Value Interpretable Dynamic Treatment Regimes

Anonymous authors
Paper under double-blind review

Abstract

Understanding the rationale behind the decision-making process proves invaluable in various applications of AI and ML, such as in healthcare. In this paper, we propose Value Interpretable Dynamic Treatment Regimes (VIDTR) to identify an optimal decision tree policy when finding the best treatment regime in sequential settings. Our approach works by identifying and integrating rewards over state regions that geometrically form a disjoint union of boxes at each iteration step. We assume that we have access to the transitions and rewards that generated the trajectory data. We compare and contrast 3 different optimization techniques that approximates the final policy. One which uses a Q-tree structure to come up with an estimate of the parameters, a procedure where we frame this as a mixed integer optimization problem, and a final procedure based on genetic algorithms where we cross-over based on the sub-conditions for each time and lengthstep. Finally, through various experiments, we examine how our method performs compared to existing methods in the simulated setup. We also give theoretical results on the various optimizations.

1 Introduction

It is no coincidence that widespread adoption of AI algorithms is a requirement in this age. For the adoption of these algorithms in human-centric fields such as healthcare, the interpretability of the techniques is a necessity. For physicians to make quick decisions about the treatment they want to give a patient, the algorithm output must be interpretable. Recently, there has been a surge in research dedicated to the design of artificial intelligence tools in healthcare ((3), (1), (6)).

(13) posits that we are going to save up to 10^8 dollars in 10 years by implementing AI systems in healthcare. To enable the fusion of AI tools in healthcare, there has been an uptake of interest in the development of human-algorithm "centaurs" ((19), (23), (24)). These centaurs usually work based on the principle of generating algorithmic feedback, which goes through external human validation to generate a final output. As an example in (19), a survey is used to generate expert human output that can guide the decision-making provided by the machine learning algorithm.

To further develop this line of research, we wish to develop tools that ease the burden of decision making on doctors. Toward that goal, we must create algorithms that are 'interpretable'. Reinforcement learning- a branch of machine learning that enables the automation of dynamic decision making is to be deployed in fields such as healthcare. (31) provides a recent account of the different reinforcement learning methods that are being deployed in this field.

(29) gives a deep reinforcement learning approach for the real-time treatment of sepsis, where the authors propose a weighted dueling deep Q network with embedded human expertise. In this paper, a Q target value with adaptive dynamic weight is designed to improve the accuracy of the estimate. In (12) the authors come up with a continuous space and action solution to provide recommendations for the treatment of sepsis inspired by the Deep Deterministic Policy Gradient (DDPG) method. In (16) a supervised learning framework is used to determine the actions to

be taken by human experts. Further deep reinforcement learning is used to estimate the value function for expert actions with the highest probabilities, and recommendations are made amongst treatment options.

The literature on dynamic treatment regimes (5) provides the concept of a DTR from a statistical perspective and provides the basis for future developments. (21) work on structured nested models is fundamental for the development of statistical methods in dynamic treatment regimes, particularly in handling time-varying treatments and confounders. (14) discusses adaptive strategies in clinical trials that form the basis for understanding sequential treatment adjustments. (17) discusses the use of deep RL to optimize treatment regimes in healthcare, addressing challenges such as high-dimensional data and complex treatment sequences. (20) discusses the development of treatment rules that maximize patient outcomes, providing performance guarantees for the derived policies.

When we are provided the environmental parameters for a Markov Decision Process, we come up with an approach to design a model-based interpretable greedy algorithm to compute interpretable policies, called the Value Interpretable Dynamic Treatment Regimes (VIDTR). The VIDTR algorithm provides inbuilt interpretability by performing backward induction on regularized versions of the Bellman equations. Borrowing ideas from (32) we design a model-based version of their technique, where we add volume promotion and complexity penalization errors as part of the regularization. This is added to the derived L^2 penalty obtained from the Bellman equations. We provide bounds in a simulated setting for the experiments and verify them in the grid-based setup, where we perform sensitivity analysis with respect to the parameters of the experimental setup.

We exhibit better theoretical bounds than the model-free case and exihibit the working of this method in the grid-based data where we get a perfect error score of zero.

Other literature in the space of model-based interpretable reinforcement learning would be from [Petrik and Luss], where they provide an approach to obtain Interpretable Policies by solving the long horizon reinforcement learning problem when framed as a **Mixed Integer Linear Programming** [MILP] problem. Solving the global problem is not scalable when compared to the greedy approach presented in our paper.

Further we compare and contrast with ideas from [Julien Grand Clement] where he frames the problem as a minimization problem over a set of trees over an MDP. They evaluate the cost of each tree when an MDP is fixed and pick the minimum by doing reverse induction on the space of trees. Our approach is similar in that we do backwards induction on the space of policies, however we use a greedy algorithm to pick the splits.

Further we recall the paper from the authors on the cost of interpretability - blocked value iteration. The above paper speaks of different notions of bounding the cost of interpretability when different notions of interpretability are presented. In this paper, the notion of intepretability assumed would be list based maps.

Our contributions in the following paper can be summarized as follows.

1. **Model based interpretable policies for MDPs**: We extend the paper by (33) to derive a model-based version of interpretable dynamic treatment regimes where we compute a greedy model-based interpretable policy at different time steps.

2. **Implementation** : We design a data structure called DisjointBoxUnion in which we can perform geometric set operations on a disjoint union of boxes. Examples of this would include unions, intersections, subtractions, and integration of a function over this region. We also design a data structure called a constraint-condition, from which we can derive the constraints over which we perform set operations over.

3. **Theoretical bounds** : We compute theoretical bounds which assess the performance of the derived interpretable policy for the different time and length steps.

4. **Q-Trees based optimization** : We propose an optimization of the above where we extend Zhang's tree-based approach where they solve the one-parameter interval compounding problem to solving an optimization problem where we need to compute the compounding intervals.

5. **Tight Value Interpretable Dynamic Treatment Regimes** : We design a tight version of the algorithm under assumptions of the transitions and rewards being close to each other in time and by taking a convex approximation of the objective.

6. **Evolutionary algorithms** : We design and provide an implementation of a genetic algorithm based approach for solving the subproblem in each time and lengthstep for the VIDTR.

7. **MIO optimization** : To derive an exact solution, we frame the above problem as a mixed integer optimization problem and derive an exact solution correspondingly.

8. **Experimental setup** : We design experiments and setups where we test the final policy and review the performance of the derived interpretable policy. We compare our work against (8), (32) and an extension of the (32) work by using disjoint box unions to store regions and integrate functions over.

9. **Real world data** : We check the performance of the derived policy on real-world data, where we have to designate treatments to patients at the different timesteps.

2 Literature Review

In this section we review the literature which consists of different ways to derive the optimal interpretable policy for a Markov Decision Process.

The first step to perform is to derive estimates of the transition and reward vectors for the offline reinforcement learning problem. (2) devise an active learning procedure to derive the model parameters for Markov chains and Markov decision processes. The paper presents a model based active learning sampling procedure that chooses examples which are most informative with respect to the current model hypothesis. This is based on active learning of MDPs using the Baum Welch algorithm. The authors of (30) propose two regularized maximum likelihood estimation algorithms for learning the transition probability model and policy. They also provide a sample complexity result that shows that a lower regret can be achieved with a relatively small number of training samples. (26) formalizes the problem of learning an accurate model for an MDP. Here we sample states from regions where we wish to maximize the entropy of the empirical state frequency.

After deriving the transitions and rewards, we move on to find interpretable policies. Here we present approaches in the literature when the same is done. The approach presented in (9) works based on reverse induction of the Bellman equations. After having obtained the transitions and the rewards for the different timesteps to compute the value functions. After deriving the value function, we fit a classification tree T_t which fits the value function at that timestep. Our approach computes the (approximate) solution to the forward RL problem and then performs reverse induction using the mean squared error with regularization on the complexity and volume promotion at that timestep. The approach presented in (7) introduces a fitted Q value iteration procedure where we similarly learn the Q-values by doing a backwards induction on the Bellman equations. This approach is a model free version of the one presented in (9). The authors of (25) introduce parameterized versions of decision trees over which we can differentiate on. Assuming this as the class of policies which we need to optimize over, they introduce a policy gradient method over this set of policies where we differentiate over the splitting parameters. This ensures that we maximize for the total discounted return over the whole space. In our approach we compute the best split in a greedy fashion at each time and lengthstep. In (28) we obtain optimal decision tree policies using an MDP formulation. Here we assume a linear programming formulation to compute the optimal policy for the MDP. To

this formulation, we add additional constraints to optimize only over the class of all decision tree policies. Our method differs in the sense that we obtain a greedy solution which is arguably quicker but would yield a larger runtime.

We can derive the optimal interpretable policies directly from the trajectory data. Here we present approaches from the literature where this is done. Our work can be thought of as a model based version of the algorithm presented in (32), where the authors compute list based approximations to the final policy. They perform regression to obtain the estimates of the Q-values at the different timesteps, they pull back the Q-values using Bellman's equations for which they fit a decision tree at each timestep. We directly estimate the model parameters to which we perform reverse induction with the Bellman equations. In the paper by (11), the authors introduce the genetic programming for reinforcement learning approach based on model based reinforcement learning and genetic programming, which autonomously learns policy equations from pre-existing state-action trajectory samples. In (4) the authors construct a verifiable policy from the ones trained using a DNN as in the algorithm presented in (22). This involves coming up with an imitation learning procedure to the algorithm learnt by the DNN in (22).

(15) presents a parametric gradient based policy update where we have access to the self driving car environment to perform the updates. Here we compute the return for a parametric policy, where interpretability is defined based on the values taken by the parameters. For the tree based setup, one fixes a tree-based structure based on expert knowledge and in the RL algorithm we optimize over the parameter space to tune the parameters. Our approach works based on greedy splits and the tree structure is also optimized for. In (27) we wrap a base MDP to form an IBMDP and solve using a modified RL algorithm. The solution is a decision tree policy for the base environment. In (10), the authors represent a decision tree where the leaves represent parametric policies and the various parameters represent the splits. With the total return over the MDP as the objective, we seek to optimize over the given parameters. (18) defines the notion of interpretability as given by simple closed form formulas in the policy space using which we rank actions. In this case, the search for the optimal policy turns out to be a multi-armed bandit problem where each arm represents a policy in the shortest expression to express it.

3 Problem Formulation

We aim to answer the following question, given knowledge of the modelling parameters, how does one come up with an optimal interpretable policy?

3.1 Notation

1. T : Time Horizon of the MDP
2. $(S_t)_{t=1}^T$: The state spaces for the different time steps for the MDP.
3. $\mathcal{A}$: Action space for the MDP
4. $P_t : S_{t+1} \times S_t \times \mathcal{A} \to [0,1], \forall t \in [0,T]$: The probability transition kernels for the different timesteps for the MDP.
5. $r_t : S_t \times \mathcal{A} \to \mathbb{R}, \forall t \in [0,T]$: The reward maps for the different timesteps in the MDP.
6. $\gamma \in (0,1)$: Discount factor for the Markov Decision Process
7. $(L_t)_{t=1}^T$: Maximum allowed depths for the different lengthsteps.
8. k: Maximum complexity of the final policy.
9. $(\rho_t)_{t=1}^T$: The complexity constants in the constraint condition for the different timesteps.

10. $(\eta_t)_{t=1}^T$: Volume split promotion constant in the optimization problem for the different timesteps.
11. $[(R_t^l, a_t^l)] \forall t \in [0, T]$ and $\forall l \in [0, l_t^*]$: The obtained interpretable policy at the different time and lengthsteps.

In this problem setting, we are given an MDP $((P_t)_{t=1}^T, (r_t)_{t=1}^T)$ over a finite horizon T and a discount factor γ. The algorithm proceeds by performing a backwards induction on the regularized Bellman equation. The parameters we tune in the algorithm are the maximum complexity k, maximum lengths per time $(L_t)_{t=1}^T$, η which is the volume promotion constant and ρ which is the complexity constant. The goal is to find interpretable policies $(\pi_t^I)_{t=1}^T$ such that an interpretable policy at the tth timestep is characterized by $[(R_t^l, a_t^l)_{l=1}^{L_t}]$ where R_t^l are the regions that are described by the conditions and a_t^l are the actions chosen at that length and timestep.

4 Algorithm Design

Since we perform reverse induction on the Bellman Equations, we start the procedure from the last timestep. At the last timestep, we perform the following analysis -:

4.1 Timestep T

At timestep T, we find a policy $\pi_T^I \in \mathcal{I}$ which approximates $\max_a r_T(s, a)$. At the first length-step, we perform

$$\pi_T^I(s) = \begin{cases} a_1 \text{ if } s \in R_1 \\ \arg\max_a r_T(s, a) \text{ otherwise} \end{cases}$$

Here $R_1 \in \mathcal{R}_1$ which given a maximum complexity k look like:

$$\{x : \exists i_1, i_2, ..., i_k s.t \pm x_{i_1} \leq c_1, ..., \pm x_{i_k} \leq c_k\}$$

Here we assume $i_1, i_2, ..., i_k \in \{1, 2, 3..., d\}$.

When compared to the optimal policy, we get $||V_{\pi^*} - V_{\pi_I}||$, which is given by:

$$\frac{1}{|S|} \int_S \max_a [r_T(s, a)] ds - \int_{R_1} r_T(s, a_1) ds - \int_{S - R_1} \max_a [r_T(s, a)] ds$$

We now wish to add regularization terms that penalize complexity and promote splitting, towards this goal we get:

This results in the following optimization problem:

$$\Psi_1^{T*} := \min_{R_1, a_1} \Psi(R_1, a_1) = \frac{1}{|S|} \int_{R_1} [\max_a [r_T(s, a)] - r_T(s, a_1)] ds - \eta_T V(R_1) + \rho_T c(R_1)$$

We provide an alternative to evaluate errors using the trajectory measure in the state action space instead of a uniform measure on the state space. This would borrow from the original IDTR method and one in which we evaluate error functionals over the given trajectories rather than the general state space.

Assume that the trajectories in this case are given by $[(X_{it}, A_{it}, R_{it})]$ for $i \in [N]$ and $t \in [T]$. Then what we observe is that the empirical error functional is given by:

$$\Psi_1^T := \min_{R_1, a_1} \frac{1}{N} \left[\sum_{i=1}^N (\max_{a \in A} r_T(X_{iT}, a) - r_T(X_{iT}, a_1) - \eta) \mathcal{I}(X_{iT} \in R_1) \right] + \rho_T c(R_1)$$

Here, since we try to compute $\min_{R\in\mathcal{R}}(-V(R))$, we maximize over $\eta V(R)$ at each lengthstep thereby promoting the volume splitting at each lengthstep. $c(R)$ is a parameter that would measure the complexity of the partitioning set. In the example of lists from the IDTR paper, we could measure the total number of parameters involved in the split. For example

$$c(\{x : \exists j_1, j_2 \text{ s.t } \pm x_{j_1} \leq c_1, \pm x_{j_2} \leq c_2\}) = 2 - I(c_1 = \pm\infty) - I(c_2 = \pm\infty)$$

In the next lengthstep we have:

$$\pi_I(s) = \begin{cases} a_1 \text{ if } s \in R_1 \\ a_2 \text{ if } s \in R_2 - R_1 \\ \arg\max_a r_T(s,a) \text{ otherwise} \end{cases}$$

We now assume $G_1^T = R_1^T$ The optimization problem is given by:

$$\Psi_2^T := \min_{R_2,a_2} \Psi(R_2, a_2) = \frac{1}{|S|}\int_{R_2 - G_1} [\max_a [r_T(s,a)] - r_T(s,a_1)]ds - \eta_T V(R_2 - G_1) + \rho_T c(R_2)$$

For the next lengthstep, we see that the empirical error is given by:

$$\Psi_1^T := \min_{R_1,a_1} \frac{1}{N}\left[\sum_{i=1}^{N} (\max_{a\in A} r_T(X_{iT}, a) - r_T(X_{iT}, a_1) - \eta)\mathcal{I}(X_{iT} \in R_2 - G_1)\right] + \rho_T c(R_2)$$

For the general lengthstep $l < l_T^*$ and timestep T, we have that the optimization problem is given by:

$$\Psi_l^{T*} := \min_{R_l^T, a_l^T} \Psi^T(R_l^T, a_l^T) = \int_{R_l^T - G_{l-1}^T} [\max_\alpha (r_T(s,\alpha)) - r_T(s, a_l^T)]ds - \eta_T V(R_l^T - G_{l-1}^T) + \rho_T c(R_l^T)$$

For the general lengthstep, we see that the empirical VIDTR error is given by:

$$\Psi_l^T := \min_{R_1^T, a_1^T} \frac{1}{N}\left[\sum_{i=1}^{N} [\max_{\alpha\in A} r_T(X_{iT}, \alpha) - r_T(X_{iT}, a_1^T)) - \eta_T]\mathcal{I}(X_{iT} \in S - G_{l_T - 1})\right] + \rho_T c(R_T)$$

Then we have

$$G_l^T = G_{l-1}^T \cup R_l^T$$

When $l = l_T^*$ we have:

$$\pi_I(s) = a_{l_T^*} \text{ if } s \in S - G_{l_T - 1}^T$$

$$\min_{a_l} \frac{1}{|S|}\int_{S - G_{l_T^* - 1}} [\max_{\alpha\in A}(r_T(s,\alpha)) - r_T(s, a_l)]ds$$

For the last lengthstep, we see that the empirical error is given by:

$$\Psi_l^T := \min_{a_l} \frac{1}{N}\left[\sum_{i=1}^{N} (\max_{\alpha\in A} r_T(X_{iT}, \alpha) - r_T(X_{iT}, a_1) - \eta)\mathcal{I}(X_{iT} \in R_l^T - G_{l-1}^T)\right] + \rho_T c(R_l^T)$$

For this particular case, we have that the estimator $Y(\lambda)$ is given by:

$$Y_{m,T}(s,a) := \left(\max_{\alpha\in A} r_T^m(s,\alpha) - r_T^m(s,a)\right)$$

In the above, if we take $m = (P, r)$ as the model parameters, then what follows is given below. To perform backward induction on the V-values we would need access to the optimal V values which we now compute using the Bellman's equation; we have

$$V_T^m(s) = \max_{\alpha \in A} r_T^m(s, \alpha)$$

In the empirical situation the value functions for the trajectory space points are estimated as follows:

$$V_T^{m,I}(s) = \begin{cases} r_T^m(s, a_1^t) \text{ if } s \in R_1^T \\ r_T^m(s, a_2^t) \text{ if } s \in R_2^T \\ ... \\ r_T^m(s, a_{l_T}^T) \text{ if } s \in R_{l_T}^T \end{cases}$$

4.2 Timestep $t < T$

For lengthstep $l < l_t^*$, we have:

$$\pi_I(s) = \begin{cases} a_1 \text{ if } s \in R_1 \\ \arg_{a \in A} \max[r_t(s, a) + \gamma P_t^a V_{t+1}(s)] \text{ otherwise} \end{cases}$$

Here the optimizing equation is given by:

$$\Psi_t(R_l, a_l) := \int_{R_l - G_{l-1}} ([\max_a [r_t(s, a) + \gamma P_t^a V_{t+1}(s)] \\ -[r_t(s, a_l) + \gamma P_t^a V_{t+1}^I(s)]])ds - \eta_t V(R_l - G_{l-1}) + \rho_t c(R_l)$$

Here we have $\Psi_l^{t*} := \min_{R_l, a_l} \Psi_t(R_l, a_l)$

We can derive an empirical version of the loss similar to the one in the IDTR paper as below:

$$\Psi_t(R_l, a_l) := \frac{1}{N}\left[\sum_{i=1}^N [\max_{a \in A}(r_t(X_{it}, a) + \gamma P_t^a V_{t+1}(X_{it})) - [r_t(X_{it}, a_1) + \gamma P_t^{a_1} V_{t+1}^I(X_{it})]) - \eta]\mathcal{I}(X_{it} \in S - G_{l_t - 1})]\right] + \\ \rho_T c(R_t)$$

We wish to evaluate the arg min of the above expression. In my implementation, we loop over all possible conditions by having multiple nested loops. This assumes we loop over all possible differences in each dimension. However, for the VIDTR and other papers, what one does is that we consider the co-variate order statistics (of dimension n^2) and we have $d_t(d_t + 1)/2$ choices for the other variables with evaluation being $O(n)$. We derive a $O(n^3)$ estimate to go over and evaluate all of the expressions in the above.

For the proof of the algorithm in VIDTR, they simplify the following equation:

$$\frac{1}{n}\sum_{i=1}^n \left[I(x \in G, x \in R)c_1 + I(x \in G, x \notin R)c_2 + I(x \in R, x \in G)c_3 + \rho c(R)\right]$$

In the above, we have $I(x \in G, x \in R) = I(x \in R)I(x \in G)$ and the full $I(x \in G)$ term gets pulled outside. This leaves us with the following:

$$F(\tau) = I(x \in G)[\sum_i I(x \in R)c_i' + I(x \notin R)c'']$$

For R being a monomial type thing $I(x \leq \tau)$, we see that the above is given by:

$$F(\tau) := I(x \in G)[\sum_i I(x_i \leq \tau)c_i' + I(x > \tau)c'']$$

If $[x_i^{(j)}]$ for $j \in [n]$ are the order statistics of the above, then what we notice is that:

$$F(x^{j+1}) - F(x^j) = I(x \in G)[c_j - c'']$$

Since G is fixed for each subsequent ith evaluation, we can evaluate the membership $I(x \in G)$ in $O(d)$ time and then perform the subsequent evaluations in $O(\tau)$ time which would give us $O(d+\tau)$ order of magnitude in time.

