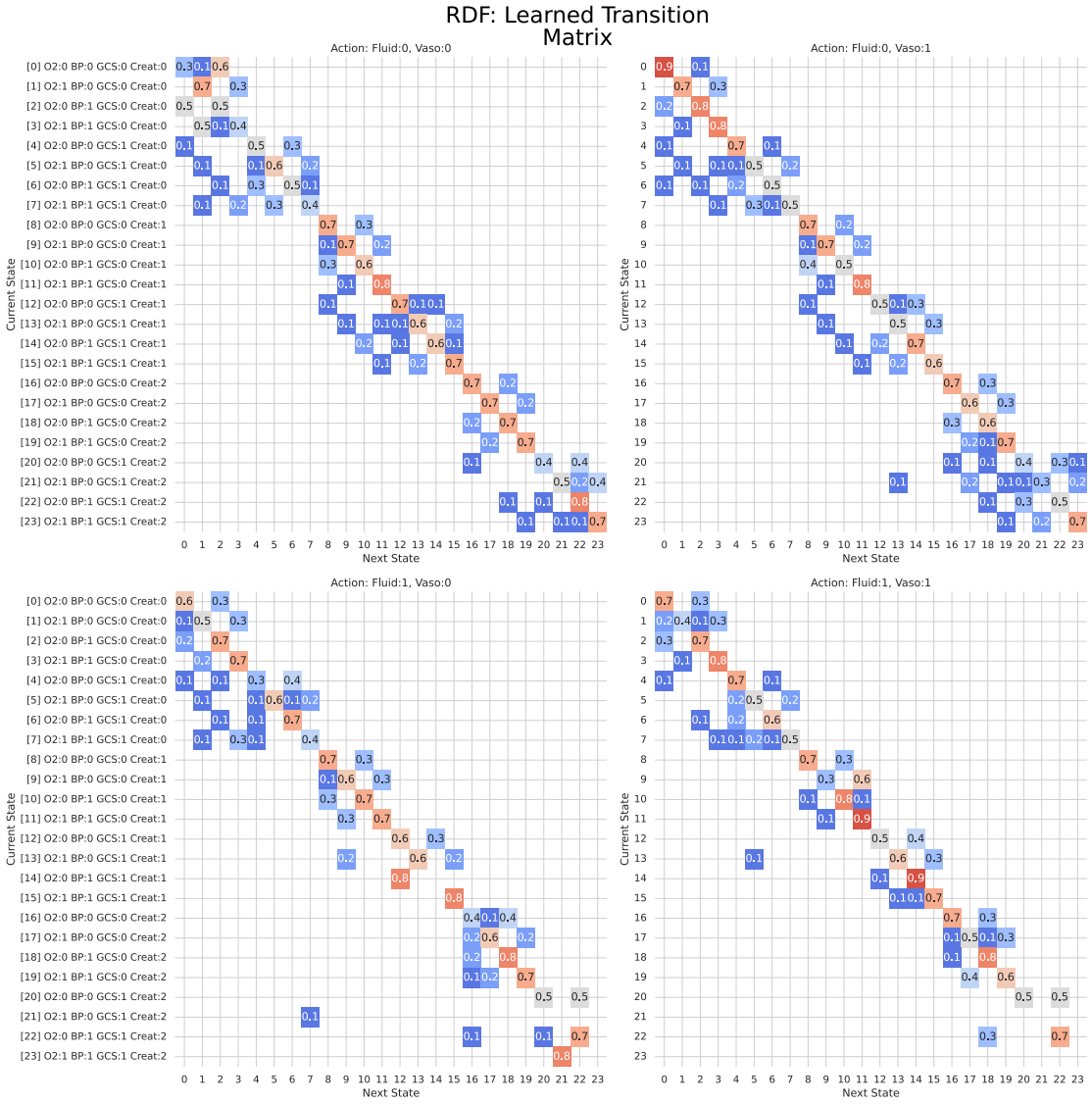

Figure 13: RDF Transition Dynamics for MIMIC Hypotension data.