Can we generalize this approach similarly to our setting? Yes - in our setting we can see that the integrals are given by:

$$\sum_{i=1}^{n} C_i \mathcal{I}(X_i \in R_l - G_{l-1}) - \eta V(R_l - G_{l-1}) + \rho_t c(R_l)$$

In the above, if we revert back to the $1-D$ case- where we see that the expression for R is $R = (x \leq \tau)$ and G_l is empty. Then we are still in good shape. Otherwise what we observe is the following:

$$\sum_{i=1}^{n} C_i \mathcal{I}(X_i \leq \tau) - \eta(\tau - a) + c$$

We can simplify the above expression by evaluation of $F(\tau)$ for the different order statistics and the linear part is $O(1)$ in time. This would give us $O(\tau)$ evaluation time. What about the case when G is not empty, then we observe that the objective is given as:

$$\sum_{i=1}^{n} C_i \mathcal{I}[X_i \in R_l - G] - \eta V([0,\tau] - G) + \rho_t c(R_l)$$

In the above, we take the $-G$ part outside which ends up giving us:

$$\mathcal{I}[X_i \notin G] \sum_{i=1}^{n} C_i \mathcal{I}[X_i \in R_l] - \eta V([0,\tau] - G) + \rho_t c(R_l)$$

We can simplify the volume term above and by assuming that $G = \sqcup_{j=1}^{K}[a_j, b_j]$. In this case, we observe that the difference interval $[0,\tau] - G$ can be computed in atmost k steps.

$$\sum_{i=1}^{n} C_i \mathcal{I}(X_i \in R_l - G_{l-1})$$

Taking $I(X_i \notin G)$ outside, we see that the above simplifies as:

$$\mathcal{I}(X_i \notin G_{l-1}) \sum_{i=1}^{n} C_i \mathcal{I}(X_i \leq \tau)$$

The above can further be evaluated in quick time.

Furthermore we see that the volume term is given by:

$$-\eta V([0,\tau] - G)$$

If we assume that $G := \sqcup_{i=1}^{k}[a_i, b_i]$, we can see that the evaluation of the above term should take utmost $O(k)$ time.

How do the authors of the IDTR paper get the $O(nlogn)$ algorithm for $D-$space?

How do we modify the above to get a stochastic version of the algorithm? - akin to what happens when we go from gradient descent to stochastic gradient descent to stochastic batched gradient descent. What does the algorithm look like here?

$$\Psi_t(R_l, a_l) := \frac{1}{M}\Bigg[\sum_{i=1}^{M}(\max_{a\in A}(r_t(X_{it}, a) + \gamma P_t^a V_{t+1}(X_{it})) - [r_t(X_{it}, a_1) + \gamma P_t^{a_1} V_{t+1}^I(X_{it})] - \eta)\mathcal{I}(X_{it} \in S - G_{l_t-1})]\Bigg] + \rho_T c(R_T)$$

where we sample M different trajectories from the total N trajectories.

How do we solve this fast? We currently evaluate the functional for every time and lengthstep and then obtain the min vals. In the IDTR paper, for instance, they notice that we need to evaluate the operator for only certain order statistics of the error functional. Can we do better in this case?

In the above, we add the index t to indicate the regions, models, and actions at time t. In that case, we have the above function as the objective function.

The optimization equation for $l = l_{max}$ is given by:

$$\Psi_l^{t*} := \min_{a_l} \Psi_t(a_l) = \int_{S-G_{l_{\max}-1}} [\max_{a_l \in A}[(r_t(s, a) - r_t(s, a_l)] + \gamma(P_t^a V_{t+1}(s) - P_t^{a_l} V_{t+1}^I(s))]$$

The empirical error for the last length-step $l = l_{max}$ is given by:

$$\Psi_t(R_l, a_l) := \frac{1}{N}\Bigg[\sum_{i=1}^{N}[\max_{a\in A}(r_t(X_{it}, a) + \gamma P_t^a V_{t+1}(X_{it})) - [r_t(X_{it}, a_l) + \gamma P_t^{a_l} V_{t+1}^I(X_{it})]]\mathcal{I}(X_{it} \in S - G_{l_{max}-1})]\Bigg]$$

We also note the expressions for $V_t^m(s)$:

$$V_t^m(s) = \max_a [r_t^m(s, a) + \gamma P_{m,t}^a V_{t+1}^m(s)]$$

The empirical version is given by:

$$V_t^m(X_{it}) := \max_a [r_t^m(X_{it}, a) + \gamma P_t^a V_{t+1}^m(X_{it})]$$

We have the following expression for $V_t^{I,m}(s)$:

$$V_t^{I,m}(s) = \begin{cases} r_t^m(s, a_1^t) + \gamma P_{m,t}^{a_1^t} V_{t+1}^{I.m}(s) \text{ if } s \in R_1^t \\ r_t^m(s, a_2^t) + \gamma P_{m,t}^{a_2^t} V_{t+1}^{I,m}(s) \text{ if } s \in R_2^t \\ ... \\ r_t^m(s, a_{l_t}^t) + \gamma P_{m,t}^{a_{l_t}^t} V_{t+1}^{I,m}(s) \text{ if } s \in R_{l_t}^t \end{cases}$$

Empirically, we can see that the interpretable value function is given by:

$$V_t^{I,m}(X_{it}) = \begin{cases} r_t^m(X_{it}, a_1^t) + \gamma P_t^{m,a_1^t} V_{t+1}^{I,m}(X_{it}) \text{ if } X_{it} \in R_1^t \\ r_t^m(X_{it}, a_2^t) + \gamma P_t^{m,a_2^t} V_{t+1}^{I,m}(X_{it}) \text{ if } X_{it} \in R_2^t \\ ... \\ r_t^m(X_{it}, a_{l_t}^t) + \gamma P_t^{m,a_{l_t}^t} V_{t+1}^{I,m}(X_{it}) \text{ if } X_{it} \in R_{l_t}^t \end{cases}$$

When we are in the discrete setting, the integrals may be replaced by summations to yield an appropriate result. In the next subsections, we notice that the state space at every time and length step is given by a disjoint union of boxes, i.e. $S - G_{l-1} := \sqcup_{i=1}^{k_l} B_i^l$ with the knowledge that the state space is given by a disjoint union of k_l boxes at timestep t and lengthstep l. In this setting (assuming that $S - G_{l-1} := \sqcup_{i=1}^{k_{l-1}} B_i^l$), we notice that the error functional is given by the following summation.

Here the optimizing equation is given by:

$$\Psi_t(R_l, a_l) := \sum_{i=1}^{k_l} \sum_{s \in B_i} [[\max_a [r_t(s,a) + \gamma P_t^a V_{t+1}(s)] \\ -[r_t(s,a_l) + \gamma P_t^{a_l} V_{t+1}(s)]]) - \eta_t] + \rho_t c(R_l)$$

We then compute the optimal error as follows: $\Psi_l^{t*} := \min_{R_l, a_l} \Psi_t(R_l, a_l)$, where we have the arg mins giving the optimal region and action R_l^{t*}, a_l^{t*}.

The optimization equation for $l = l_{max}$ is given by:

$$\Psi_l^{t*} := \min_{a_l} \Psi_t(a_l) = \int_{S - G_{l_{\max}-1}} [\max_{a_l \in A} [[(r_t(s,a) - r_t(s,a_l)] + \gamma(P_t^a V_{t+1}(s) - P_t^{a_l} V_{t+1}^I(s))]] ds$$

If we assume that $S - G_{l_{max}-1} := \sqcup_{i=1}^{k_{l_{max}-1}} B_i^l$ Here we have the optimization equation for $l = l_{max}$ to be given by:

$$\Psi_t(R_l, a_l) := \sum_{i=1}^{k_{l_{max}-1}} \sum_{s \in B_i} [\max_a [(r_t(s,a) - r_t(s,a_l)) + \gamma(P_t^a V_{t+1}(s) - P_t^{a_l} V_{t+1}^I(s))] - \eta_t] + \rho_t c(R_l)$$

In the IDTR paper, the analysis is such that we look at the error functional when summing over only the trajectory data. In this case, we see that the error functional is given as:

In the setup above, we need not always know the transitions and rewards. In that case, we estimate these quantities from trajectory data using inverse reinforcement learning techniques or otherwise. In this case, we replace $r_t \leftrightarrow \hat{r}_t$ and $P_t \leftrightarrow \hat{P}_t$, where $\hat{P}_t$ and $\hat{r}_t$ are the estimated transitions and reward kernels from the trajectory data. We can get similar discretized equations when we replace the timestep $t \leftrightarrow T$, in this case we see that the terms from V_{t+1} onward are removed and we are left with a similar sum.

4.3 Algorithm

In this section, we describe the pseudocode for the implemented algorithms.

4.3.1 Integrating over the general measure over the state space

Algorithm 1: Value Interpretable Dynamic Treatment Regimes

Input: $S, A, T, \gamma, (P_t)_{t=1}^T, (r_t)_{t=1}^T, (l_t^*)_{t=1}^T, \eta, \rho, \kappa, st, M$
Output: $(\pi_t^I)_{t=1}^T$, $T\epsilon$
bounds $\leftarrow$ **bounds**(S);
$R \leftarrow S$;
$T\epsilon \leftarrow 0$;
$\mathcal{C} \leftarrow$ **conditions**(S);
for $t \in \{T-1, T-2, \ldots, 0\}$ **do**
  for $l \in [1, l_t^* - 1]$ **do**
    $\epsilon_{\min} \leftarrow \infty$;
    $c^* \leftarrow$ None;
    $a^* \leftarrow$ None;
    $M \leftarrow \min\{M, |\mathcal{C}|\}$;
    for $i \in [M]$ **do**
      $\epsilon \leftarrow 0$;
      $C[i] \leftarrow i$-th condition in $\mathcal{C}$;
      for $a \in A$ **do**
        $bf(s) \leftarrow \max_a(r_t(s,a) + \gamma P_t^a V_{t+1}(s)) - (r_t(s,a) + \gamma P_t V_{t+1}(s))$;
        $\epsilon \leftarrow \int_{C[i]}(bf(s) + \eta_t)\, ds + \rho_t \cdot \text{Complexity}(C[i])$;
        if $\epsilon < \epsilon_{\min}$ **then**
          $c^* \leftarrow C[i]$;
          $a^* \leftarrow a$;
          $\epsilon_{\min} \leftarrow \epsilon$;
    $R \leftarrow R - c^*$;
    $\mathcal{C} \leftarrow \mathcal{C} - c^*$;
    $T\epsilon \leftarrow T\epsilon + \epsilon_{\min}$;
    $\pi_t^I|_{c^* \cap R} \leftarrow a^*$;

Algorithm 2: Value Interpretable Dynamic Treatment Regime - Last lengthstep

while $t \in [T-1, T-2, ..., 0]$ **do**
  $l = l_t^*$;
  $\epsilon_{min} := \infty, a^* := None$;
  while $a \in A$ **do**
    $bf(s) = \max_a(r_t(s,a) + \gamma P_t^a V_{t+1}(s))$ - $(r_t(s,a) + \gamma P_t V_{t+1}(s))$;
    $\epsilon = \int_R (bf(s) + \eta_t) ds$;
    if $\epsilon < \epsilon_{min}$ **then**
      $a^* = a, \epsilon_{min} = \epsilon$;
    $T\epsilon += \epsilon_{min}; \pi_t^I|_R := a^*$;

4.4 Implementation

We provide the code to implement this method. We note that to perform the integration step in the algorithm, we compute the regions over which we integrate. Assuming that we start out with a box to perform the initial integration, we notice that when we remove or add conditions, we are left either with a box or a disjoint union of boxes (DBU). We create a data structure of the same name and store this structure using the centers and lengths for the same. For this data structure, we ensure that it has the following functionalities:

Algorithm 3: Optimal policy computation - Value Iteration

Data: $S, A, T, \gamma, (P_t)_{t=1}^T, (r_t)_{t=1}^T$
Output: $(\pi_t)_{t=1}^T, (V_t)_{t=1}^T$
begin
  $\pi_t(s) \leftarrow a_0 \forall t \in [0,T], \forall s \in S$;
  $V_t(s) \leftarrow 0 \forall t \in [0,T], \forall s \in S$;
  while $t \in [0,T] \cap \mathcal{Z}$ **do**
    while $s \in S$ **do**
      $\bar{v} = \inf$;
      while $a \in A$ **do**
        $\texttt{b\_val} = r_t(s,a) + \gamma P_t^a V_{t+1}(s)$;
        if $\texttt{b\_val} \geq \bar{v}$ **then**
          $\bar{v} = \texttt{b\_val}$;
          $\bar{a} = a$;
          $\pi_t(s) = \bar{a}$;
          $V_t(s) = \bar{v}$

1. **Subtraction of DBUs**: This works by performing disjoint unions over subtracted rectangles. Given rectangles $A_1, B_1 \in \mathbb{R}^{d_1}$ and rectangles $A, B \in \mathbb{R}^{d_2}$. On a rectangle level the logic we employ is the following:

$$(A_1 \times A) - (B_1 \times B) := [(A_1 - B_1) \times A] \sqcup [(A_1 \cap B_1) \times (A - B)]$$

   We perform the above operation until we reach the dimension of 1, here we can manually compute the subtraction of two intervals.

2. **Union of DBUs**: This works by performing the following logic on the level of rectangles:

$$A \cup B := A \sqcup (B - A)$$

   The subtraction of two rectangles can be performed as above.

3. **Intersection of DBUs**: This works by performing a disjoint union over the set of all the rectangles that we take a union of. The logic on the level of rectangles in this scenario is given by:

$$(A \times A_1) \cap (B \times B_1) := (A \cap B) \times (A_1 \cap B_1)$$

4. **Integration of a function**: Given a function f, we perform the Riemannian integration of the function in the DBU. Given a $DBU := A_1 \sqcup A_2 ... \sqcup A_n$. This is given by:

$$\int_{DBU} f := \sum_{i=1}^{n} \int_{A_i} f$$

   The integral of f over a box is the discrete Riemannian sum of the function. We are given stepsizes $(s_d)_{d=1}^D$ and we sum up the function by spacing points on the grid using the given stepsizes. This is done by performing the sum of f with points spaced out by s_d in the corresponding dth dimension.

4.5 Empirical estimation of the rewards and transitions

We empirically estimate the transitions and rewards for an MDP. In this algorithm, we compute the transition and reward probabilities as empirical estimates of what is observed in the trajectory

Algorithm 4: Empirical estimation of the transitions and the rewards

Data: $(s_t^i)_{t=1}^T, (a_t^i)_{t=1}^T, (r_t^i)_{t=1}^T \forall i \in [1, N]$
Output: $(P_t)_{t=1}^T, (r_t)_{t=1}^T$: The estimated transitions and rewards
begin
  $T(s'|s,a) = 0 \forall s, s' \in S, a \in A$;
  $O(s,a) = 0 \forall s \in S, a \in A$;
  $T_r(s,a) = 0 \forall s, s' \in S, a \in A$;
  $P_{emp}(s'|s,a) = 0 \forall s, s' \in S, a \in A$;
  $r_{emp}(s,a) = 0 \forall s \in S, a \in A$;
  while $t \in [T-1]$ **do**
    while $i \in [1, N]$ **do**
      $T((s_{t+1}^i)|(s_t^i),(a_t^i)) += 1$;
      $T_r((s_t^i),(a_t^i)) += r_t^i$;
      $O((s_t^i),(a_t^i)) += 1$
  while $t \in [T-1]$ **do**
    while $i \in [1, N]$ **do**
      $P_{emp}((s^i)_{t+1}|(s^i)_t,(a^i)_t) = T((s^i)_{t+1}|(s^i)_t,(a^i)_t)/O((s^i)_t,(a^i)_t)$;
      $r_{emp}(s_t^i, a_t^i) = T_r(s_t^i, a_t^i)/O((s_t^i),(a_t^i))$;

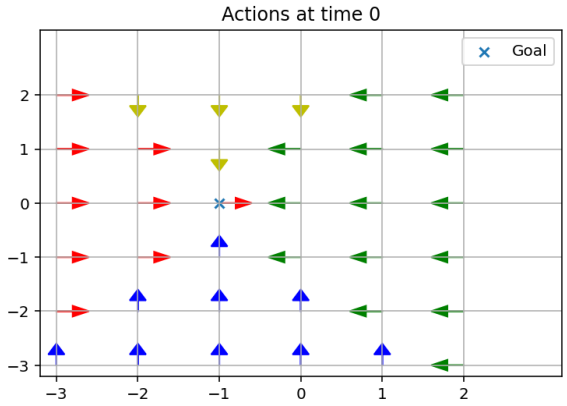
data. The algorithm works by scanning the trajectories while storing the empirical estimates of the transition probabilities and the rewards.

An issue with using the empirical estimation to compute the transitions and the rewards is that we would need to discretize the state space beforehand. Since the data we have from the IDTR paper are continuous data, we need an additional step of preprocessing to get the algorithm to work. We do this by employing the following procedure - given a point in the DBU, we find a closest point in the DBU.

5 Grid Environment - VIDTR

For the grid environment where the agent needs to reach the goal denoted by X, we demonstrate the effectiveness of the VIDTR. At each coordinate point the arrow mark denotes the direction the agent takes to reach the goal. We assume the environmental parameters are known to the agent, and compute the optimal policies and rewards using a value iteration approach. The optimal policies are given by:

Figure 1: Optimal policies - VIDTR for all $t \in [4]$



In this setting, we see that the optimal interpretable policies obtained by the VIDTR look like:

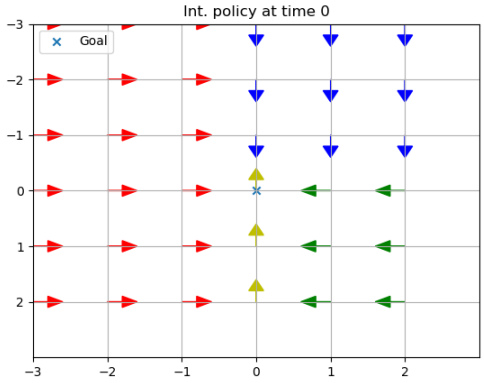


Figure 2: Time $T = 0$ for the VIDTR

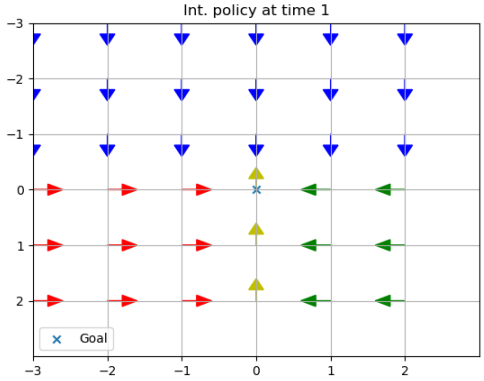


Figure 3: Time $T = 1$ for the VIDTR

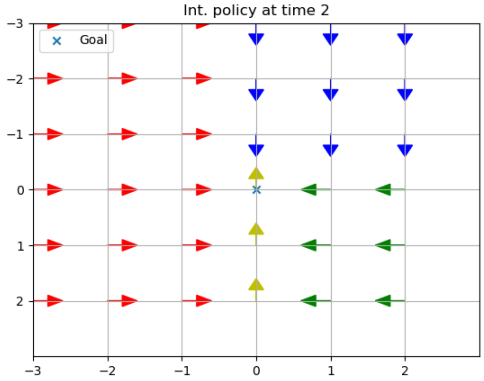


Figure 4: Time $T = 2$ for the VIDTR

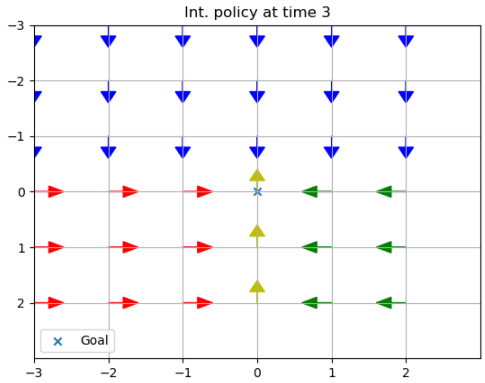


Figure 5: Time $T = 2$ for the VIDTR

6 Theoretical Results

In this section we establish the theoretical results that establish the convergence of the value interpretable dynamic treatment regimes method.

6.1 Bounds on the errors

Assumption 1. *We list the assumptions that we use to prove the error bounds:*

1. *Lipschitz continuity of rewards in the second coordinate:* $|r_t(s,b) - r_t(s,a)| \leq C_r d_A(a,b)\ \forall\ t \in [0,T] \cap \mathbb{Z}$
2. *Distance between actions is bounded:* $d_A(a,b) \leq D$
3. *Smoothness in the kernels:* $|P_t(s'|s,a) - P_t(s'|s,b)| \leq C_p d_A(a,b)\ \forall t \in [0,T] \cap \mathbb{Z}$
4. *Bounds on the value functions:* $V_{t+1}(s) \leq W_{t+1}$
5. $S = \prod_{i=1}^{N}(b_i, a_i)$*, here we assume* $|b_i - a_i| \leq S^{1/d}$

For each t, we measure the cost of interpretability to be the sum of the quantities we aim to minimize, which is the adjusted value metric. Here for different timesteps t it is given by:

$$C_t = \sum_{l=1}^{l_{max}} \Psi_{tl}^*$$

So the final cost we obtain is given by:

$$C = \sum_{t=1}^{T} C_t$$

Recall that for time T and lengthstep 1, we have:

$$\Psi_{T1}^* = \min_{R,a_1} \int_R [\max_a(r_T(s,a)) - r_T(s,a_1)]ds - \eta_T V(R) + \rho_T c(R_1)$$

We have the following to be true:

$$|r_T(s,a^*(s)) - r_T(s,a_1)| \leq C_r d_A(a^*(s),a_1)$$

Assuming the rewards are Lipschitz and that the action space is bounded we get the below inequality:

$$\Psi^*_{T1} \leq \min_{R_1}[|R|(C_r D - \eta) + \rho_T c(R_1)]$$

We have that $G_l = \cup_{i=1}^{l} R_i$ this would imply for a general l, we get the following equation:

$$\Psi^*_{Tl} \leq \min_{R_l}(|R_l - G_l|(C_r D - \eta) + \rho c(R_l))$$

For $l = l_{max}$, we get:

$$\Psi^*_{Tl_{max}} \leq C_r D|S - G_{l_{max}-1}|$$

The total error would be the sum of ([6.1] + [6.1] (for different lengths l) + [6.1]).

For a general $t = t$ and $l = 1$, we have that the error is given by:

$$\Psi^*_{t1} \leq \int_R [\max_a(r_t(s,a) + \gamma P^a_t V_{t+1}(s)) - (r_t(s,a) + \gamma P^a_t V_{t+1}(s))]ds - \eta_t V(R) + \rho_t c(R)$$

If $a^*(s) := \max_a(r_t(s,a) + \gamma P^a_t V_{t+1}(s))$, then the following is true:

$$\Psi^*_{t1} \leq \int_R [r_t(s,a^*(s)) + \gamma P^{a^*(s)}_t V^*_t(s) - r_t(s,a) - \gamma P^a_t V_{t+1}(s)]ds - \eta_t V(R) + \rho_t c(R)$$

We can then split the rewards term as:

$$[r_t(s,a^*(s)) - r_t(s,a)] \leq C_r D d_A(a^*, a)$$

We then see that we can bound the kernel term using a triangle inequality, this works as:

$$(P^{a^*(s)}_t V^*_{t+1}(s) - P^a_t V_{t+1}(s)) \leq (P^{a^*(s)}_t V^*_{t+1}(s) - P^a_t V^*_{t+1}(s) + P^a_t V^*_{t+1}(s) - P^a_t V_{t+1}(s))$$

The above then simplifies as follows:

$$|P^{a^*(s)}_t - P^a_t| V^*_{t+1}(s) \leq C_P d_A(a^*(s), a) W_{t+1}$$

We can simplify the other term:

$$|P^a_t V^*_{t+1}(s) - P^a_t V_{t+1}(s)| \leq |P^a_t||V^*_{t+1}(s) - V_{t+1}(s)|$$

We can the bound the kernel as $|P^a_t| \leq 1$. This gives us the below inequality:

$$\Psi^*_{t1} \leq \min_{R_1}(C_r D|R_1| + \gamma C_p D W_{t+1}|R_1| + \int_{R_1} |V_{t+1}(s') - V^I_{t+1}(s')|ds' - \eta|R_1| + \rho_t c(R_1))$$

We can simplify the computation of $\int_{R_1}[V_{t+1}(s') - V^I_{t+1}(s')]ds'$ for different values of t. We notice that for $t = T$, we have $V^*_T(s) := \max_a(r_T(s,a))$, and that $V^I_T(s) := r_T(s,a_i)$ for $s \in R_i$. This implies that

$$|V^*_T(s) - V^I_T(s)| \leq \max_a(r_T(s,a)) - r_T(s,a_i))$$

As done previously the above can be bounded by the Lipschitz constant on the rewards and the distance between the actions. When the above gets integrated over a region R, we see:

$$\int_R |V^*_T(s) - V^I_T(s)| \leq C_r D|R|$$

We now simplify the calculation of $\int_R |V_t^*(s) - V_t^I(s)|ds$ in terms of the future timesteps. This gives us the following:

$$|(V_t^* - V_t^I)(s)| \leq \max_a (r_t(s,a) + \gamma P_t^a V_{t+1}(s)) + \gamma (P_t^a V_{t+1}^*(s) - P_t^{a_i} V_{t+1}^I(s))$$

The rewards term can now be bouded by $C_r D$ and we see that the kernel terms split as:

$$|(V_t^* - V_t^I)(s)| \leq C_r D + \gamma (P_t^a V_{t+1}^*(s) - P_t^{a_i} V_{t+1}^I(s))$$

We add and subtract $P_t^{a_i} V_{t+1}^*$ to the above transition terms to get:

$$\leq C_r D + \gamma (P_t^a V_{t+1}^* - P_t^{a_i} V_{t+1}^* + P_t^{a_i} V_{t+1}^* - P_t^{a_i} V_{t+1}^I)$$

We can simplify the above as follows:

$$\leq C_r D + \gamma [(P_t^a - P_t^{a_i}) V_{t+1}^* + P_t^{a_i} (V_{t+1}^* - V_{t+1}^I)]$$

The above is then simplified using the fact that the norm of the transitions are bounded by 1.

$$\leq C_r D + 2\gamma ||V_{t+1}^*|| + \gamma ||V_{t+1}^* - V_{t+1}^I||$$

If we assume that $d_t := ||V_t^*(s) - V_{t+1}^I(s)||$, we then get the following recursive inequality:

$$d_t \leq C_r D + 2\gamma ||V_{t+1}^*|| + \gamma d_{t+1}$$

Before solving the above recursively, we establish bounds on $||V_t^*||$.

$$V_t(s) = \max_a (r_t(s,a) + \gamma P_t^a V_{t+1}(s))$$

We can simplify the above as follows:

$$||V_t(s)|| \leq ||r_t|| + \gamma ||V_{t+1}(s)||$$

We also have that $||V_T(s)|| \leq \bar{r}$. Then we can see that the following is true:

$$||V_{T-1}(s)|| \leq \bar{r} + \gamma \bar{r}$$

We make the claim that :

$$||V_t(s)|| \leq \bar{r}[1 + \gamma + \gamma^2 + ... + \gamma^{T-t}]$$

This gives us:

$$||V_t(s)|| \leq \frac{1 - \gamma^{T-t+1}}{1 - \gamma} \bar{r}$$

Returning back to the simplification of d_t, we see that for $t = T$:

$$|V_T^* - V_T^I|(s) \leq C_r D$$

We also have $||V_T(s)|| \leq \bar{r}$.
For $t = T - 1$, we have the following:

$$||V_{T-1} - V_{T-1}^I||(s) \leq C_r D(1 + \gamma) + 2\gamma \bar{r}$$

For $t = T - 2$, we have:

$$d_{T-2} \leq (C_r D + 2\gamma \bar{r})(1 + \gamma + \gamma^2) + 2\gamma^2 \bar{r}(1 + \gamma)$$

We then make the following lemma, which we show inductively:

$$d_{T-t} \leq (C_r D + 2\gamma\bar{r})(1 + \gamma + ... + \gamma^t) + 2\gamma^2\bar{r}(1 + 2\gamma + 3\gamma^2 + ... + (t-1)\gamma^{t-2} + (t-1)\gamma^{t-1})$$

We show that the above holds by inserting the recursion equation 6.1. This gives us:

$$\begin{aligned} d_{T-t-1} \leq C_r D + 2\gamma\bar{r}\frac{1-\gamma^{t+2}}{1-\gamma} + \\ +(C_r D + 2\gamma\bar{r})[\gamma + \gamma^2 + ... + \gamma^{t+1}] + \\ 2\gamma^3\bar{r}(1 + 2\gamma + 3\gamma^2 + ... + (t-1)\gamma^{t-2} + (t-1)\gamma^{t-1}) \end{aligned}$$

The above is then simplified as follows:

$$\begin{aligned} d_{T-t-1} \leq (C_r D + 2\gamma\bar{r})[1 + \gamma + ... + \gamma^{t+1}] + \\ 2\gamma\bar{r}(\gamma + \gamma^2 + ... + \gamma^{t+1}) + \\ 2\gamma^3\bar{r}(1 + 2\gamma + 3\gamma^2 + ... + (t-1)\gamma^{t-2} + (t-1)\gamma^{t-1}) \end{aligned}$$

By taking γ inside the above is then simplified as:

$$\begin{aligned} d_{T-t-1} \leq (C_r D + 2\gamma\bar{r})[1 + \gamma + ... + \gamma^{t+1}] \\ +2\gamma^2\bar{r}[1 + \gamma + \gamma^2 + ... + \gamma^t] \\ +2\gamma^2\bar{r}(\gamma + 2\gamma^2 + ... + (t-1)\gamma^{t-1} + (t-1)\gamma^t) \end{aligned}$$

We can simplify the above as follows:

$$\begin{aligned} d_{T-t-1} \leq (C_r D + 2\gamma\bar{r})[1 + \gamma + ... + \gamma^{t+1}] + \\ 2\gamma^2\bar{r}(1 + 2\gamma + 3\gamma^2 + ... + t\gamma^{t-1} + t\gamma^t) \end{aligned}$$

We now list how we simplify $(1 + 2\gamma + 3\gamma^2 + ... + t\gamma^{t-1} + t\gamma^t)$. We define the following:

$$S_{t-1} := 1 + 2\gamma + 3\gamma^2 + ... + (t-1)\gamma^{t-2}$$

We then have that:

$$\gamma S_{t-1} := \gamma + 2\gamma^2 + 3\gamma^3 + ... + (t-1)\gamma^{t-1}$$

We then get that 6.1 - 6.1 is given by:

$$S_{t-1}(1-\gamma) = 1 + \gamma + ... + \gamma^{t-2} - (t-1)\gamma^{t-1}$$

This then simplifies as:

$$S_{t-1} = \frac{1-\gamma^{t-1}}{(1-\gamma)^2} - \frac{(t-1)\gamma^{t-1}(1-\gamma)}{(1-\gamma)^2}$$

The above can then be simplified as:

$$S_{t-1} := \frac{1 - \gamma^{t-1}(-t\gamma^{t-1} + \gamma^{t-1})(1-\gamma)}{(1-\gamma)^2}$$

We can then simplify the above as:

$$S_{t-1} = \frac{1 - \gamma^{t-1} - t\gamma^{t-1} + \gamma^{t-1} - t\gamma^t - \gamma^t}{(1-\gamma)^2}$$

This is then simplified as:

$$S_{t-1} = \frac{1-\gamma^{t-1}[\gamma + t(1+\gamma)]}{(1-\gamma)^2}$$

The above can be simplified as:

$$d_{T-t-1} \leq (C_r D + 2\gamma\bar{r})\frac{1-\gamma^{t+2}}{1-\gamma} + \\ 2\gamma^2\bar{r}[\frac{1-\gamma^{t-1}[\gamma + t(1+\gamma)]}{(1-\gamma)^2} - (t-1)\gamma^{t-1}]$$

We recall the following bounds for W_t from above:

$$W_t \leq \frac{1-\gamma^{T-t+1}}{1-\gamma}\bar{r}$$

Recall how the errors were computed below:

$$\Psi^*_{t1} \leq \min_{R_1}(C_r D|R_1| + \gamma C_p D W_{t+1}|R_1| + \int_{R_1} |V_{t+1}(s') - V^I_{t+1}(s')|ds' - \eta|R_1| + \rho_t c(R_1))$$

In the above equation, we insert the values of W_t and bounding $|V_{t+1}(s) - V^I_{t+1}(s)|$, we see that:

$$\Psi^*_{t1} \leq \min_{R_1}(C_r D|R_1| + \gamma C_p D\frac{1-\gamma^{T-t}}{1-\gamma}\bar{r}|R_1| + \\ (C_r D + 2\gamma\bar{r})\frac{1-\gamma^{T-t}}{1-\gamma}|R_1| + \\ 2\gamma^2\bar{r}\frac{1-\gamma^{T-t-3}[\gamma^3 - 1(T-t-2)(1+\gamma^2(1+\gamma))]}{(1-\gamma)^2}|R_1|) + \\ -\eta|R_1| + \rho_t c(R_1)$$

For different time-steps t and length-steps l_t, we see that the optimizing function Ψ^*_{tl}. We assume that $G_{t,l-1} = \cup_{j=1}^{l-1} R_{tj}$. This implies:

$$\Psi^*_{tl} \leq \min_{R_l}(C_r D|R_l - G_{t,l-1}| + \gamma C_p D\frac{1-\gamma^{T-t}}{1-\gamma}\bar{r}|R_l - G_{t,l-1}| + \\ (C_r D + 2\gamma\bar{r})\frac{1-\gamma^{T-t}}{1-\gamma}|R_l - G_{t,l-1}| + \\ 2\gamma^2\bar{r}\frac{1-\gamma^{T-t-3}[\gamma^3 - 1(T-t-2)(1+\gamma^2(1+\gamma))]}{(1-\gamma)^2}|R_l - G_{t,l-1}|) + \\ -\eta|R_l - G_{t,l-1}| + \rho_t c(R_l)$$

Theorem 1. *The optimization function below where we have $S = \Pi_{i=1}^{D}[a_i, b_i]$ and choose R from one of the following list sets with k leaves. The sets are:*

$$R = \{S, \emptyset, \{x|\exists j \in [N],\ s.t\ x_j \leq c_j\}, \\ \{x|\exists j,k\ s.t\ x_j \leq c_j, x_k \leq c_k\}, ..., \{x|\exists j_1, .., j_k \in [N],\ s.t\ x_{j_i} \leq c_{j_i} \forall i \in [K]\}\}$$

If $\rho > 0$, we observe the following:

$$min_R[|R|A_t + \rho c(R)]$$

The problem [1] is upper bounded by:

1. $A_t > 0 \implies$ *We choose* $R = \emptyset$*, which gives us a* 0 *valued objective.*
2. $A_t < 0 \implies$ *We choose* $R = S$*, which then gives us* $c(R) = 0$ *and we see that the objective is bounded by* $A_t|S|$*.*

How do we modify the above theorem so that we can compute convergence bounds for the same?

From the above, we see that we can bound $\Psi^*_{tl} \leq |A_t||R_l - G_{t,l-1}|$, adding up the errors for different length steps l we see that we can bound the error at time step t with $\Psi^*_t \leq |A||S|$. How do we further simplify these bounds? Right now the bounds are a bit coarse. The total error would then be less than or equal to $|A||S|T$.

Theorem 2. *We claim that:*

$$Y_{m.t} \geq 0 \forall m, \forall t$$

And we also have the following to be true:

$$V_t(s) \geq V^I_t(s) \forall t > 0, \forall s \in S$$

We show this inductively; for instance, when we have:

Proof. For $t = T$ we have the following to be true:

$$Y_{m,T}(s, a) := \max_{\alpha \in \mathcal{A}} r^m_T(s, \alpha) - r^m_T(s, a)$$

In the above, we have that $Y_{m,T}(s, a) \geq 0$, for the value terms in the last timestep T, we see that the following holds true:

$$V_T(s) := \max_{\alpha}[r_T(s, \alpha)]$$

We then see that:

$$V^I_T(s) := r_T(s, a_i) \text{ if } s \in R_i$$

for appropriately defined a_i and R_i.
Assume inductively that $V_t(s) \geq V^I_t(s)$, $\forall s \in S, \forall t \geq t_0 - 1$. In this case, what we observe is that:

$$Y_{m,t}(s, a) := \max_{\alpha \in \mathcal{A}}[r_t(s, \alpha) + \gamma P^\alpha_t V_{t+1}(s)] - [r_t(s, a) + \gamma P^a_t V^I_{t+1}(s)]$$

To the above, we then have:

$$\max_{\alpha \in \mathcal{A}}[r_t(s, \alpha) + \gamma P^\alpha_t V_{t+1}(s)] \geq \max_{\alpha \in \mathcal{A}}[r_t(s, \alpha) + \gamma P^\alpha_t V^I_{t+1}(s)]$$

Thus we can see that:

$$\max_{\alpha \in \mathcal{A}}[r_t(s, \alpha) + \gamma P^\alpha_t V^I_{t+1}(s)] \geq [r_t(s, a) + \gamma P^a_t V^I_{t+1}(s)]$$

In the end, we have $Y_{m,t}(s, a) \geq 0$. We inductively extend the proof for the value functions as fol.....

$$V_t(s) := \max_{\alpha \in \mathcal{A}}[r_t(s, \alpha) + \gamma P^\alpha_t V_{t+1}(s)]$$

For the case of interpretable value functions, we observe:

$$V^I_t(s) := r_t(s, a_i) + \gamma P^{a_i}_t V^I_{t+1}(s)$$

To the equation 6.1, we modify the second term to get the following to be true:

$$\gamma P^\alpha_t V_{t+1}(s) \geq \gamma P^\alpha_t V^I_{t+1}(s)$$

This gives us the following to be true:

$$V_t(s) - V^I_t(s) \geq \max_{\alpha \in \mathcal{A}}[r_t(s, \alpha) + \gamma P^\alpha_t V_{t+1}(s)] \geq \max_{\alpha \in \mathcal{A}}[r_t(s, \alpha) + \gamma P^\alpha_t V^I_{t+1}(s)] \geq r_t(s, a_i) + \gamma P^{a_i}_t V^I_{t+1}(s)$$

□

7 Algorithm

In this section, we describe the pseudocode for the algorithms implemented.

Algorithm 5: Value Interpretable Dynamic Treatment Regimes (VIDTR)

Input: $S, A, T, \gamma, (P_t)_{t=1}^{T}, (r_t)_{t=1}^{T}, (l_t^*)_{t=1}^{T}, \eta, \rho, \kappa, st, M$
Output: $\pi_t^I,\ T\epsilon$
bounds $\leftarrow$ **bounds**(S);
$R \leftarrow S,\ T\epsilon \leftarrow 0,\ \mathcal{C} \leftarrow$ **conditions**(S);
for $t = T-1$ **to** 0 **do**
  for $l = 1$ **to** $l_t^* - 1$ **do**
    $\epsilon_{\min} \leftarrow \infty,\ c^* \leftarrow \varnothing,\ a^* \leftarrow \varnothing$;
    $M \leftarrow \min\{M, |\mathcal{C}|\}$;
    for $i = 1$ **to** M **do**
      $\epsilon \leftarrow 0$;
      $C[i] \leftarrow i$-th condition in $\mathcal{C}$;
      foreach $a \in A$ **do**
        $bf(s) \leftarrow \max_{a'}\Big(r_t(s,a') + \gamma P_t^{a'} V_{t+1}(s)\Big) - (r_t(s,a) + \gamma P_t^{a} V_{t+1}(s))$;
        $\epsilon \leftarrow \int_{C[i]} (bf(s) + \eta_t)\, ds + \rho_t \cdot \text{Complexity}(C[i])$;
        if $\epsilon < \epsilon_{\min}$ **then**
          $c^* \leftarrow C[i],\ a^* \leftarrow a,\ \epsilon_{\min} \leftarrow \epsilon$;
    $R \leftarrow R - c^*,\ \mathcal{C} \leftarrow \mathcal{C} - c^*$;
    $T\epsilon \leftarrow T\epsilon + \epsilon_{\min}$;
    $\pi_t^I|_{c^* \cap R} \leftarrow a^*$;

In the step where the maximum computation is performed, improvement is required. Currently, we calculate the objective for each condition R and each action a, this requires us to iterate over each and every condition R. We find an approximation as follows : assume that we know the optimal action a_t^* and condition R_t^* at time t. Can we find R_{t+1}^* and a_{t+1}^* from this information? It turns out that when the transitions and rewards are close to each other in time, the optimal regions and actions are close to each other.

8 Evolutionary Algorithms

We recall an optimization procedure from the literature on soft computing termed as *genetic algorithms*. This works in the following manner, given a population of genes, and a fitness function $\mathcal{F}$ over this set, we aim to find the gene that maximizes the fitness value. The steps that are used in this procedure are:

1. Initialization : Randomly pick a set of N genes from the population.
2. Fitness evaluation : Evaluate the fitness of each member from the N genes and give the picked genes a probability measure based on this weighting.
3. Crossover recombination : From the picked genes with a probability measure, pick two genes which we use for crossover.
4. Mutation : When the children from the crossover are obtained, a mutation is performed.
5. Population growth : Return to the fitness evaluation step to continue the cycle for T timesteps.