D.2.3. DF Transition Dynamics for MIMIC Hypotension data

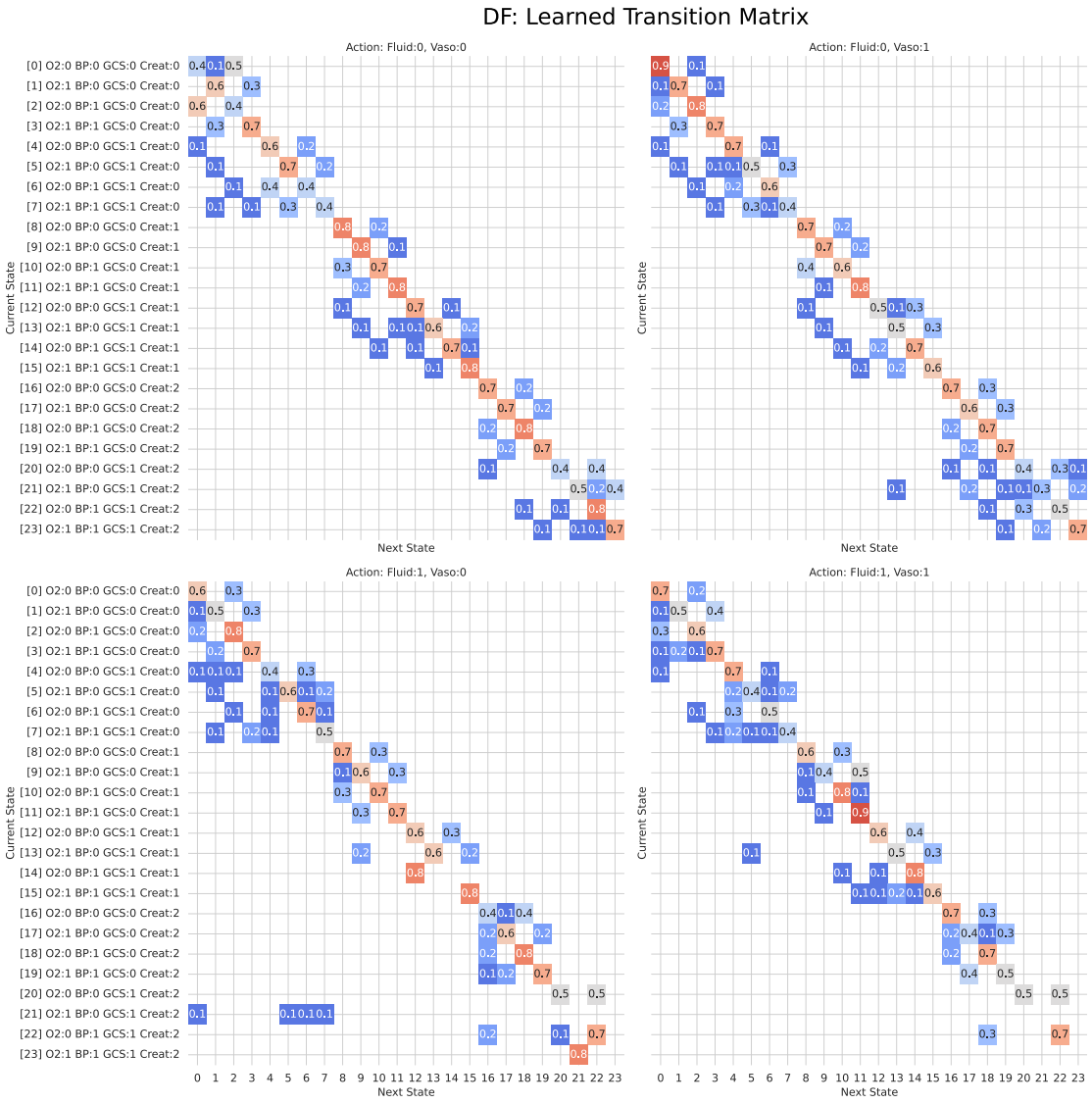

Figure 14: DF Transition Dynamics for MIMIC Hypotension data.