Here we propose a method to perform the optimization of the underlying subproblem based on the usage of evolutionary algorithms. We perform optimization at each subproblem for the different time and lengthsteps. We aim to find the minimizer region and action for the following problem:

$$R \mapsto \int_{R-S} (\max_{\alpha}[r_t(s,\alpha) + \gamma P_t^{\alpha} V_{t+1}(s)] - [r_t(s,a) + \gamma P_t^{a} V_{t+1}(s)] - \eta) ds + \rho c(R)$$

We assume the regions are chosen from a space with dimension D, and we choose R as a product of subintervals $[a_{i1}, b_{i1}] \times [a_{i2}, b_{i2}] \times ...[a_{ik}, b_{ik}]$. We assume to know the bounds and stepsizes for each dimension.
The initialization phase proceeds by picking indices $[i_1, i_2, ..., i_k]$ where $i_t \in [d]$. We then pick $a_{it} < b_{it}$ from bounds$[t]$ by picking a number randomly in $[a_{it}, a_{it} + stepsizes[t], ..., b_{it}]$.
The fitness function in this setting is given by the integral of the adjusted Bellman equation over the region minus the remaining space; this looks like

$$R \mapsto \int_{R-S} (\max_{\alpha}[r_t(s,\alpha) + \gamma P_t^{\alpha} V_{t+1}(s)] - [r_t(s,a) + \gamma P_t^{a} V_{t+1}(s)] - \eta) ds + \rho c(R)$$

Further let us assume $\propto$ denotes the crossover operator.
Given 2 conditions, we pick a random index in each condition say r_1 and r_2, and we switch the condition at that index between the two conditions.

$$[(a_{i_1}, b_{i_1}) \times (a_{i_2}, b_{i_2}) \times (a_{i_3}, b_{i_3}) ... \times (a_{i_k}, b_{i_k})] \propto [(a_{j_1}, b_{j_1}) \times (a_{j_2}, b_{j_2}) \times (a_{j_3}, b_{j_3}) ... \times (a_{j_l}, b_{j_l})] :=$$

$$[(a_{i_1}, b_{i_1}) \times (a_{i_2}, b_{i_2}) \times ... \times (a_{j_{r_2}}, b_{j_{r_2}}) ... \times (a_{i_k}, b_{i_k})] \times [(a_{j_1}, b_{j_1}) \times (a_{j_2}, b_{j_2}) \times ... (a_{i_{r_1}}, b_{i_{r_1}}) ... \times (a_{j_l}, b_{j_l})]$$

Here we pick index r_2 in the second vector and we pick index r_1 in the first vector. We then substitute the conditions occuring at the conditions.
The fitness of a condition $R := [(a_{j_1}, b_{j_1}) \times (a_{j_2}, b_{j_2}) \times ... \times (a_{i_{r_1}}, b_{i_{r_1}}) \times ... \times (a_{j_l}, b_{j_l})]$ is given by the corresponding integral:

$$\int_R [(\max_{\alpha}[r_t(s,\alpha) + P_t^{\alpha} V_{t+1}(s)] - [r_t(s,a) + \gamma P_t^{a} V_{t+1}(s)] - \eta] ds$$

A mutation could involve choosing a random index in the condition and adding noise to the lower or upper values of the condition present in that index.

$$\mathcal{M}[(a_{i_1}, b_{i_1}) \times (a_{i_2}, b_{i_2}) \times (a_{i_3}, b_{i_3}) ... \times (a_{i_k}, b_{i_k})] :=$$
$$[(a_{i_1}, b_{i_1}) \times (a_{i_2}, b_{i_2}) \times (a_{i_3}, b_{i_3}) ... \times (a_{i_r} + \epsilon_1, b_{i_r} + \epsilon_2) \times ... \times (a_{i_k}, b_{i_k})]$$

In the above, we have that either ϵ_1 or ϵ_2 is 0.
Another idea would be to optimize directly over the space of list-based policies. Here we need to introduce a corresponding crossover and a mutation operator. Given π_1 and π_2, two list based policies that are as follows:

$$\pi_1(s) = \begin{cases} a_1, & \text{if } s \in R_1, \\ a_2, & \text{if } s \in R_2, \\ \vdots & \\ a_{d_1}, & \text{if } s \in R_{d_1}, \end{cases} \qquad \pi_2(s) = \begin{cases} b_1, & \text{if } s \in Q_1, \\ b_2, & \text{if } s \in Q_2, \\ \vdots & \\ b_{d_2}, & \text{if } s \in Q_{d_2}. \end{cases}$$

The crossover operation in the following would be to pick 2 indices $r_1 \in [d_1]$ and $r_2 \in [d_2]$, we would then copy and paste whatever occurs in each of the policy after the picked index into the other. This

would give us the following modified policies:

$$\hat{\pi}_1(s) = \begin{cases} a_1, & \text{if } s \in R_1, \\ a_2, & \text{if } s \in R_2, \\ \vdots & \\ a_{r_2-1}, & \text{if } s \in R_{r_2-1} \\ b_{r_2}, & \text{if } s \in Q_{r_2} \\ b_{r_2+1}, & \text{if } s \in Q_{r_2+1} \\ \vdots & \\ b_{d_2}, & \text{if } s \in Q_{d_2}, \end{cases} \qquad \hat{\pi}_2(s) = \begin{cases} b_1, & \text{if } s \in Q_1, \\ b_2, & \text{if } s \in Q_2, \\ \vdots & \\ b_{r_1-1}, & \text{if } s \in Q_{r_1-1} \\ a_{r_1}, & \text{if } s \in R_{r_1} \\ a_{r_1+1}, & \text{if } s \in R_{r_1} \\ \vdots & \\ a_{d_1}, & \text{if } s \in R_d \end{cases}$$

The mutation operator in the above is given as follows; one would pick a random index $i_1 \in [d_1]$ and modify the action chosen in this condition $a_{i_1} \leftrightarrow a_r$ where r is a random index chosen in $[d_1]$. The fitness value in this case is given by integration of the adjusted Bellman equation with actions and regions as dictated from the policy.

$$\mathcal{F}[\pi] := \int_{R_i - \cup_{j<i} R_j} [\max_{\alpha} [r_t(s,\alpha) + \gamma P_t^{\alpha} V_{t+1}(s)] - [r_t(s,\pi(s)) + \gamma P_t^{\pi(s)} V_{t+1}(s)] - \eta] ds + \rho c(R_i)$$

9 Subproblem MIO formulation

Here we present a mixed-integer formulation for the VIDTR local optimization problem. It looks like:

$$\min_{z_i, a_j < b_j} \sum_{i=1}^{N} z_i U_{iat}^l$$

Here we have the following constraints:

$$x_{ij} \geq a_j - M(1 - z_i)$$

and the other constraint being:

$$x_{ij} \leq b_j + M(1 - z_i)$$

In the above, we have $z_i = 1$ which denotes the event that i belongs to the chosen box. We have $-C < a_j < b_j < C$. In that case, we can choose $M = C$. One approach would be to simplify the following MIO problem and obtain solutions for other simplified problems which could work as an upper bound to the problem in our scenario.
The simplification here would be to order the X_i values in one dimension and perform the branch-and-bound procedure using this enumeration. In order to simplify the problem in 9, we have the following coefficient values:

$$U_{iat}^l := [\max_{\alpha} [r_t(X_{it},\alpha) + \gamma P_t^{\alpha} V_{t+1}(X_{it})] - [r_t(X_{it},a) + \gamma P_t^{a} V_{t+1}^I(X_{it})] - \eta_t] \mathcal{I}[X_{it} \notin G_t^{l-1}]$$

Here we have:

$$z_i := \mathcal{I}[X_{it} \in R] = \Pi_{j=1}^{d} \mathcal{I}[a_j \leq (X_{it})_j \leq b_j]$$

If the problem involving coefficients U_{iat}^l is called $\mathcal{P}_{iat}^l$, then how do we derive relationships between problems differing in length step l, time step t, actions a.
How do we zoom in on the current problem to solve it using a branch-and-bound procedure or otherwise? First, notice that the a_j and b_j values can be chosen from the jth coordinate of the X_{it} values:

$$[a_j < b_j] \in [(X_{it})_j]$$

So, this is actually a purely integer optimization problem.
For given $t \in [T], l \in [l_t], a \in \mathcal{A}_t$ values, how would we simplify this problem:
We would expand z to take the values in $[0, 1]$ and each $a_j < b_j$ can take the values in $[0, C]$ which would give us a linear programming problem to solve that gives us an initial lower bound for the solution M^* and a fractional estimate of the final solution. We then apply the branch-and-bound procedure where we set $z_i = 0/1$, and then continue solving the subproblems, eliminating the cases where the problem yields a lower bound greater than M^*.
Lemma: If $U_i' > U_i \forall i \in [N]$ then the corresponding problem $\mathcal{P}_{i'}$ has solutions that are more valued than the solutions in $\mathcal{P}_i$.
Proof: Let $i_1', i_2', ..., i_q'$ be the points chosen for the problem U_i'.. We then have :

$$\sum_{k=1}^{q} z_{i_k'} U_{i_k}' \geq \sum_{k=1}^{q} z_{i'} U_{i_k} \geq \min \mathcal{P}(U_i)$$

How do we extend this analysis to our scenario?
Can you draw the branch-and-bound MILP solution in this scenario with the intention of extending it to future time-steps, length-steps, and actions? No, we are able to draw relations between the objectives for the various lengthsteps, but the solutions to the MIO problem is not continous or smooth in the coefficient values of the objective. For the different timesteps, we see that we can draw relations between the coefficients of the objective and the constraints - however non smoothness of the MIO solutions with respect to the coefficients implies we are unable to derive relations.
A possible issue with this MIO-based optimization is non-scalability. However we derive exact solutions to the same.

10 Future work

The future work on extending this line of research could include:

1. Given the causal model representing the MDP, how would we extend this approach to derive interpretable policies?
2. Debug VIDTR on simulated data and get the method to work on simulated RL environments.
3. Use distributed systems and multithreading to further scale and build the RL method.
4. Complete the convergence theory on using genetic algorithms to derive bounds on the method and it's working.

11 Code

We provide the code for the methods on the Github repo https://github.com/DeepakBadarinath?tab=repositories, which is the github repo of Deepak Badarinath. We test the algorithm's working on grid data. Further tests for the VIDTR on the experimental setup and RL Environments is currently under testing.

12 Appendix

12.1 Approximation to the VIDTR based on tightness of the transitions P_t and the rewards r_t

Assume that $||P_t - P_{t+1}|| < \epsilon_P$ and $||r_t - r_{t+1}|| < \epsilon_R$, where we measure the norm by looking at the differences in all the entries. For this section, we assume that $l_k^* = L \forall k$. Here we give the computation of the argmin. We recall the computation of optimal regions and actions.

$$(R_t^l, a_t^l) := \arg\min_{R_t^l, a_t^l}[\int_{R_t^l - \cup_{i<l} R_t^i} (\max_a [r_t(s,a) + \gamma P_t^a V_{t+1}(s)] -$$
$$[r_t(s, a_t^l) + \gamma P_t^{a_t^l} V_{t+1}(s)])ds - \eta_t V(R_t^l - \cup_{i<l} R_t^i) + \rho_t c(R_t^l)]$$

We define the following operator:

$$\Psi_t(R, a|r, P) := \int_{R_t^l - \cup_{i<l} R_t^i} (\max_a [r_t(s,a) + \gamma P_t^a V_{t+1}(s)] - [r_t(s, a_t^l) + \gamma P_t^{a_t^l} V_{t+1}(s)])ds -$$
$$\eta_t V(R_t^l - \cup_{i<l} R_t^i) + \rho_t c(R_t^l)$$

We then have the optimal actions and regions given by:

$$(R_t^{l*}, a_t^{l*}) := \arg\min_{R_t^l, a_t^l} \Psi(R_t^l, a_t^l | r_t, P_t)$$

In the next subsection, we list various theorems that follow as a consequence of the tightness assumption of the prob. kernels and rewards.

12.1.1 Theorems and lemmas

In this subsection, we list the main theorems used in the text: The assumption we have is tightness of the dynamics and the reward kernels i.e. $||P_t - P_{t+1}|| < \epsilon_P$ and $||r_t - r_{t+1}|| < \epsilon_R$.

Theorem 3. *We have the following theorem that relates how the value functions for the different times relate to each other:*

$|V_{T-i} - V_{T-i-1}| \leq C\frac{1-\gamma^{i+1}}{1-\gamma}$

Here we can see that $C := \epsilon_r + 2\gamma\epsilon_P R$

From this we can see that:

$$||V_{T-i}|| < C\frac{1-\gamma^{i+1}}{1-\gamma} + R$$

We also have the following theorem:

Theorem 4.

$$||\Psi_{t+1} - \Psi_t|| \leq |R|[3\epsilon_R + \gamma C\frac{1-\gamma^{T-t-1}}{1-\gamma}]$$

If we have a uniform continuity of Ψ_t for the different time steps $t \in [T]$. We can see that the minimums of $\Psi_t(R, a|r, P)$ are also close by. This step can also be useful in showing that the argmins of Ψ are close, i.e., R_{t+1}^{l*} lies in a region around R_t^{l*}.

12.1.2 Proofs

Theorem (2.2)
We have tightness in the transition kernels and in the rewards. This means that $d_R(r_t, r_{t+1}) < \epsilon_R$ and we have $d_P(P_t^a, P_{t+1}^a) < \epsilon_P$. How do we show that Ψ_t and Ψ_{t+1} are close? Recall that Ψ_t and Ψ_{t+1} are given by:

$$\Psi_t(R, a|r, P) := \int_{R_t^l - \cup_{i<l} R_t^i} (\max_a (r_t(s,a) + \gamma P_t^a V_{t+1}(s)) - [r_t(s, a_t^l) + \gamma P_t^{a_t^l} V_{t+1}(s)])ds$$
$$-\eta_t V(R_t^l - \cup_{i<l} R_t^i) + \rho_t c(R_t^l)$$

$$\Psi_{t+1}(R,a|r,P) := \int_{R^l_{t+1}-\cup_{i<l}R^i_{t+1}} (\max_a(r_{t+1}(s,a)+\gamma P^a_{t+1}V_{t+2}(s))$$
$$-[r_{t+1}(s,a^l_{t+1})+\gamma P^{a^l_{t+1}}_{t+1}V_{t+2}(s)])ds - \eta_{t+1}V(R^l_{t+1}-\cup_{i<l}R^i_{t+1})+\rho_{t+1}c(R^l_{t+1})$$

Let us start with length $l=1$:

$$\Psi^{(1)}_{t+1}(R,a|r,P) - \Psi^{(1)}_t(R,a|r,P) := \int_R (\max_a[r_{t+1}(s,a)+\gamma P^a_{t+1}V_{t+2}(s)]-$$
$$\max_a[r_t(s,a)+\gamma P^a_t V_{t+1}(s)])ds + \int_R -[(r_{t+1}-r_t)(s,a)+\gamma(P^a_{t+1}V_{t+2}-P^a_tV_{t+1})(s)]ds$$

Here we get:

$$\begin{aligned}|(\Psi^{(1)}_{t+1}-\Psi^{(1)}_t)(R,a|r,P)| := &\int_R (r_{t+1}(s,a^*_1)+\gamma P^{a^*_1}_{t+1}V_{t+2}(s))ds\\ &-\int_R (r_{t+1}(s,a)+\gamma P^a_{t+1}V_{t+2}(s))ds\\ &-\int_R (r_t(s,a^*_0)+\gamma P^{a^*_0}_t V_{t+1}(s))ds\\ &+\int_R (r_t(s,a)+\gamma P^a_t V_{t+1}(s))ds\end{aligned}$$

We split and expand each term in the above expression - this gives us:

$$\begin{aligned}\Psi^{(1)}_t(R,a|r,P) = &\int_R (r_{t+1}(s,a^*_1(s))-r_t(s,a^*_0(s)))ds+\\ &\gamma\int_R (P^{a^*_1(s)}_{t+1}V_{t+2}-P^{a^*_0(s)}_t V_{t+1})(s)ds\end{aligned}$$

We split and expand each term in expression 12.1.2 - this gives us:

$$+\int_R |r_{t+1}-r_t|(s,a)ds+$$
$$\gamma\int_R (P^a_tV_{t+1}-P^a_{t+1}V_{t+2})(s)ds$$

Continuing the expansion of the terms in the first expression, we get:

$$\begin{aligned}\int_R [&r_{t+1}(s,a^*_1(s))-r_{t+1}(s,a^*_0(s))+\\ &r_{t+1}(s,a^*_0(s))-r_t(s,a^*_0(s))]ds\end{aligned}$$

We then have the first expression to be $|r_{t+1}(s,a^*_1(s))-r_{t+1}(s,a^*_0(s))|$ which gives us the following inequality:

$$r_{t+1}(s,a^*_1(s))-r_{t+1}(s,a^*_0(s)) \le C_r d_A(a^*_1(s),a^*_0(s))$$

The next expression gives us this inequality:

$$r_{t+1}(s,a^*_0(s))-r_t(s,a^*_0(s)) \le \epsilon_R$$

After integration, we get the final expression:

$$C_r D|R| + \epsilon_r|R|$$

Continuing to expand upon the first term, we see that we get:

$$\gamma \int_R (P_{t+1}^{a_1^*(s)} V_{t+2} - P_{t+1}^{a_1^*(s)} V_{t+1} \\ + P_{t+1}^{a_1^*(s)} V_{t+1} - P_t^{a_0^*(s)} V_{t+1})(s) ds$$

Further expanding the above expression we get:

$$\gamma \int_R |V_{t+2} - V_{t+1}|(s) ds + 2 \int_R |V_{t+1}(s)| ds$$

Expanding the third and fourth terms respectively in 12.1.2 we get:

$$\leq \epsilon_r |R|$$

Fourth term expansion gives us:

$$\leq \gamma \int_R (P_t^a V_{t+1} - P_{t+1}^a V_{t+1} \\ + P_{t+1}^a V_{t+1} - P_{t+1}^a V_{t+2})(s) ds$$

Expanding the above expression gives us:

$$\gamma [2 \int_R |V_{t+1}| + \int_R |V_{t+1} - V_{t+2}|(s)]$$

Combining the first two expressions of 12.1.2 we get the following:

$$C_r D |R| + \epsilon_r |R| \\ + \gamma [\int_R |V_{t+2} - V_{t+1}|(s) ds + \int_R |V_{t+1}(s)| ds]$$

With the expansion and combination of the third and fourth term we get:

$$\epsilon_r + \gamma [2 \int_R |V_{t+1}| + \int_R |V_{t+1} - V_{t+2}|]$$

Adding all the terms together from 12.1.2 we get the following expression:

$$(C_r D + 2\epsilon_R)|R| + 2\gamma \int_R |V_{t+1}(s)| ds \\ + \int_R |V_{t+2}(s) - V_{t+1}(s)| ds$$

We try to bound $|V_t(s)|$ and $\int_R |V_{t+2}(s) - V_{t+1}(s)| ds$ for which we redo the VIDTR bounds as done in the main manuscript.

$$||V_{T-1}(s)|| \leq \bar{r} + \gamma \bar{r}$$

We make the claim that :

$$||V_t(s)|| \leq \bar{r}[1 + \gamma + \gamma^2 + ... + \gamma^{T-t}]$$

This gives us:

$$||V_t(s)|| \leq \frac{1 - \gamma^{T-t+1}}{1 - \gamma} \bar{r}$$

We obtain the above 12.1.2 to bound the value function at time t. Now we wish to bound the norm of the difference between the value functions integrated in a region R. In lieu of that goal, we expand things out to get:

$$\begin{aligned}|V_{t+1}(s) - V_t(s)| &= |\max_a f_{t+1}(s,a) - \max_a f_t(s,a)| \\ &\leq \max_a |f_{t+1}(s,a) - f_t(s,a)|\end{aligned}$$

We can show the above by first showing:

$$|\max_a f(a) - \max_a g(a)| \leq \max_a |f(a) - g(a)|$$

The above statement we show by first showing that:

$$\max_a (f+g)(a) \leq \max_a f(a) + \max_a g(a)$$

This holds trivially since if we take $a^* := \arg\max_a (f+g)$ and $a_f := \arg\max_a f(a)$ and $a_g := \arg\max_a g(a)$. Then what we have is the following:

$$f(a_f) \geq f(a^*), g(a_f) \geq g(a^*)$$

Summing together what we have above, we get the required answer.

If we then take $g := (g - f)$, then what we have is that:

$$\max_a (f + g - f)(a) \leq \max_a f(a) + \max_a (g-f)(a)$$

This then gives us:

$$\max_a g(a) - \max_a f(a) \leq \max_a (g-f)(a)$$

Recalling the definition of V we then get the following: From here we get:

$$\max_a [(r_{t+1}(s,a) - r_t(s,a)) + \gamma(P^a_{t+1} V_{t+2}(s) - P^a_t V_{t+1}(s))]$$

We then see:

$$\leq \epsilon_R + \gamma |V_{t+2} - V_{t+1}| + \gamma |P^a_{t+1} - P^a_t| V_{t+1}$$

From here we see:

$$\leq \epsilon_R + \gamma \epsilon_P ||V_{t+1}|| + \gamma |V_{t+2} - V_{t+1}|$$

At timestep T, we see that:

$$|V_T - V_{T-1}| \leq \epsilon_R + \gamma \bar{r} + C_r D$$

We get the above by expanding the terms as follows:

$$|V_T - V_{T-1}| \leq |\max_a (r_T(s,a)) - \max_a (r_{T-1}(s,a) + \gamma P^a_{T-1} V_T(s))|$$

If $a_1 := \arg\max_{a \in A} r_T(s,a)$ and $a_2 := \arg\max_{a \in A} (r_{T-1}(s,a) + \gamma P^a_{T-1} V_T(s))$. Then we have the following bound to hold:

$$|r_T(s,a_1) - (r_{T-1}(s,a_2) + \gamma P^{a_2}_{T-1} V_T(s))|$$

Here we bound $|r_T(s,a_1) - r_{T-1}(s,a_2)|$ by:

$$|r_T(s,a_1) - r_{T-1}(s,a_2)| \leq |r_T(s,a_1) - r_{T-1}(s,a_1) + r_{T-1}(s,a_1) - r_{T-1}(s,a_2)|$$

The above we then bound as $|r_T(s,a_1) - r_{T-1}(s,a_1)| \leq \epsilon_R$ and $|r_{T-1}(s,a_1) - r_{T-1}(s,a_2)| \leq C_r d_A(a_1,a_2) \leq C_r D$. This gives us the bound of $\epsilon_R + C_r D$ Then we can bound $\gamma P_{T-1}^{a_2} V_T \leq \gamma.1.\bar{r}$ as we have a global bound of $||V_T|| \leq \bar{r}$.

So we have that:

$$|V_T - V_{T-1}| \leq \epsilon_R + \gamma\bar{r} + C_r D$$

Assuming that $d_t := |V_t - V_{t-1}|$ we get the following recurrence relations for the different timesteps (combining eqn 12.1.2 and bounding the value function at the respective timestep 12.1.2):

$$d_{t+1} \leq \epsilon_R + \gamma\epsilon_P\bar{r}\frac{1-\gamma^{T-t}}{1-\gamma} + \gamma d_{t+2}$$

For $t = T-1$, we see the following relation:

$$d_T \leq \epsilon_R + \gamma\bar{r} + C_r D$$

We write a general formula for the recurrence relation.

For $t = T-2$ we see the following holds:

$$d_{T-1} \leq \epsilon_R + \gamma\epsilon_P\bar{r}(1+\gamma) + \gamma(\epsilon_R + \gamma\bar{r} + C_r D)$$

For $t = T-3$, we get different values which are given by:

$$d_{T-2} \leq \epsilon_R + \gamma\epsilon_P\bar{r}\frac{1-\gamma^3}{1-\gamma} + \gamma d_{T-1}$$

Bounding the Recurrence Relation

We are given the recurrence relation:

$$d_{t+1} \leq c + a\gamma^{T-t} + \gamma d_{t+2}, \quad \text{for } t = 1, 2, \ldots, T-1$$

with the terminal condition:

$$d_T \leq \epsilon_R + \gamma\bar{r} + C_r D$$

We aim to obtain a bound on d_1 in terms of the constants $c, a, \gamma, T, \epsilon_R, \bar{r}$.

We write the general equation which is given by:

$$d_{t+1} \leq c + a\gamma^{T-t} + \gamma d_{t+2} \forall t \in [T-1]$$

We now wish to find the bounds on d_1 in terms of $c, a, \gamma, T, \epsilon_R, \bar{r}$. We write things recursively and check to find a pattern. For example, we have the following for $t = 0$:

$$d_1 \leq c + a\gamma^T + \gamma d_2$$

For $t = 1$ we have the following to hold:

$$d_2 \leq c + a\gamma^{T-1} + \gamma d_3$$

Inserting the expression for d_1 into the above equation, we see that:

$$d_1 \leq c(1+\gamma) + 2a\gamma^T + \gamma^2 d_3$$

Further expansion yields the following:

$$d_1 \leq c(1+\gamma+\gamma^2) + 3a\gamma^T + \gamma^3 d_4$$

Finally we get the following pattern to hold

$$d_1 \leq c(1+\gamma+\gamma^2+...\gamma^{T-2}) + (T-1)a\gamma^T + \gamma^{T-1}d_T$$

The above then yields the following:

$$d_1 \leq c\frac{1-\gamma^{T-1}}{1-\gamma} + (T-1)a\gamma^T + \gamma^{T-1}d_T$$

Here we have $c = \epsilon_R + \frac{\gamma\epsilon_P\bar{r}}{1-\gamma}$ and $a = -\frac{\gamma\epsilon_P\bar{r}}{1-\gamma}$.

We recall 12.1.2 which is:

$$(C_r D + 2\epsilon_r)|R| + 2\gamma\int_R |V_{t+1}(s)|ds + \int_R |V_{t+2}(s) - V_{t+1}(s)|ds$$

The second integral above gives us:

$$2\gamma\int_R |V_{t+1}(s)|ds \leq 2\gamma|R|\bar{r}\frac{1-\gamma^{T-t}}{1-\gamma}$$

We now need to compute $|V_{t+2} - V_{t+1}|(s)$ in terms of the other constants. We note the calculation for the recursion done from before which is:

$$d_{t+2} \leq c + \gamma d_{t+2} + \bar{r}\gamma^{T-t-1}$$

Expanding the above gives us:

$$\leq c + \bar{r}\gamma^{T-t-1} + \gamma[c + \gamma d_{t+4} + r\gamma^{T-t-2}]$$

The above then gives us:

$$c(1+\gamma) + 2\bar{r}\gamma^{T-t-1} + \gamma^2 d_{t+4}$$

Further expanding we have the following:

$$\leq c(1+\gamma) + 2\bar{r}\gamma^{T-t-1} + \gamma^2[c + \gamma d_{t+5} + r\gamma^{T-t-3}]$$

This gives us the following expression:

$$c(1+\gamma+\gamma^2) + 3\bar{r}\gamma^{T-t-1} + \gamma^3 d_{t+5}$$

Expanding the above till the final timestep T, we see:

$$c(1+\gamma+...\gamma^{T-t-2}) + (T-t-1)\bar{r}\gamma^{T-t-1} + \gamma^{T-t-2}d_{t+(T-t)}$$

This gives us:

$$\leq c\frac{1-\gamma^{T-t-1}}{1-\gamma} + (T-t)\gamma^{T-t-1}\bar{r} + \gamma^{T-t-2}(\epsilon_r + C_r D)$$

To recall the expression for $|V_{t+2} - V_{t+1}|$ is given by:

$$|V_{t+2} - V_{t+1}| \leq c\frac{1-\gamma^{T-t-1}}{1-\gamma} + (T-t)\gamma^{T-t-1}\bar{r} + \gamma^{T-t-2}(\epsilon_r + C_r D)$$

Adding all the expressions 12.1.2 + 12.1.2 + $(C_r D + 2\epsilon_r)$ we get:

$$[(C_r D + \epsilon_r)(1 + \gamma^{T-t-2}) + \frac{[c - c\gamma^{T-t-1} + 2\gamma\bar{r} - 2\bar{r}\gamma^{T-t+1}]}{1-\gamma} + \epsilon_r]|R|$$

In the above equation, we have $c = \epsilon_R + \frac{\gamma\epsilon_P\bar{r}}{1-\gamma}$ and we have $a = \frac{\gamma\epsilon_P\bar{r}}{1-\gamma}$

The above can be written as

$$|(\Psi^{(1)}_{t+1} - \Psi^{(1)}_t)(R, a|r, P)| \leq B^{(1)}_t |R|$$

To find confidence intervals to find the optimal region in timestep $t+1$ when we know the optimal region in timestep t, we need to have the condition that the arg mins of the 2 functions are close, if the two functions are close. This holds, for instance, when the error functionals are convex. This would imply that the optimal regions and actions are close in time:

$$|(R^{(1)}_{t+1}, a^{(1)}_{t+1}) - (R^{(1)}_t, a^{(1)}_t)| \leq C^{(1)}_t$$

For lengthsteps 2 and more, we have that the regions we integrate are different. In that case the integrals are :

$$\Psi_{t+1}(R, a) := \int_{R - A_{t+1}} f_{t+1}(s, a)ds - \eta V(R - A_{t+1}) + \rho_{t+1} c(R)$$

The next integral is given by:

$$\Psi_t(R, a) := \int_{R - A_t} f_t(s, a)ds - \eta V(R - A_t) + \rho_t c(R)$$

Expanding out the integrals, we get:

$$\Psi_{t+1}(R, a) := \int_{R - A_{t+1} - A_t} f_{t+1}(s, a)ds + \int_{A_t \cap (R - A_{t+1})} f_{t+1}(s, a)ds - \eta V(R - A_{t+1}) + \rho_{t+1} c(R)$$

For Ψ_t we have the following to be true:

$$\Psi_t(R, a) := \int_{R - A_t - A_{t+1}} f_t(s, a)ds + \int_{A_{t+1} \cap (R - A_t)} f_t(s, a)ds - \eta V(R - A_t) + \rho_t c(R)$$

Subtracting from Ψ_{t+1} the value of Ψ_t, we see that we get the following:

$$\int_{R - A_{t+1} - A_t} (f_{t+1}(s, a) - f_t(s, a))ds - \eta[V(A_t - (R - A_{t+1})) - V(A_{t+1} - (R - A_t))] + \int_{A_t \cap (R - A_{t+1})} f_{t+1}(s, a)ds - \int_{A_{t+1} \cap (R - A_t)} f_t(s, a)ds$$

For the last two terms, we add and subtract $\int_{A_t \cap (R - A_{t+1})} f_t(s)ds$

$$\int_{A_t \cap (R - A_{t+1})} f_{t+1}(s, a)ds - \int_{A_t \cap (R - A_{t+1})} f_t(s, a)ds + \int_{A_t \cap (R - A_{t+1})} f_t(s, a)ds - \int_{A_{t+1} \cap (R - A_t)} f_t(s, a)ds$$

The above expression then gives us:

$$\int_{A_t \cap (R - A_{t+1})} (f_{t+1}(s,a) - f_t(s,a))ds + \int_{A_t \cap (R - A_{t+1})} f_t(s,a)ds - \int_{A_{t+1} \cap (R - A_t)} f_t(s,a)ds$$

For the volume terms we add the following:

$$-\eta[V(A_t - (R - A_{t+1})) - V(A_{t+1} - (R - A_t))]$$

In total, we see that the error is given by:

$$\Psi_{t+1}(R,a) - \Psi_t(R,a) = \int_{R - A_t - A_{t+1}} (f_{t+1}(s,a) - f_t(s,a))ds + \int_{A_t \cap (R - A_{t+1})} (f_{t+1}(s,a) - f_t(s,a))ds + \int_{A_t \cap (R - A_{t+1})} f_t(s,a)ds - \int_{A_{t+1} \cap (R - A_t)} f_t(s,a)ds + c(R)(\rho_{t+1} - \rho_t)$$

For the total we see that the following is true:

$$\Psi_{t+1}(R,a) - \Psi_t(R,a) = \int_{R - A_t - A_{t+1}} (f_{t+1} - f_t)(s,a)ds + \int_{A_t \cap (R - A_{t+1})} f_{t+1}(s,a)ds - \int_{A_{t+1} \cap (R - A_t)} f_t(s,a)ds + \int_{A_t \cap (R - A_{t+1})} (f_{t+1} - f_t)(s,a)ds + c(R)(\rho_{t+1} - \rho_t)$$

Here we add the integrals over the regions for $(f_{t+1} - f_t)(s,a)$ and noting that $[R - A_t - A_{t+1}] \sqcup [A_t \cap (R - A_{t+1})] = (R - A_{t+1})$ which gives us:

$$\int_{R - A_{t+1}} (f_{t+1}(s,a) - f_t(s,a))ds + \int_{A_t \cap (R - A_{t+1})} f_t(s,a)ds - \int_{A_{t+1} \cap (R - A_t)} f_t(s,a)ds$$

Adding the volume terms above, we end up with:

$$\Psi_{t+1}(R,a) - \Psi_t(R,a) = \int_{R - A_{t+1}} (f_{t+1}(s,a) - f_t(s,a))ds + \int_{A_t \cap (R - A_{t+1})} f_t(s,a)ds - \int_{A_{t+1} \cap (R - A_t)} f_t(s,a)ds$$

We now compute the bounds for $(f_{t+1}(s,a) - f_t(s,a))$, which is written as: We then have the following to be true:

$$f_{t+1}(s,a) = \max_a [r_{t+1}(s,a) + \gamma P^a_{t+1} V_{t+2}(s)] - [r_{t+1}(s,a) + \gamma P^a_{t+1} V_{t+2}(s)]$$
$$f_t(s,a) = \max_a [r_t(s,a) + \gamma P^a_t V_{t+1}(s)] - [r_t(s,a) + \gamma P^a_t V_{t+1}(s)]$$

The above can be extended to be as follows:

$$f_{t+1}(s,a) = [r_{t+1}(s, a^*_1(s)) + \gamma P^{a^*_1(s)}_{t+1} V_{t+2}(s)] - [r_{t+1}(s,a) + \gamma P^a_{t+1} V_{t+2}(s)]$$
$$f_t(s,a) = [r_t(s, a^*_0(s)) + \gamma P^{a^*_0(s)}_t V_{t+1}(s)] - [r_t(s,a) + \gamma P^a_t V_{t+1}(s)]$$

When we subtract f_t from f_{t+1} what we get is the following:

$$\begin{aligned} f_{t+1}(s,a) - f_t(s,a) = r_{t+1}(s, a_1^*(s)) - r_t(s, a_0^*(s)) \\ +\gamma[P_{t+1}^{a_1^*(s)} V_{t+2}(s) - P_t^{a_0^*(s)} V_{t+1}(s)] \\ -[(r_{t+1}(s,a) - r_t(s,a)) + \gamma(P_{t+1}^a V_{t+2}(s) - P_t^a V_{t+1}(s))] \end{aligned}$$

We can split the first part of the above equation using a triangle inequality as follows:

$$\begin{aligned} r_{t+1}(s, a_1^*(s)) - r_t(s, a_0^*(s)) = r_{t+1}(s, a_1^*(s)) - r_{t+1}(s, a_0^*(s)) \\ +r_{t+1}(s, a_0^*(s)) - r_t(s, a_0^*(s)) \end{aligned}$$

We then see that we can bound the above by $\epsilon_r + C_r d_A(a_1^*(s), a_0^*(s))$

The next part can be split as:

$$\begin{aligned} P_{t+1}^{a_1^*(s)} V_{t+2}(s) - P_{t+1}^{a_1^*(s)} V_{t+1}(s) \\ P_{t+1}^{a_1^*(s)} V_{t+1}(s) - P_t^{a_0^*(s)} V_{t+1}(s) \end{aligned}$$

Further splitting of the above gives us the following upper bound:

$$|V_{t+2}(s) - V_{t+1}(s)| + |P_{t+1}^{a_1^*(s)} - P_t^{a_0^*(s)}||V_{t+1}(s)|$$

The $|P_{t+1}^{a_1^*(s)} - P_t^{a_0^*(s)}|$ can be further divided as $|P_{t+1}^{a_1^*(s)} - P_{t+1}^{a_0^*(s)}| + |P_{t+1}^{a_0^*(s)} - P_t^{a_0^*(s)}|$.

The first part of this can be bounded by $C_p d_A(a_1^*(s), a_0^*(s))$ and the next part can be bounded by ϵ_r.

If we are looking to bound $|f_{t+1}(s) - f_t(s)|$ we see that we need to bound $|r_{t+1}(s,a) - r_t(s,a)| + |P_{t+1}^a V_{t+2} - P_{t+1}^a V_{t+1} + P_{t+1}^a V_{t+1} - P_t^a V_{t+1}|$. We can bound this as $|P_{t+1}^a||V_{t+2} - V_{t+1}| + |P_{t+1}^a - P_t^a||V_{t+1}|$. We finally expand this bound by $|V_{t+2} - V_{t+1}| + \epsilon_p |V_{t+1}|$.

Finally when we add all the bounds we get the following upper bound:

$$\begin{aligned} (C_r + \gamma C_p) d_A(a_1^*(s), a_0^*(s)) + 2\gamma |V_{t+2} - V_{t+1}| \\ +(\gamma + 1)\epsilon_r + \gamma \epsilon_p |V_{t+1}| \end{aligned}$$

We need to multiply the above with the volume of the respective regions. This gives us the following answer as an upper bound:

$$\begin{aligned} [(C_r + \gamma C_p) d_A(a_1^*(s), a_0^*(s)) + 2\gamma |V_{t+2} - V_{t+1}| \\ +(\gamma + 1)\epsilon_r + \gamma \epsilon_p |V_{t+1}|] V(R - A_{t+1}) \\ +|\int_{A_t \cap (R - A_{t+1})} f_t(s,a) ds - \int_{A_{t+1} \cap (R - A_t)} f_t(s,a) ds| \end{aligned}$$

Here we have that A_t and A_{t+1} are the regions chosen for the previous length and timestep, i.e. $A_t := \arg\min_{R \in \mathcal{R}} \Psi_t^{(1)}$ and $A_{t+1} := \arg\min_{R \in \mathcal{R}} \Psi_{t+1}^{(1)}$.
From here we bound the integral differences as follows:

$$|\int_A f(s,a) ds - \int_B f(s,a) ds| \leq M_f (V(A) + V(B))$$

Above we have that $M_f := \max_{s \in S, a \in A} f(s,a)$.
We can compute M_f as follows:

$$M_f := \max_{s \in S, a \in A} |f_t(s,a)|$$

Expanding out the above term we get:

$$f_t(s,a) = [r_t(s, a_0^*(s)) + \gamma P_t^{a_0^*(s)} V_{t+1}(s)] - [r_t(s,a) + \gamma P_t^a V_{t+1}(s)]$$

In the above, we bound $|r_t(s, a_0^*(s)) - r_t(s,a)| \leq C_r d_A(a, a_0^*(s)) \leq C_r|A|$. We then bound:

$$\gamma|P_t^{a_0^*(s)} - P_t^a||V_{t+1}| \leq \gamma C_p d_A(a_0^*(s), a)|V_{t+1}| \leq \gamma C_p|A||V_{t+1}|$$

So we see that:

$$M_f := D(C_r + \gamma C_p||V_{t+1}||)$$

We also sum up the volumes for the above regions by noting that:

$$V(A_t \cap (R - A_{t+1})) + V(A_{t+1} \cap (R - A_t)) = \\ V((A_t - A_{t+1}) \cap R) + V((A_{t+1} - A_t) \cap R)$$

Adding up all of the terms in the above bounds, what we get is the following:

$$D(C_r + \gamma C_p||V_{t+1}||)[V((A_t - A_{t+1}) \cap R) + V((A_{t+1} - A_t) \cap R)] + \\ [(C_r + \gamma C_p)D + 2\gamma|V_{t+2} - V_{t+1}| + (\gamma+1)\epsilon_r + \gamma\epsilon_p||V_{t+1}||]V(R - A_{t+1}) + \\ c(R)(\rho_{t+1} - \rho_t)$$

Assume that ρ is homogeneous in time; i.e we have ($\rho_t = \rho_{t+1} \forall t \in [T]$). In this case the above expression simplifies as:

$$D(C_r + \gamma C_p||V_{t+1}||)(V((A_t - A_{t+1}) \cap R) + V((A_{t+1} - A_t) \cap R)) + \\ [(C_r + \gamma C_p)D + 2\gamma|V_{t+2} - V_{t+1}| + (\gamma+1)\epsilon_r + \gamma\epsilon_p||V_{t+1}||]V(R - A_{t+1})$$

The above can also be written as:

$$D(C_r + \gamma C_p||V_{t+1}||)V((A_t \triangle A_{t+1}) \cap R) + \\ [(C_r + \gamma C_p)D + 2\gamma|V_{t+2} - V_{t+1}| + (\gamma+1)\epsilon_r + \gamma\epsilon_p||V_{t+1}||]V(R - A_{t+1})$$

How do we simplify the expressions from here? We do know that the following holds:

$$|(\Psi_{t+1}^{(1)} - \Psi_t^{(1)})(R, a|r, P)| \leq B_t^{(1)}|R|$$

From here, can we show that the optimal regions which are the argmins for the error functionals are close? This would be true if $\Psi_t^{(1)}$ and $\Psi_{t+1}^{(1)}$ were convex. In the expression three lines above, if we knew that A_t and A_{t+1} were close, what would the above expression look like? Furthermore, unless $\Psi_t^{(1)}$ and $\Psi_{t+1}^{(1)}$ are convex, we would not know if the optimal regions are close.

12.1.3 Convexity assumptions and more

In our case, let us assume that the assumption that Ψ_t is convex is true. As stated in the original IDTR paper,

We list the attempts made to simplify the above expression manually:

1) Assume that we have the assumption from the IDTR paper to be true. In that case, what we see is that the following is true:

Assumption 4i) from the IDTR paper gives us the following statement:

$$\Psi_t^{(l)}(R_t^{l*}, a_t^{l*}) - \Psi_t^{(l)}(R, a_t^{l*}) \geq \kappa\rho_t^2(R, R_t^{l*})$$

As stated in the IDTR paper, the above assumption implies that the error functional behaves nearly quadratic around the optimal threshold values, which is a common condition in parametric models.