D.2.4. MLE Transition Dynamics for MIMIC Hypotension data

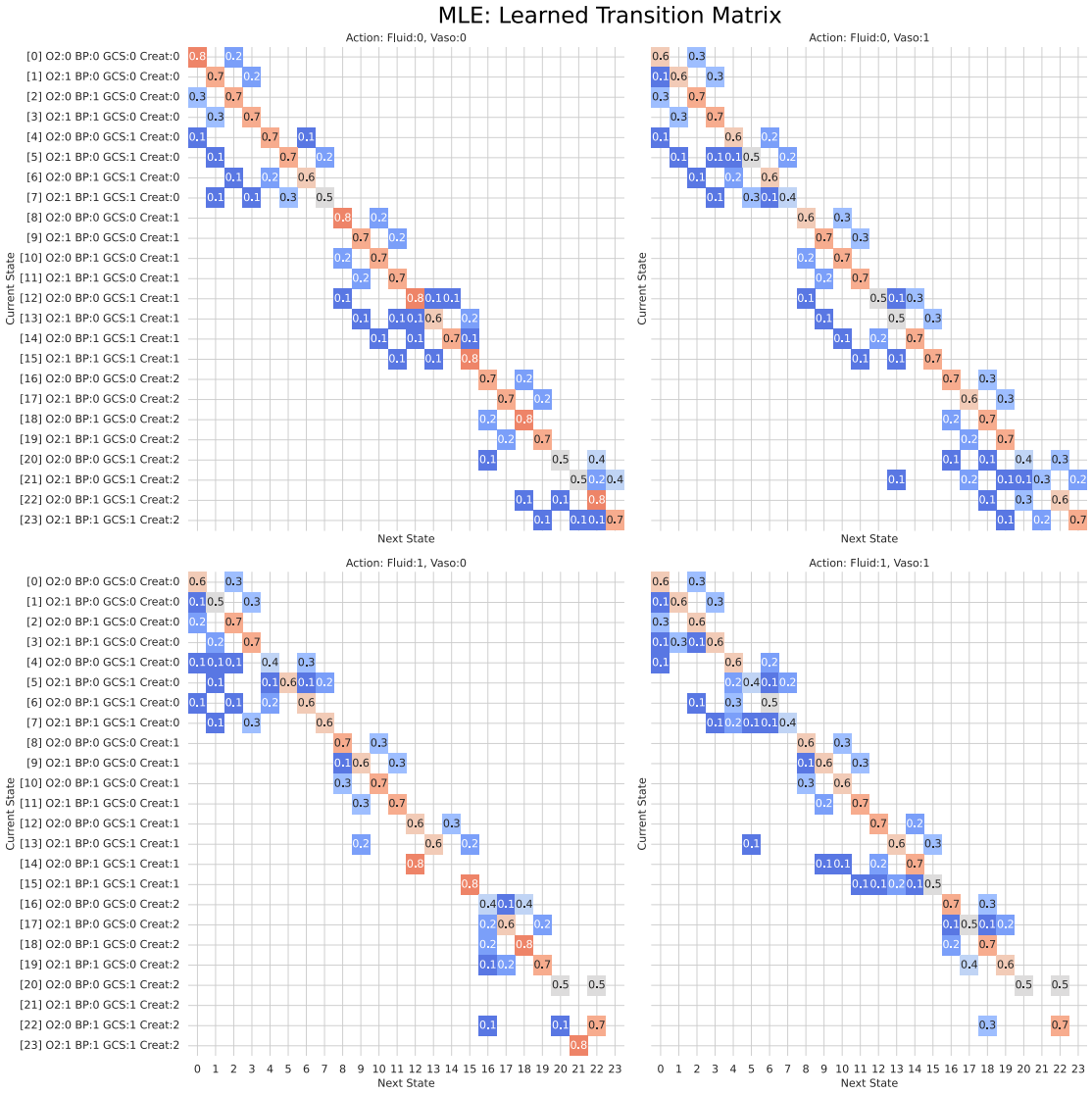

Figure 15: MLE Transition Dynamics for MIMIC Hypotension data.