For $t+1$ onwards, we have the following to be true:

$$\Psi^{(l)}_{t+1}(R^{l*}_{t+1}, a^{l*}_{t+1}) - \Psi^{(l)}_{t+1}(R, a^{l*}_{t+1}) \geq \kappa \rho^2_t(R, R^{l*}_{t+1})$$

Subtracting from equation 12.1.3, the equation 12.1.3, we see the following to be true:

$$\begin{aligned}(\Psi^{(l)}_{t+1}(R^{l*}_{t+1}, a^{l*}_{t+1}) - \Psi^{(l)}_{t}(R^{l*}_{t}, a^{l*}_{t}))\\ -(\Psi^{(l)}_{t+1}(R, a^{l*}_{t+1}) - \Psi^{(l)}_{t}(R, a^{l*}_{t}))\\ \geq \kappa(\rho^2_{t+1}(R, R^{l*}_{t+1}) - \rho^2_t(R, R^{l*}_t))\end{aligned}$$

Later we show in corr. 1 that if two functions are close, then the arg mins of the functions are close. Using that theorem what we see that:

$$\begin{aligned}(\Psi^{(l)}_{t+1}(R^{l*}_{t+1}, a^{l*}_{t+1}) - \Psi^{(l)}_{t}(R^{l*}_{t}, a^{l*}_{t}))\\ \leq C^t_1 V((A_t \Delta A_{t+1}) \cap R) + C^t_2 V(R - A_{t+1})\\ -\kappa[V^2(R \Delta R^{l*}_{t+1}) - V^2(R \Delta R^{l*}_t)]\end{aligned}$$

When we further simplify the above, we see that the following is true:

$$\begin{aligned}(\Psi^{(l)}_{t+1}(R^{l*}_{t+1}, a^{l*}_{t+1}) - \Psi^{(l)}_{t}(R^{l*}_{t}, a^{l*}_{t}))\\ \leq (C^t_1 + C^t_2)V(S)\\ -\kappa[V^2(R \Delta R^{l*}_{t+1}) - V^2(R \Delta R^{l*}_t)]\end{aligned}$$

Moving the last term above to the left we see that the following is true:

$$\begin{aligned}\kappa[V^2(R \Delta R^{l*}_{t+1}) - V^2(R \Delta R^{l*}_t)] \leq\\ (C^t_1 + C^t_2)V(S) - [\Psi^{(l)}_{t+1}(R_{t+1}, a_{t+1}) - \Psi^{(l)}_t(R_t, a_t)]\end{aligned}$$

Above we simplify the final term using the triangle inequality as follows:

$$\begin{aligned}\Psi^{(l)}_{t+1}(R_{t+1}, a_{t+1}) - \Psi^{(l)}_t(R_t, a_t) \leq\\ (\Psi^{(l)}_{t+1}(R_{t+1}, a_{t+1}) - \Psi^{(l)}_{t+1}(R_t, a_t))\\ +(\Psi^{(l)}_{t+1}(R_t, a_t) - \Psi^{(l)}_t(R_t, a_t))\end{aligned}$$

Above we need to check whether $\Psi^{(l)}_{t+1}$ is Lipschitz and how the bounds for the difference unravel itself. We can bound the second term using the general upper bound as in 12.1.2. Completing the above term, we get the following to be true:

$$\begin{aligned}\Psi^{(l)}_{t+1}(R_{t+1}, a_{t+1}) - \Psi^{(l)}_t(R_t, a_t) \leq\\ (C^t_1 + C^t_2)V(S) +\end{aligned}$$

Q) How do we check the Lipschitzness of the error functional Ψ_t? Recall that the following is true for $\Psi_t(R, a)$.

$$\begin{aligned}\Psi_t(R', a') - \Psi_t(R, a) = \int_{R'} f_t(s, a')ds - \eta_t V(R' - A_{t+1}) + \rho_t c(R')\\ -(\int_R f_t(s, a)ds - \eta_t V(R - A_{t+1}) + \rho_t c(R))\end{aligned}$$

Expanding out the above terms, we see that we get:

$$\begin{aligned}\int_{R'} f_t(s,a')ds - \int_R f_t(s,a)ds + \\ \rho_{t+1}[c(R') - c(R)] + \\ \eta_t[V(R - A_{t+1}) - V(R' - A_{t+1})]\end{aligned}$$

Expanding out the term in the integral, we get:

$$\begin{aligned}\int_{R'} f_t(s,a')ds - \int_{R'} f_t(s,a)ds \\ + \int_{R'} f_t(s,a)ds - \int_R f_t(s,a)ds\end{aligned}$$

We expand the above equation to get:

$$\int_{R'} (f_t(s,a') - f_t(s,a))ds + \int_{R'} f_t(s,a)ds - \int_R f_t(s,a)ds$$

Expanding the above term, we see that we get:

$$|f_t(s,a') - f_t(s,a)| \leq |r_t(s,a) - r_t(s,a')| - \gamma[P_t^a - P_t^{a'}]V_{t+1}(s)$$

Expanding on the above bounds we see that we get:

$$\leq (C_r d_A(a,a') + \gamma C_p ||V_{t+1}||d_A(a,a'))|R'|$$

For the other integral, we see that we have the following to be true:

$$\leq \int_{R'-R} f_t(s,a)ds - \int_{R-R'} f_t(s,a)ds$$

We bound the above term as follows:

$$|r_t(s,a) + \gamma P_t^a V_{t+1}| \leq \bar{r} + \gamma ||P_t^a||||V_{t+1}||$$

After integration over the respective regions, we see that we are left with the following bound:

$$(\bar{r} + \gamma ||V_{t+1}||)(V(R' - R) - V(R - R'))$$

Adding up the sums we see that the following is true:

$$\begin{aligned}(\bar{r} + \gamma ||V_{t+1}||)(V(R' - R) - V(R - R')) \\ + (C_r + \gamma C_p)||V_{t+1}||d_A(a,a'))V(R')\end{aligned}$$

For the total bounds, we see that the following is true:

$$\begin{aligned}(\bar{r} + \gamma ||V_{t+1}||)(V(R' - R) - V(R - R')) \\ + (C_r + \gamma C_p)||V_{t+1}||d_A(a,a'))V(R') \\ + (C_1^t + C_2^t)V(S)\end{aligned}$$

Recall that in the RHS we have $R = R_t^{l*}$ and $R' = R_{t+1}^{l*}$ and we also have the LHS looked like from before and then we see that the following is true:

$$\begin{aligned}2\kappa V(R)(V(R\Delta R_{t+1}^{l*}) - V(R\Delta R_t^{l*})) \leq (\bar{r} + \gamma ||V_{t+1}||)(V(R_{t+1}^{l*} - R_t^{l*}) - V(R_t^{l*} - R_{t+1}^{l*})) \\ + (C_r + \gamma C_p)||V_{t+1}||d_A(a,a'))V(R_{t+1}^{l*}) \\ + 2(C_1^t + C_2^t)V(S)\end{aligned}$$

Above we see that the LHS is given by:

$$(V(R\Delta R_{t+1}^{l*}) - V(R\Delta R_t^{l*})) \leq [(\bar{r} + \gamma||V_{t+1}||)(V(R_{t+1}^{l*} - R_t^{l*}) - V(R_t^{l*} - R_{t+1}^{l*})) \\ +(C_r + \gamma C_p)||V_{t+1}||d_A(a, a'))V(R_{t+1}^{l*}) \\ +2(C_1^t + C_2^t)V(S)]\frac{1}{2\kappa V(R)}$$

2) Check the convexity of the error functional Ψ_t bit by bit, which would give us:

Convexity of Ψ_t: Here we check if Ψ_t is convex and how far it is from being convex. In this setup, we recall the definition of Ψ_t which is given by:

$$\Psi_t(R, a|r, P) := \int_{R-A_t} (\max_a [r_t(s, a) + \gamma P_t^a V_{t+1}(s)] - [r_t(s, a) + \gamma P_t^a V_{t+1}(s)]) \\ -\eta V(R - A_t) + \rho c(R)$$

We note that $\Psi_t(R|\alpha) := \int_R f_t(s|\alpha)ds$, f_t is appropriately defined. For the $1 - D$ case we have that the error functional is given as:,

$$\Psi_t((a, b)|\alpha) := \int_a^b f_t(s|\alpha)ds$$

Here we can check convexity as:

$$\Psi_t(\lambda(a_1, b_1) + (1 - \lambda)(a_2, b_2)|\alpha) = \int_{a_\lambda}^{b_\lambda} f_t(s|\alpha)ds$$

In the above equation, we have that $a_\lambda := \lambda a_1 + (1 - \lambda)a_2$ and we have that $b_\lambda := \lambda b_1 + (1 - \lambda)b_2$. In one scenario, let us assume we integrate over regions (a_1, b_1) and (a_2, b_2), where we know that the following holds: $a_1 < b_1 < a_2 < b_2$, and in this case we see that the following is true:

$$a_1 < a_\lambda < a_2 < b_1 < b_\lambda < b_2$$

For convexity to hold, we see that the following equation must be true:

$$\int_{a_\lambda}^{b_\lambda} f_t(s|\alpha)ds \leq \lambda \int_{a_1}^{b_1} f_t(s|\alpha)ds + (1 - \lambda) \int_{a_2}^{b_2} f_t(s|\alpha)ds$$

On the LHS, we add and subtract $\int_{a_1}^{a_\lambda} f_t(s|\alpha)ds$, we also add and subtract $\int_{b_\lambda}^{b_2} f_t(s|\alpha)ds$ on the LHS to give the full integral. This gives us the following integral on the LHS:

$$\int_{a_1}^{b_1} f_t(s|\alpha)ds - \int_{a_1}^{a_\lambda} f_t(s|\alpha)ds - \int_{b_\lambda}^{b_2} f_t(s|\alpha)ds$$

We can now split the integrals on the RHS as follows:

$$\lambda \int_{a_1}^{b_1} f_t(s|\alpha)ds = \lambda[\int_{a_1}^{a_\lambda} f_t(s|\alpha)ds + \int_{a_\lambda}^{a_2} f_t(s|\alpha)ds + \int_{a_2}^{b_1} f_t(s|\alpha)ds]$$

We also split the next term on the RHS as :

$$(1 - \lambda) \int_{a_2}^{b_2} f_t(s|\alpha)ds = (1 - \lambda)[\int_{a_2}^{b_1} f_t(s|\alpha)ds + \int_{b_1}^{b_\lambda} f_t(s|\alpha)ds + \int_{b_\lambda}^{b_2} f_t(s|\alpha)ds]$$

If we move the two extra terms that we added and subtracted on the LHS to the right, what we see is that we get:

$$\int_{a_1}^{b_2} f_t(s|\alpha)ds \leq \int_{a_1}^{a_\lambda} (1+\lambda) f_t(s|\alpha)ds + \int_{a_\lambda}^{a_2} \lambda f_t(s|\alpha)ds + \int_{a_2}^{b_1} f_t(s|\alpha)ds + \int_{b_1}^{b_\lambda} (1-\lambda) f_t(s|\alpha)ds + \int_{b_\lambda}^{b_2} (2-\lambda) f_t(s|\alpha)ds$$

Canceling out the integral over the region $[a_2, b_1]$ with the total integral on the left hand side, what we see is that we get:

$$\int_{a_1}^{a_2} f_t(s|\alpha)ds + \int_{b_1}^{b_2} f_t(s|\alpha)ds \leq \int_{a_1}^{a_\lambda} (1+\lambda) f_t(s|\alpha)ds + \int_{a_\lambda}^{a_2} \lambda f_t(s|\alpha)ds + \int_{b_1}^{b_\lambda} (1-\lambda) f_t(s|\alpha)ds + \int_{b_\lambda}^{b_2} (2-\lambda) f_t(s|\alpha)ds$$

Further breaking down the integral on the left, we see that we get:

$$\int_{a_1}^{a_2} f_t(s|\alpha)ds = \int_{a_1}^{a_\lambda} f_t(s|\alpha)ds + \int_{a_\lambda}^{a_2} f_t(s|\alpha)ds$$

We also break the integral over $[b_1, b_2]$ apart, which gives us:

$$\int_{b_1}^{b_2} f_t(s|\alpha)ds = \int_{b_1}^{b_\lambda} f_t(s|\alpha)ds + \int_{b_\lambda}^{b_2} f_t(s|\alpha)ds$$

Moving the corresponding terms on the LHS to the RHS, we get:

$$\lambda[\int_{a_1}^{a_2} f_t(s|\alpha)ds - \int_{b_1}^{b_2} f_t(s|\alpha)ds] \geq \int_{a_\lambda}^{a_2} f_t(s|\alpha)ds - \int_{b_\lambda}^{b_2} f_t(s|\alpha)ds$$

Does the above inequality hold for:

$$f_t(s,\alpha) = \max_{a\in\mathcal{A}}[r_t(s,a) + \gamma P_t^a V_{t+1}(s)] - [r_t(s,\alpha) + \gamma P_t^\alpha V_{t+1}(s)]$$

In our experiments, we can see that this is not necessarily true. The other option is to have convexity as part of the statements in our main theorem. However, this does not hold for our grid-based experiments.

Below we list the theorems and lemmas required to check closeness of the argmins of two functions, given that the two functions are close:

Lemma 1. *If $\alpha \in \mathcal{A}, \beta \in \mathcal{B}$ and we have that*

$$|f(\alpha|\beta) - g(\alpha|\beta)| \leq C$$

Then we can see that the following holds $\forall \alpha \in \mathcal{A}, \beta \in \mathcal{B}$:

$$|\min_{\beta\in\mathcal{B}} f(\alpha|\beta) - \min_{\beta\in\mathcal{B}} g(\alpha|\beta)| \leq C$$

Proof. Assume that $m_f := \min_\beta f(\alpha|\beta)$ and assume that $m_g := \min_\beta g(\alpha|\beta)$.
Let us fix an arbitrary $\alpha_0 \in \mathcal{A}$, then $\exists\ \beta' \in \mathcal{B}$ such that:

$$m_f(\alpha_0) \leq f(\alpha_0|\beta') < m_f(\alpha_0) + \epsilon$$

$$m_f(\alpha_0) - g(\alpha_0|\beta') \leq f(\alpha_0|\beta') - g(\alpha_0|\beta') < m_f(\alpha_0) + \epsilon - g(\alpha_0|\beta')$$

We then have:

$$m_f(\alpha_0) - g(\alpha_0|\beta') \leq C$$

We then have:

$$g(\alpha_0|\beta') \geq m_f(\alpha_0) - C$$

We also have:

$$m_g(\alpha_0) \geq m_f(\alpha_0) - C$$

Above we define $m_g(\alpha) := \min_{\beta \in \mathcal{B}} g(\alpha|\beta)$ and apply the inequality when $\alpha = \alpha_0$. This implies that:

$$m_f(\alpha_0) - m_g(\alpha_0) \leq C$$

Exchanging the role of f and g, we see that:

$$m_g(\alpha_0) - m_f(\alpha_0) \leq C$$

This gives us that:

$$|m_f(\alpha_0) - m_g(\alpha_0)| \leq C$$

□

Under what conditions can we show that the arg mins of the functions are close to one another? Further, is it true that the arg mins of the functions are unique?

Lemma 2. *Assume $f : \mathbb{R}^d \to \mathbb{R}$ to be convex, Lipschitz with constant C, i.e., $||f(x) - f(y)|| \leq C||x - y||$, and differentiable. In addition, assume $|\partial_{x_i} f| \leq D$ for all $x \in \mathbb{R}^d$. We then have that*

$$dD||x - y|| \leq ||f(x) - f(y)||$$

Proof. We find that the following holds:

$$f(tx + (1-t)y) \leq tf(x) + (1-t)f(y)$$

When we move a part of the RHS to the left-hand side, we see that:

$$f(tx + (1-t)y) - f(y) \leq t(f(x) - f(y))$$

Dividing by t on the LHS and having $t \to 0$, we see:

$$\frac{f(tx + (1-t)y) - f(y)}{t} \leq (f(x) - f(y))$$

If we have $\lim_{t \to 0}$ on the LHS, what we get is:

$$||x - y|||\partial_{x_i} f| \leq ||f(x) - f(y)||$$

From here we get the implied inequalities by looking at the bounds over $\partial_i f \leq D$ and summing the LHS inequality d times. □

Lemma 3. *Assume f, g to be convex. Lipschitz with constant C, further assume that the derivatives are bounded by D. Also, assume that f and g are uniformly close by ϵ. Assume x_f is the minimizer of f, and x_g is the minimizer of g. In this case, we can show that x_f and x_g are close:*

$$||x_f - x_g|| \leq \frac{2\epsilon}{dD}$$

Proof. We have the following to be true:

$$||x_f - x_g|| \leq \frac{1}{dD}||f(x_f) - f(x_g)||$$

Adding and subtracting $g(x_g)$, we get:

$$||f(x_f) - f(x_g)|| \leq ||f(x_f) - g(x_g) + g(x_g) - f(x_g)||$$

Further splitting of the above equation yields:

$$||f(x_f) - f(x_g)|| \leq ||f(x_f) - g(x_g)|| + ||g(x_g) - f(x_g)||$$

Since g, f are uniformly close, which also implies that the minimums are uniformly close, what we see is that:

$$||x_f - x_g|| \leq \frac{2\epsilon}{dD}$$

□

Can we obtain bounds for the derivatives to be used in the following lemma? We then bound the derivative in this case. In this case, we need to differentiate the following with respect to $\frac{\delta}{\delta s}$:

$$\begin{aligned} f_t(s|\alpha) := \max_{\alpha}[r_t(s,\alpha) + \gamma P_t^{\alpha} V_{t+1}(s)] \\ -[r_t(s,a) + \gamma P_t^{a} V_{t+1}(s)] - \eta \end{aligned}$$

Differentiating the above with respect to s, we see that we get:

$$\begin{aligned} \max_{\alpha}[r_t'(s,\alpha) + \gamma P_t^{\alpha} V_{t+1}^{'}(s)] \\ -[r_t^{'}(s,a) + \gamma P_t^{a} V_{t+1}^{'}(s)] \end{aligned}$$

If we assume $||r_t'|| \leq \bar{R}$, where taking the bar represents the fact that we are on the space of derivatives, we see that the following is true for the upper bound $D = \bar{R}$. Assuming bounds on the rewards and value functions, we see that we get the following to be true:

$$|V_t'(s)| \leq 1 + \gamma\bar{R} + \gamma^2\bar{R}^2 + ... + \gamma^{T-1}(\bar{R})^{T-1}$$

We can bound ϵ which represents the distances between the error functionals for the different timesteps, using the expression from 12.1.2.

$$\begin{aligned} \epsilon := D(C_r + \gamma C_p||V_{t+1}||)V(S)+ \\ [(C_r + \gamma C_p)D + 2\gamma|V_{t+2} - V_{t+1}| + (\gamma+1)\epsilon_r + \gamma\epsilon_p||V_{t+1}||]V(S) \end{aligned}$$

Under the strong assumption of convexity for the error functional $\Psi_t^{(l)}$, we see that the bounds around which the optimal condition and action are located are given by:

$$\rho_t((R_t^{l*}, a_t^{l*}), (R_{t+1}^{l*}, a_{t+1}^{l*})) \leq \frac{\bar{R}\gamma - 1}{d(\bar{R}\gamma)^T - 1)}\epsilon$$

What we see in the above equation is that the optimal regions and optimal actions all differ by a term that is larger than $V(S)$. So we would need to change the bounds and look for alternatives.

Hence using this, we derive confidence intervals around the optimal region derived at a given time and lengthstep. Towards this goal, we derive the following theorem regarding the tight value interpretable dynamic treatment regimes approach:

Theorem 5. *Given an MDP with parameters* $(P_t)_{t=1}^T$ *and* $(r_t)_{t=1}^T$ $\forall t \in [T]$*, and assume we have the tightness constraints from before where we know that* $||P_t - P_{t+1}|| < \epsilon_P$ *and* $||r_t - r_{t+1}|| < \epsilon_R$*. In this situation, we see that we can obtain the following confidence intervals around the optimal regions as given by 5:*

$$\rho_t((R_t^{l*}, a_t^{l*}), (R_{t+1}^{l*}, a_{t+1}^{l*})) \leq \frac{\bar{R}\gamma - 1}{d(\bar{R}\gamma)^T - 1)}\epsilon$$

Here we have ϵ *to be defined as in 12.1.3*

The above gives us a method to compute the optimal region for timestep $t+1$ and lengthstep l, if we know the optimal region for timestep t and lengthstep l.

12.1.4 Strong convexity implies argmins are close

Q) Can we compute the errors in approximation; at least for the 1 dimensional case? The problem we wish to solve here is the following.

$$\min_g \int_D |f(x) - g(x)|dx$$

We have the following condition that we subject our problem to:

$$\nabla^2 g \succ 0$$

In the 1-D case, this problem can be restated as:

$$\min_g \sum_{i=0}^{N} ||f(x_i) - g(x_i)||_{2/1}$$

This is subject to the following condition:

$$g(x_{i+1}) + g(x_{i-1}) - 2g(x_i) \geq 0$$

We solve a general version of the problem given by the following. We solve the matrix positive hyperplane problem for a given vector f and matrix A. The problem is defined as follows:
Given f values on the $\mathbb{R}^1$ a vector of length $(c, 1)$ and a matrix A of dimensions (r, c) of full-row rank. Solve the following problem:

$$\min_g ||f - g||$$

Subject to:

$$Ag \succ 0$$

Compute the SVD of $A := U\Sigma V^T$, we then have the following condition $U\Sigma V^T g \succ 0$. Define $y := V^T g$. Multiply by V^T on the left in the corresponding optimization problem 12.1.4. We get the following modified problem

$$\min_y ||V^T f - y||$$

Assuming that singular values are $\lambda_1, \lambda_2, ..., \lambda_r$. The condition which this is subject to is given by:

$$U[\lambda_1 y_1, \lambda_2 y_2, ..., \lambda_r y_r]^T \succ 0$$

Since $\lambda_1, \lambda_2, ..., \lambda_r > 0$, we get that this condition then gives us $(y_1, y_2, ..., y_r)^T \in U^{-1}\mathbb{H}_+^r$. To the optimization problem 12.1.4 we left multiply by $[[U, O_{r,c-r}], [O_{c-r,r}, I_{c-r,c-r}]]$ where O is a zero matrix with the dimensions in the subscript. We can do this and the norm does not change since the above matrix is orthogonal since each block is orthogonal. We then get the following in the norm:

$$||[[U, O], [O, I]]^T V^T f - [U\Pi_r y, y_{r+1}, ..., y_c]^T||$$

Here and elsewhere we have that $\Pi_i y = [y_1, y_2, ..., y_i]^T$, where $y = [y_1, y_2, ..., y_n]^T$
This is subject to

$$\Pi_r y \in U^{-1}\mathbb{H}_+^r$$

After left multiplying by U, we see that this splits as:

$$||[U\Pi_r V^T f, \Pi_{c-r} V^T f]^T - [U\Pi_r y, y_{r+1}, y_{r+2}, ..., y_c]^T||$$

The above 12.1.4 then splits as:

$$||U\Pi_r V^T f - U\Pi_r y|| + ||\Pi_{c-r} V^T f - \Pi_{c-r} y||$$

This is subject to $\Pi_r y \in U^{-1}\mathbb{H}_+^r$
Since there are no conditions on $\Pi_{c-r} y$, we can have $\Pi_{c-r} y = \Pi_{c-r} V^T f$, this then implies that the second part of 12.1.4 can be taken to be 0 which leaves us with:

$$||U\Pi_r V^t f - U\Pi_r y||$$

Here we must have that $U\Pi_r y \in \mathbb{H}_+^r$ and this can be any r dimensional positive vector. This implies that we can choose $(U\Pi_r y)_i = (U\Pi_r V^t f)_i$ if $(U\Pi_r V^t f)_i > 0$, otherwise we must have $(U\Pi_r y)_i = 0$.

$$\begin{aligned}(U\Pi_r y)_i &= (U\Pi_r V^t f)_i \text{ if } (U\Pi_r V^t f)_i \geq 0 \\ &= 0 \text{ if } (U\Pi_r V^t f)_i < 0\end{aligned}$$

Let P_n be a positive vector map from $\mathbb{R}^n \to \mathbb{R}^n$ such that if $v_i \geq 0$ then $(P_n v)_i = v_i$ and if $v_i < 0$ then $(P_n v)_i = 0$.

$$\begin{aligned}(P_n v)_i &= v_i \text{ if } v_i \geq 0 \\ &= 0 \text{ if } v_i < 0\end{aligned}$$

Then we find that the following is true for $\Pi_r y = U^{-1} P_r(U\Pi_r V^t f)$ and we must have the norm given by $||(I - P_r)U\Pi_r V^t f||$. We formally state the theorem as follows:

Theorem 6. *Given a vector f of dimensions $(c, 1)$ and a matrix A of dimension (r, c) of full row rank, the solution to the following constrained optimization problem is given:*

$$\min_g ||f - g||$$

Subject to:

$$Ag \succ 0$$

If we assume $y := V^t g$, then we see that the solution is given by

$$\Pi_r y = U^t P_r(U\Pi_r V^t f)$$

and we also have:

$$\Pi_{c-r} y = \Pi_{c-r}(V^t f)$$

where Π_r is the projection map from $\mathbb{R}^c \to \mathbb{R}^r$ and P_r is the positivity map mapping $(v_i)_{i=1}^r \to (\mathcal{I}(v_i > 0)v_i)_{i=1}^r$. The minimizing norm is given by $||(I - P_r)U\Pi_r V^t f||$.

For another approach to compute the convex envelope of a given function, there is an approach in the paper 'Approximation of Convex Envelope Using Reinforcement Learning' where they compute the convex approximation using a Q-learning-based approach. Future steps could include extending this approach and completing the next steps listed.

12.1.5 Pitfalls with tightness assumptions

1. The objective need not be convex in the parameters (experimentally can see this is not true for grid-based settings).

2. Construction of a convex approximation seemed complicated and we would not know if inverse Lipschitzness holds here.

3. Further this method was abandoned since we can just search for the optimal answer by assuming custom confidence intervals. (i.e search for optimal answer at time $t+1$ and length l by searching a fixed length around the optimal answer at time t and length l) I see no real purpose of deriving exact confidence bounds.

13 Two Variable Optimization Problem

The optimization problem in this setting is given by:

$$(\hat{\tau}, \hat{\sigma}) := \arg\min_{\tau,\sigma} \sum_{i=1}^{n} U_{ia} I[X_{ij} \leq \tau, X_{ik} \leq \sigma]$$

14 Q-Tree Algorithm Design

The plan is to construct a binary tree of height $O(\log_2 n)$ that has n leaves. We wish to fill these leaf positions using the variables U_{ia}. The plan would be to propagate the sums of the values from the leaves to the upper nodes, and finally to the root.

Each node in the tree would contain information on the sums of the Us in the corresponding subtree, as well as information on the thresholding sum, i.e., the minimum value among the sum of all Us that satisfies $X_{i_s k} \leq \sigma$, where σ can take the value of any $X_{i_s k}$ in the sub-tree.

Each node has information on the following: The sum of the U values in the sub-tree which is given by:

$$\sum_{i \in T} U_i$$

Also- each node has information on the best thresholding sum which is to find the optimal σ that minimizes $\sum_{h \leq s} U_{i_h a} I(X_{i_h k} \leq \sigma)$ in $O(log_2 n)$ time.

Updating the sum after the addition of a new U is easy, it involves the propagation of the sums from the base leaf level to the top root. For updating the best thresholding sum, they say that this is given by either the best thresholding sum in its left child or the sum of all $U_{i_s a}$ in the left child plus the best thresholding sum in the right child, whichever is smaller.

The first case corresponds to picking the answers obtained using only those leaf values on the left, and in the second case what we see is that we get the answers when using all the leaves from the left in addition to the answers obtained from using the answers obtained from the right nodes.

More formally - if we are looking at the case where we have 4 leaves corresponding to X_1, X_2, X_3, X_4. Say we have 2 nodes N_1 and N_2 such that N_1 connects X_1 and X_2 and N_2 connects X_3 and X_4. Say we have a node N connecting N_1 and N_2.

In this case, at N_1 we pick $\min(X_1, X_1 + X_2)$ and at N_2 we pick $\min(X_3, X_3 + X_4)$. At node N, we pick $Ans(N_1)$ versus $N_1 + Ans(N_2)$ as the answer to the root.

What exactly happens algebraically? Let us introduce the following rank operators. We assume that we have $[X_{ij}]_{i=1}^n$ and $[X_{ik}]_{i=1}^n \forall i \in [n]$

$$r_J : [n] \to [n]$$

where $r_J(i) :=$ (Rank of X_{ij} in $X_{tj} \forall t \in [n]$) and we define similarly $r_K(i) :=$ (Rank of X_{ik} in $X_{tk} \forall t \in [n]$).

$$r_K : [n] \to [n]$$

What we see is that U_{ia} goes to the leaf $r_K(r_J(i))$.

Can you update the above analysis so that we can see whether smaller versions of this problem are solved for each time and lengthstep?

We see that the $U_{ia}s$ are added so that they follow the order statistics for the $X_i^{(k)}$ values for the different order statistics k.

14.1 Runtime analysis

Here, we analyze the total time taken for the algorithm design. We first sort the $X_{\cdot j}$ terms, which is necessary to check how to add the U terms. This takes time $O(nlogn)$, we then add each U_i to the tree. This is done after computing its location on the basis of the order statistics of the $X_{\cdot k}$ terms. That can be done as follows: sort the $X_{\cdot k}$ terms that would take $O(nlogn)$ time and locate the position of the U which would take $O(logn)$ time. The addition of U terms would also take $O(logn)$. Furthermore, we add n of these U terms which would give us a total time of $O(nlogn)$.

We can then loop over all of the actions for the different U_{ia} terms, which gives us a m_t factor, and looping over all possible terms adds a $C_2^{d_t}$ factor, which means that the total run time is $O(nlognd_t^2 m_t)$.

14.2 Greater than or equal to

Assume the scenario where we are in the following setup: The optimization problem for the greater than or equal to setup:

$$(\hat{\tau}, \hat{\sigma}) := \arg\min_{\tau,\sigma} \sum_{i=1}^n U_{ia} I[X_{ij} \geq \tau, X_{ik} \geq \sigma]$$

How do we design an algorithm in this case? Maybe we can do this by first reverting back to the 1-D setup where we have the following to be true:

$$\hat{\tau} := \arg\min_{\tau} \sum_{i=1}^n U_{ia} I[X_{ij} \geq \tau]$$

$$F(\tau) = \sum_{i=1}^n U_{ia} I[X_{ij} \geq \tau]$$

Order the $(X_{ij})_{j=1}^n$ values wrt j to get $X_{ij^{(1)}} \leq X_{ij^{(2)}} \leq ..., \leq X_{ij^{(n)}}$. We then see that [14.2] is a piecewise constant function of τ.

In particular, we have that if $\tau \in (X_{ij^{(l)}}, X_{ij^{(l+1)}}]$ then $F(\tau) = F(X_{ij^{(l+1)}})$, so it is really a matter of checking the order of τ in the sorted list and choosing the node on the right. Writing the whole thing out, we see that $F(\tau) = U_{na}$ for $\tau \in (X_{ij^{(n-1)}}, X_{ij^{(n)}}]$. The complete expansion looks like:

$$\begin{aligned}
F(\tau) &= U_{n,a} \text{ if } \tau \in (X_{ij^{(n-1)}}, X_{ij^{(n)}}] \\
&= \min(U_{n-1,a} + U_{n,a}, U_{n,a}) \text{ if } \tau \in (X_{ij^{(n-2)}}, X_{ij^{(n)}}] \\
&= \min(U_{n-2,a} + U_{n-1,a} + U_{n,a}, U_{n-1,a} + U_{n,a}, U_{n,a}) \text{ if } \tau \in (X_{ij^{(n-3)}}, X_{ij^{(n)}}] \\
&= ... \\
&= ... \\
&= \min(U_{n-k-1,a} + F(U_{n-k,a}), F(U_{n-k,a})) \text{ if } \tau \in (X_{ij^{(n-k-1)}}, X_{ij^{(n-k)}}]
\end{aligned}$$

for differing values of $k \in [n]$

We also see from the above that $\hat{\tau}$ must be one of the X_{ij} values for different j.

For the binary greater than setting, we see that the value function is given by:

$$(\hat{\tau}, \hat{\sigma}) := \arg\min_{\tau,\sigma} \sum_{i=1}^{n} U_{ia} I[X_{ij} \geq \tau, X_{ik} \geq \sigma]$$

We again design a binary tree. And we insert U_{ia} into $\bar{r}_j : [n] \to [n]$, where we have that $\bar{r}_j(i)$ is the inverse rank map for $(X_{ij})_{j=1}^n$, where we have that $\bar{r}_j(i)$ is $(t \to n - Rank(X_{ij}) + 1)$. We assume that one needs to take $\bar{r}_k \circ \bar{r}_j$ and correspondingly insert the U_t element accordingly.
As for the node values and the thresholding sums, we would store the greater than or equal to sums respectively. In particular, if we have U to be the node in question and its children given by $V_1, V_2, ..., V_q$, we would have the thresholding sum to be given by:

$$\min_{k \in [q]} \sum_{i=1}^{k} V_{q-i+1}$$

The above can be written as $\min(V_q, V_{q-1} + V_q, V_{q-2} + V_{q-1} + V_q, ..., V_1 + V_2 + V_3 + ... + V_q)$. This would give the same time analysis as for the lesser than equal to scenario.

14.3 Interval case design and analysis

What about the following scenario from the IDTR paper where we have the optimization equation to be given by:

$$(\hat{\tau}, \hat{\sigma}) := \arg\min_{\tau > \sigma} \sum_{i=1}^{n} U_{ia} I[\sigma \leq X_i < \tau]$$

In the above scenario, we define the objective as follows:

$$F(\sigma, \tau) := \sum_{i=1}^{n} U_{ia} I[\sigma \leq X_i < \tau]$$

where the above equation is defined on the $y > x$ hyperplane, i.e $\tau > \sigma$. In this setting, we define the following as:

$$F(\tau | \sigma_0) := \sum_{i=1}^{n} U_{ia} I[\sigma_0 \leq X_i < \tau]$$

In this scenario, it should be noted that there should be a minimal gap between $\tau > \sigma$, otherwise, in the event that $U_{ia} > 0$, we see that the objective is 0 and we get one answer in all regions. Is this why our plots are wrong? All of them involve arrows in a singular direction over the whole state space.

In the above, we see that the domain of the above function is $[\sigma_0, X^{(n)}]$, we assume that $\sigma_0 \in [X_{i^{(l-1)}}, X_{i^{(l)}}]$:

$$\begin{aligned}
F(\tau|\sigma) &= U_{n,a} \text{ if } \tau \in (\sigma_0, X_{i^{(l)}}] \\
&= \min(U_{n-1,a} + U_{n,a}, U_{n,a}) \text{ if } \tau \in (X_{ij^{(n-2)}}, X_{ij^{(n)}}] \\
&= \min(U_{n-2,a} + U_{n-1,a} + U_{n,a}, U_{n-1,a} + U_{n,a}, U_{n,a}) \text{ if } \tau \in (X_{ij^{(n-3)}}, X_{ij^{(n)}}] \\
&= ... \\
&= ... \\
&= \min(U_{n-k-1,a} + F(U_{n-k,a}), F(U_{n-k,a})) \text{ if } \tau \in (X_{ij^{(n-k-1)}}, X_{ij^{(n-k)}}]
\end{aligned}$$

for different values of $k \in [n]$

For the one-parameter case this corresponds to the following algorithmic problem -:

$$(\hat{\sigma}, \hat{\tau}) := \frac{1}{N} \min_{\tau > \sigma} \sum_{i=1}^{n} U_{ia} I[\sigma \leq X_i < \tau]$$

This corresponds to the following algorithmic problem:

- Order $X^{(1)} \leq X^{(2)} \leq X^{(3)} \leq ...X^{(N)}$
- Let $U^{(1)}, U^{(2)}..., U^{(N)}$ using the corr. ordering from $X^{(i)}$, more formally if we assume that we have a rank map $r_X : [N] \to [N]$ where $r_X(i) :=$ Rank of X_i in $X_1, ..., X_N$, then we have the following true where we have $U^{(i)} := r_X^{-1}(U_i)$.
- Then the optimization problem corresponds to the following problem:

  $$\min_{i \in [n], k} \sum_{l=0}^{k} U^{(i+l)}$$

  where in the above we have that $i + k \leq n$
- The above can be thought of as the following problem: Sort $X^{(i)} \forall i \in [n]$ and then get the corresponding $U^{(i)}$ in the same order as $X^{(i)}$. This should take $O(nlogn)$ time. Get an array a, such that $a[i] := U^{(i)}$.
- Now solve using Kadane's algorithm the problem of finding indices $j \geq i$ for an array $a[1, 2, ..., n]$ such that $\sum_{k=i}^{j} a[i]$ is minimized. This takes $O(n)$ time and $(\hat{\sigma}, \hat{\tau}) := (X_{r_X^{-1}(i)}, X_{r_X^{-1}(j)})$ are the minimizing values for the above.
- This solves the problem when we have one condition in our one-parameter estmation problem and it takes $O(nlogn)$ time.

14.4 Interval Estimation Problem : Binary Scenario

Here we list how we would solve the interval estimation problem for the binary scenario when we have to compute the answer to the following objective function:

$$((\hat{\sigma}_1, \hat{\tau}_1), (\hat{\sigma}_2, \hat{\tau}_2)) := \arg_{\sigma_1, \tau_1} \min_{\sigma_2, \tau_2} \sum_{i=1}^{n} U_{ia} I[\sigma_1 \leq X_{ii_1} < \tau_1, \sigma_2 \leq X_{ii_2} < \tau_2]$$

In the following subsection, we assume the rank maps to be the same, i.e if r_{i_1} is the rank map for $(X)_{i,i_1}$ with i ranging from $[0,n]$ and we have r_{i_2} being the rank map for $(X)_{i,i_2}$ with i ranging from $[0,n]$. We can define a notion of equality amongst maps and say the following when $r_{i_1} = r_{i_2}$. If we pick elements $\sigma_2 := X_{i^{(l)}i_2}, \tau_2 := X_{i^{(l+1)}i_2}$ and if we choose $\sigma_1 := X_{i^{(j)}i_1}, \tau_1 := X_{i^{(j+1)}i_2}$. Then

Recall for the 1−D scenario, how we solve the problem in [14.5]. After rearrangement of $(X_{ii_1})_{i\in[n]}$ and of $(X_{ii_2})_{i\in[n]}$ using their order statistics we get $X^{(1)}_{i_1} \leq X^{(2)}_{i_1} \leq ... \leq X^{(n)}_{i_1}$, and we get $X^{(1)}_{i_2} \leq X^{(2)}_{i_2} \leq ... \leq X^{(n)}_{i_2}$, now we need to pick indices $l_1 < r_1$ and $l_2 < r_2$ such that we minimize the sum:

$$\sum_{k\in[l_1,r_1)\cap[l_2,r_2)]} U_{k,a}$$

Since the intersections can be any possible interval $\in [n]$. We see that this corresponds to the sum in Kadane's algorithm itself, i.e., we first solve:

$$\min_{j,k+j\leq n} \sum_{i=j}^{k+j} U_{i,a}$$

Recall from Kadane's algorithm, we can solve the above in $O(nlogn)$ After solving the above, we obtain constants $l < r$ to minimize [14.4]. In this scenario, the minimizing x values turn out to be $X^{(l)}_{i_1}$ and $X^{(r)}_{i_1}$, and the minimizing y values are now $X^{(l)}_{i_2}$ and $X^{(r)}_{i_2}$. The following again takes $O(nlogn)$ time since we sort twice and pick the respective elements. The above analysis can also be extended to the case where we have q conditions involved for this setting, i.e when we have the following optimization problem:

$$\begin{aligned}
&[(\hat{\sigma}_1,\hat{\tau}_1),(\hat{\sigma}_2,\hat{\tau}_2),...,(\hat{\sigma}_q,\hat{\tau}_q)] := \\
\arg_{(\sigma_1,\tau_1),...,} \min_{(\sigma_q,\tau_q)} &\sum_{i=1}^{n} U_{i,a} I[\sigma_1 \leq X_{ii_1} < \tau_1, \\
&\sigma_2 \leq X_{ii_2} < \tau_2, \\
&..., \\
&..., \\
&\sigma_q \leq X_{ii_q} < \tau_q]
\end{aligned}$$

In the event that each of the $(X_{ii_1}, X_{ii_2}, ..., X_{ii_q})$ have the same rank maps $r_{X_{i_k}}$ for all $i \in [n]$, then after a reordering the problem [14.4] can be thought of as:
Given vectors $(a_1[1,2,...n], a_2[1,2,...,n], ..., a_q[1,2,...,n])$ and given a vector $val[1,2,...,n]$. Find $[(\hat{i}_1,\hat{i}_2,...,\hat{i}_q),(\hat{j}_1,\hat{j}_2,...,\hat{j}_q)]$, such t

We would then order $(X_{ii_t})\forall i \in [n]$, and for different values of $t \in [q]$. We then apply Kadane's algorithm to the minimization problem over the U values, which gives us the following minimization problem:

$$\min_{j,k+j\leq n} \sum_{i=j}^{k+j} U_{i,a}$$

After obtaining the coefficients (l,r) that minimize [14.4], we see that we need to choose $((X_{li_1},X_{ri_1}),(X_{li_2},X_{ri_2}),...,(X_{li_q},X_{ri_q}))$ for the different coefficients i_t and for different values of $t \in [q]$ as optimal coefficients. We can compute the total time for the computation in 14.4 as $O(C^d_q qnlogn)$.

14.5 Interval estimation problem : Binary scenario fresh note

The above holds true in the case that we have alignment in rank maps at every coordinate. This need not always hold. Recall the binary version of the interval estimation problem:

$$((\hat{\sigma}_1, \hat{\tau}_1), (\hat{\sigma}_2, \hat{\tau}_2)) := \arg_{\sigma_1,\tau_1} \min_{\sigma_2,\tau_2} \sum_{i=1}^{n} U_{ia} I[\sigma_1 \leq X_{ii_1} < \tau_1, \sigma_2 \leq X_{ii_2} < \tau_2]$$

Let (σ_1, τ_1) and (σ_2, τ_2) be chosen such that:

$$\sigma_1 \leq X_1^{(i_1)} < X_1^{(i_1+1)} < ... < X_1^{(k_1)} < \tau_1$$

In the above, we have the X_1 vector to be $(X_{.i_1})$. If we use the rank map notation for the above what we see is the following:

$$\sigma_1 \leq X_1^{r_1^{-1}(i_1)} < X_1^{r_1^{-1}(i_1+1)} < ... < X_1^{r_1^{-1}(k_1)} < \tau_1$$

We can also obtain a similar inequality involving σ_2 and τ_2, which looks as follows:

$$\sigma_2 \leq X_2^{r_2^{-1}(i_2)} < ... < X_2^{r_2^{-1}(k_2)} < \tau_2$$

Then what we have is the following:

$$\sum_{j \in [r_1^{-1}(i_1),...,r_1^{-1}(k_1)] \cap [r_2^{-1}(i_2), r_2^{-1}(i_2+1),...,r_2^{-1}(k_2)]} U_j$$

The game is now, how would we pick (i_1, k_1) and (i_2, k_2) to minimize the sum in 14.5. Here we would use the concepts of Q trees to simplify our analysis. In the Q-tree analysis, which is done much later, we see that we can compute the answer for when we only have to compute the upper bounds. One idea is to store both the lower and upper thresholding values, and also to store the subtree sum. But on the top node here, we would be storing the answer to minimization and the maximization problems separately, this would correspond to solving the greater, and lesser than equal to problems separately.

Assume the setup with M main node, L left node, and R right node. Recall the conditions we had for the greater than and lesser than conditions, for the upper bound problem we had. For the problem $(X_{ii_1} \leq \sigma)$

$$\min(L.TC, S(L) + R.TC)$$

Similarly for the problem $(X_{ii_1} \geq \sigma)$, we have the problem given by:

$$\min(R.TC, S(R) + L.TC)$$

Maybe the way out would be to store the compounded conditions, which would look as:

$$\min(L.TC, S(L) + R.TC, R.TC, S(R) + L.TC)$$

What about the ordering in the leaf values? Can you active recall the proofs for all three scenarios in the one-parameter setup to get a better picture of why the ordering occurs the way it does? Why does the ordering appear the way it does in the IDTR paper? What about the dual-leaf approach? Store both values. What about the ordering then?

14.6 U values

What about the U-values in this situation, how do we add these to our analysis? - Kadane's algorithm requires us to compute the order statistics for the various $(U_{ia})_{i\in[n]}$. In this situation, we have the U values given by:

$$U_{ia}(X) := (\max_a[r_t(X_{it}, a) + \gamma P_t^a V_{t+1}(X_{it})] - [r_t(X_{it}, a_1) + \gamma P_t^{a_1} V_{t+1}^I(X_{it})] - \eta)\mathcal{I}[X_{it} \notin G_l^t]$$

In the situation where we take the MEU-α score, we see that the U values are given by:

$$U_{ia}(X) := (MEU_\alpha^{\mathcal{M}}(Y_{m,t}(X, a)) - \eta)\mathcal{I}[X_{it} \notin G]$$

In the above equation, we have the following expression true for the $Y_{m,t}$ values.

$$Y_{m,t}(s, a) := (\max_a[r_t(X_{it}, a) + \gamma P_t^a V_{t+1}(X_{it})] - [r_t(X_{it}, a_1) + \gamma P_t^{a_1} V_{t+1}^I(X_{it})] - \eta)\mathcal{I}[X_{it} \notin G]$$

14.7 Q-tree analysis

How does this change when the maximum complexity changes? In particular-when we have the maximum complexity q and we allow for a greater than or equal to t, our objective looks like the following:

$$(\hat{\tau}_1, \hat{\tau}_2, ..., \hat{\tau}_q) := \arg\min_{\tau_1,..,\tau_q} \sum_{i=1}^{n} U_{ia} I[X_{ii_1} \leq \tau_1, ..., X_{ii_q} \leq \tau_q]$$

The algorithm proposed in the paper has time complexity $O(2^q n^q d_t^q m_t)$ and they do this using brute force methods where they first loop over all actions that take - m_t time, then they loop over all q-fold combinations of $[1, 2, ..., n]$ which gives d_t^q factor; we also get the factor 2^q due to the signs added $s_{i_1}, ..., s_{i_q}$. We also loop over all possible τ values in each dimension, which gives us the n^q factor.

14.8 Q-tree Algorithm Design

Assume for the objective in 14.11, we wish to design a Q-tree based approach. Recall a Q-tree is defined as a tree with Q children for each node. That would work as follows:

Create a Q-tree which stores information in the sub-trees, namely the sum of all the sub-trees on a given node. Here we define a Q-tree as a tree with Q nodes at each lengthstep. For equation 14.11, we define the following rank operators $\forall i_j \in [q]$, where we have $j \in [q]$. This looks as follows:

$$r_{i_j} : [n] \to [n]; (t \to \text{Rank}(X_{tj})) \text{ in } X_{\cdot j}$$

In the above, we have $j = (1, 2, ..., q)$, if we follow the same composition operation from before we see that leaf U_{ia} goes to the leaf $r_{i_q} \circ r_{i_{q-1}} ... \circ r_{i_1}$.
In the following, we would store the sum of all the subtrees and the best thresholding sum. In this case, the best thresholding sum would be given by going over the children while adding the thresholding; all the while noting which is minimum. In particular, if we have U as the node in question and its children given by $V_1, V_2, ..., V_q$, then we have the following to be the thresholding sum at each node:

$$\min_{k\in[q]} \sum_{i=1}^{k} V_i$$

14.9 Runtime Analysis

The creation of each rank operator r_{i_t} takes $O(n \log_q n)$ amount of time. Insertion of a U_{ia} in the tree would take $\log_q n$ time, and after insertion, we propagate the sum upward in $O(qn)$ time, and we also update the thresholding sum in $O(qn)$ time. With this knowledge of how to quickly solve the objective in the simplified IDTR paper, we return to the VIDTR.

14.10 Actual Objective: IDTR

In IDTR, we see that the objective function that we need to solve is given by:

$$(\hat{R}, \hat{a}) := \arg \max_{a \in \mathcal{A}, R \in \mathcal{R}} \sum_{i=1}^{n} [(U_{ia} + \zeta/n) I[X_{it} \in R \cap \hat{G}_{tl}] + V_i I[X_{it} \in \hat{G}_{tl} - R]] + \rho c(R)$$

In the above, if we have $\hat{G}_{tl} := \sqcup_{i=1}^{N_{tl}} B_{tl}^i$, a disjoint union of boxes as we identified and computed at each step, then the simplification is:

$$(\hat{R}, \hat{a}) := \arg \max_{a \in \mathcal{A}, R \in \mathcal{R}} \sum_{i=1}^{N} \sum_{j=1}^{N_{tl}} [U'_{ia} I[X_{it} \in R \cap B_{tl}^j] + V_i I[X_{it} \in B_{tl}^j - R] + \rho c(R)]$$

In the above, we have $\hat{G}_{tl} := S - \cup_{k<l} \hat{R}_{tk}$. What is the simplification using the DBU setup? Can you derive a q-tree algorithm for the different complexities here?

14.11 VIDTR objective

In the VIDTR, we see that the objective is given by:

$$(\hat{R}, \hat{a}) := \arg_{R \in \mathcal{R}} \max_{a \in \mathcal{A}} \sum_{i=1}^{N} [\max_{a} (r_t(X_{it}, a) + \gamma P_t^a V_{t+1}(X_{it})) -$$
$$(r_t(X_{it}, a_1) + \gamma P_t^{a_1} V_{t+1}^I)(X_{it}) - \eta] \mathcal{I}[X_{it} \in R - G_l^t] + \rho c(R)$$

In addition to the above, we evaluate

$$\mathcal{I}[X_{it} \in R - G_l^t] := \mathcal{I}[X_{it} \in R] \mathcal{I}[X_{it} \notin G_l^t]$$

In the above, we have $G_l^t := \cup_{j=1}^{l-1} R_j^t$, where we can see that the objective is given by:

$$(\hat{R}, \hat{a}) := \arg_{R \in \mathcal{R}} \max_{a \in \mathcal{A}} \sum_{i=1}^{N} U_{ia}(X) \mathcal{I}[X_{it} \in R]$$

In the above, we have the following.

$$U_{ia}(X) := (\max_{a} [r_t(X_{it}, a) + \gamma P_t^a V_{t+1}(X_{it})] - [r_t(X_{it}, a_1) + \gamma P_t^{a_1} V_{t+1}^I(X_{it})] - \eta) \mathcal{I}[X_{it} \notin G_l^t]$$

How do we obtain conditions when we are allowed to change the bounds on all of the $i_1, i_2, ..., i_q$ coordinates above? How do we solve this problem when we have 2 constants which we need to solve for σ and τ. In particular, how do we solve:

$$(\hat{\tau}_1, \hat{\sigma}_1, \hat{\tau}_2, \hat{\sigma}_2 ..., \hat{\tau}_q) := \arg \min_{\tau_1 > \sigma_1, .., \tau_q > \sigma_q} \sum_{i=1}^{n} U_{ia} I[\sigma_1 \leq X_{ii_1} < \tau_1, ..., \sigma_q \leq X_{ii_q} < \tau_q]$$

In the above, we additionally impose the lower-bound constraints $\hat{\sigma}_1, \hat{\sigma}_2 .., \hat{\sigma}_q$. How do we solve [14.11]? This is what we ask about in the next section. We recall how to design an appropriate data structure to capture the analysis of inequality [14.11] when we have $q = 2$ and only the inequality on the right. My bet is to design a data structure with 'legs' in the back; i.e., it should store 2 cases when ($X_{ii_1} \in [\sigma_1, \tau_1]$) and ($X_{ii_1} < \sigma_1$, and $X_{ii_1} > \tau_1$).
More formally, let us note what the following steps are to prove so that the question above becomes as in an exercise sheet:

- What are the modifications we need to make regarding the data stored in each node? Is it true that we must continue to store the leg sums and the corresponding thresholding sum?
  Ans: Recall what does the IDTR store; the sum of the subtree on the right and the sum of the subtree on the left. It also stores the minimum of the thresholding sum on the left and

- How does the upward propagation occur? If we are storing the sums it is the same as before, but what about the thresholding sum?
  Ans: In the IDTR, the updates are given by $\min(A, A + B)$ where A and B are subtrees of the given node N. Can you dig further into the structure of the IDTR simplification algorithm to derive simplifications?

- Is it true that the arg mins are still attained for the order statistics? Active recall how and why this occurs for the easy case.
  Ans:

- How does addition of the coefficients U_{ia} occur, in particular what are the sorting steps involved for $(X_{ii_1})_{i=1}^n, (X_{ii_2})_{i=1}^n, ..., (X_{ii_q})_{i=1}^n$?
  Ans:

- Compare and contrast with the IDTR paper C/C++ code.
  Ans:

14.12 Objective function under Ambiguity and MEU-α metric

The objective under ambiguity constraints and under the MEU-α scoring metric assumes the same expression as 14.11 with a different expression for U_{ia}. The U_{ia} in this case is given by:

$$U_{ia} := MEU_{\alpha}^{\mathcal{M}}[Y_{m,t}(X_{it}, a)] - \eta$$

14.13 Algorithm for the same?

Can we specialize to the case when we have U_{ia} to be as in the VIDTR?

Can we specialize to the case when we have U_{ia} to be as in the AVIDTR?

14.14 Confidence Interval Estimators for P_t, r_t

14.15 One-parameter to many-parameter extension

$$F(c) := \sum_{i=1}^{N} U_i I[X_{ii_1} \leq c_1, X_{ii_2} \leq c_2, ..., X_{ii_q} \leq c_q]$$

Above we have (X_{ii_k}) to denote the multiple values taken by X in each coordinate.

$$F(\sigma, \tau) := \sum_{i=1}^{N} U_i I[\sigma_1 < X_{ii_1} \leq \tau_1, \sigma_2 < X_{ii_2} \leq \tau_2, ..., \sigma_q < X_{ii_q} \leq \tau_q]$$

Say we pick leaves $U_{l_1}, U_{l_2}, ..., U_{l_k}$ how would we reverse engineer the $(\hat{\sigma}_i)_{i=1}^n$ values that we then pick?

We choose $\hat{\sigma}_1 := X_{r_1^{-1} \circ r_2^{-1} \circ ... \circ r_q^{-1}(l_k), 1}$, we then restrict the subproblem to one with $q - 1$ indices $(X_{.,2}), ..., (X_{.,q})$ to the corresponding subindices. What that looks like here is that we pick

14.16 IDTR Optimization Problem - Active Recall And More

Q) Under what assumptions on the $(X_{j.})$ can we solve:

$$(\hat{c}_1, \hat{c}_2, ..., \hat{c}_q) := \arg \min_{c_1, c_2, .., c_q} \sum_{i=1}^{N} U_i I[X_{i1} \leq c_1, X_{i2} \leq c_2, ..., X_{iq} \leq c_q]$$

in quick time?
If all rank maps $r_i : [n] \to [n]$ are the same, then the tree we create at each timestep would be the same, and we would insert the U_i values according to the common ordering where we insert $U_i \to r(i)$ where the leaf number is $r(i)$ from the left while moving towards the right. In hindsight, this seems like not such a bad assumption provided we assume that the feature space $X_{.,1}, ..., X_{.,q}$ is all positively correlated to one another.
In the case that the rank maps are all 'close' to each other, how would our algorithm perform? Can we come up with good definitions of 'closeness' in our context?

14.17 Inhomogeneous multidimensional state spaces

Can you modify the above such that it works for multidimensional inhomogeneous state spaces? In this case, the problem looks as follows:

$$[(\hat{\tau}_1, \hat{\sigma}_1), (\hat{\tau}_2, \hat{\sigma}_2), ...(\hat{\tau}_q, \hat{\sigma}_q)] := \arg_{\tau_1 > \sigma_1, .., \tau_q > \sigma_q} \min \sum_{i=1}^{N} U_i I[(\sigma_{i1} < X_{i1} \leq \tau_{i1}), ..., (\sigma_{iq} < X_{iq} \leq \tau_{iq})]$$

In the above, X_{ij} would be a vector of dimension d_j. In that case σ_{ij} and τ_{ij} are also vectors of dimension d_j. Inequalities $<$ and $\leq$ in this case would be pointwise in all of the vector dimensions.

Can you copy the files on colab and modify the optimization algorithm to handle the inhomogeneous case?

14.18 Potential issues with the VIDTR

Here we consider scenarios where the VIDTR could potentially yield pitfalls. This in particular applies to cases where $\max_\alpha [r_t(s, \alpha) + \gamma P_t^\alpha V_{t+1}(s)] - [r_t(s, a) + \gamma P_t^a V_{t+1}(s)]$ is small. In this case, almost all positive values of the constant η we would have the U_i values be negative. In particular, we have:

$$U_{iat} := [\max_\alpha [r_t(s, \alpha) + \gamma P_t^\alpha V_{t+1}(s)] - [r_t(s, a) + \gamma P_t^a V_{t+1}(s)] - \eta] \mathcal{I}[X_{it} \notin G_{tl}]$$

which would have us choose the whole space in the first part of the optimization itself. The way to get out of this is to do some global optimization..is that possible? How would that look like?

15 Interpretable ML to interpretable RL

The interpretable ML problem is given as follows:
Given input data $(x_i, y_i)_{i=1}^N$, where $x_i \in \mathcal{X}$ are features and $y_i \in \mathcal{Y}$ are outputs, how do we find an 'interpretable' map $\hat{f} : \mathcal{X} \to \mathcal{Y}$ that represents the data distribution?
In particular, if $\mathcal{F}_{int} \subset \mathcal{F} := \{f : \mathcal{X} \to \mathcal{Y}\}$ where $\mathcal{F}_{int}$ is the subspace of interpretable functions in the space of all functions from $\mathcal{X} \to \mathcal{Y}$, how do we solve:

$$\arg_{g \in \mathcal{F}_{int}} \min L(f, g, \alpha) + R(g)$$

where L is the loss function in the function space, α is a parameter and R is a regularization function.

The intepretable RL problem can be thought of as follows: Given $\{(X_{it}, A_{it}, Y_{it})\}_{i=1}^N$ the trajectory data for N trajectories where we believe the action is taken by an expert, how do we come up with an 'interpretable' policy $\hat{\pi}$ that approximates $(X_i \to A_i)$ for all i.

The IDTR paper aims to do the following: Given trajectory data, obtain estimates of the Q-values using backwards induction on the regularized Bellman equations. This works as follows:

$$Q_T \sim ((X_{.T}, A_{.T}) \to Y_{.T})$$

In the above $\sim$ denotes a regression method we wish to use in order to derive estimates of the following at the last timestep.
For general s, if we have estimated the $Q-$values for all times $t > s$, we need to solve the following regression problem:

$$Q_s \sim ((X_{.s}, A_{.s}) \to Y_{.s} + \gamma Q_{s+1}(X_{.,s+1}, A_{.,s+1}))$$

With the knowledge of the Q_t-values, we would aim to estimate the following

$$s \to \arg_{a \in \mathcal{A}} \max Q_t(s, a)$$

using an interpretable ML technique such as CART or CORELS. Here we define the loss function for a candidate policy π to be given by :

$$L_t(\hat{\pi}_t, \pi_t) := \int_{s \in S} [Q_t(s, \hat{\pi}_t(s)) - Q_t^I(s, \pi_t(s))] ds + \lambda R(\pi)$$

In the above we have that $Q_t^I(s, a) := r_t(s, a) + \gamma Q_{t+1}^I(s, a)$. For $t = T$, we see that the loss function is:

$$L_T(\hat{\pi}_T, \pi_T) := \int_{s \in S} [r_T(s, \hat{\pi}_T(s)) - r_T(s, \pi_T(s))] ds + \lambda R(\pi)$$

In this setting, every interpretable RL method would involve t choices for the regression in Q_t values and t choices for computation of the interpretable policies π_t.

An interesting question to ask here how the interpretable ML methods would adapt to this setting.

15.1 Rule based greedy splits

For example for a rule based splitting greedy method as in the IDTR paper, the function approximation for a function f would work as follows:

$$\begin{aligned} f(x) &= a_1 \text{ if } x \in R_1 \\ &= f(x) \text{ if } x \notin R_1 \end{aligned}$$

The error would then be evaluated as $\int_{x \in R_1} L(f(x), a_1)dx - \eta V(R_1) + \rho c(R_1)$ For the general lengthstep l, we have the following to be true:

$$\begin{aligned} f(x) &= a_1 \text{ if } x \in R_1 \\ &= a_2 \text{ if } x \in R_2 - R_1 \\ &= ... \text{ ....} \\ &= a_l \text{ if } x \in R_l - \cup_{j<l} R_j \\ &= a \text{ if } x \in R - \cup_{i=1}^{l} R_i \\ &= f(x) \text{ if } x \in S - R - \cup_{i=1}^{l} R_i \end{aligned}$$

The lth lengthstep error is given by $\int_{x \in R - \cup_{i=1}^{l} R_i} L(f(x), a)dx - \eta V(R - \cup_{i=1}^{l} R_i) + \rho c(R)$.
The loss function in the space of policies Π is given by:

$$L_t(\pi, \pi')(s) := ||Q_t(s, \pi(s)) - Q_t(s, \pi'(s))||$$

The notation $\pi := a$ means the policy that takes the constant action a on S.
The lth lengthstep and tth timestep trajectory integral error is given as:

$$\sum_{i=1}^{N} [\hat{Q}_t(X_{it}, \hat{\pi}_t(X_{it})) - \hat{Q}_t(X_{it}, a) - \eta] \mathcal{I}[X_{it} \in R - \cup_{i=1}^{l} R_i] + \rho_t c(R)$$

One can write the above as a Mixed Integer Linear Program and proceed to do the simplifications per length and timestep. Is there a common framework in which you can frame this for the different time and lengthsteps?

15.2 CART algorithm based splitting

For the CART algorithm, we can do splitting at each node based on the feature vector. Say we have data $\mathcal{D} = [(x_i, y_i)]_{i=1}^{N}$ where x_i are the input vectors, and y_i are the outputs belonging to a discrete space $\mathcal{A}$.

We would try out all possible splits and pick the one which would minimize the $Q-$score based loss; For instance, if we pick the condition $s \leq c_1$, the data would get split into $\mathcal{D}_1 := \{(s, \pi(s)) | s_1 \leq c_1\}$ and $\mathcal{D}_2 := \{(s, \pi(s)) | s_1 > c_1\}$. Let the most frequently occuring element in $\mathcal{D}_1$ be a_1, and the most frequently occuring element in $\mathcal{D}_2$ be a_2. We then see that Q-loss is given by:

$$\begin{aligned} \mathcal{L}_t[(s_1 \leq c_1)] := &\frac{1}{|S_1|} \mathbb{E}[||\hat{Q}_t(s, \hat{\pi}(s)) - Q_t(s, a_1)||] + \\ &\frac{1}{|S - S_1|} \mathbb{E}[||\hat{Q}_t(s, \hat{\pi}(s)) - Q_t(s, a_2)||] \end{aligned}$$

We would then go over all possible splits of the form $[a_{i_1} < x_{i_1} \leq b_{i_1}] \wedge [a_{i_2} < x_{i_2} \leq b_{i_2}] ... \wedge [a_{i_k} < x_{i_k} \leq b_{i_k}]$; evaluate the one with the minimal loss and proceed. We can improve this by storing appropriate lower bounds and deciding to not branch when the score is high enough.

15.3 Optimal Classification Trees - by Bertsimas

In this paper, they come up with a globally optimal way to construct a decision tree. In this setup we minimize the following objective function:

$$\min_{T \in \mathcal{T}} [R_{xy}(T) + \alpha |T|]$$

The authors of the Optimal Classification Trees come up with a MIO based modeling approach to solve the above problem. In particular, they note that the building of a complete decision tree can

be described by a MIO formulation. They come up with constraints and objectives to describe that formulation, we see that most of the formulation holds (these describe how a tree is built with the MIO approach) However the objective needs to be tweaked which is what we proceed to describe in the following scenario.

$$\min \frac{1}{\widehat{L}} \sum_{t \in \mathcal{T}_L} L_t + \alpha \sum_{t \in \mathcal{T}_B} d_t$$

such that

$$L_t = \sum_{i=1}^{N} z_{it}[Q_t(x_i, \pi(x_i)) - Q_t(x_i, a)]$$

$$a_m^T x_i \geq b_t - (1 - z_{it}) \forall i \in [n]$$

$$a_m^T (x_i + \epsilon) \leq b_t + (1 + \epsilon_{max})(1 - z_{it}) \forall i \in [n], \forall t \in \mathcal{T}_B, \forall m \in [p]$$

$$\sum_{t \in \mathcal{T}_L} z_{it} = 1, \forall i \in [n]$$

$$z_{it} \leq l_t \forall t \in \mathcal{T}_L$$

$$\sum_{i=1}^{n} z_{it} \geq N_{min} l_t \forall t \in \mathcal{T}_L$$

$$\sum_{j=1}^{p} a_{jt} = d_t \forall t \in \mathcal{T}_B$$

$$b_t \in [0, d_t] \forall t \in \mathcal{T}_B$$

$$d_t \leq d_{p(t)} \forall t \in \mathcal{T}_B$$

$$z_{it}, l_{it} \in \{0, 1\}, \forall i \in [n], t \in \mathcal{T}_B$$

In the above, the equations mean the following:

1. $[x_i, \pi(x_i)]_{i=1}^N :=$ The points on the state space and the corr. policy values.
2. $z_{it} := \mathcal{I}[x_i \in \text{leaf}(t)]$
3. $Q_t(x, a)$: Estimated Q-values which is obtained after the reverse induction.
4. $l_t = 1$: If there is a leaf in node t, else this is 0.
5. $N_{min} :=$ Minimum number of points in a leaf.
6. $a_i, b_i :=$ The split at node t is given by $a_i^T x < b_i$ or $a_i^T x \geq b_i$. Here $a = 0$ would denote no splitting at node t. For a split, exactly one a_{jt} is 1, this implies we split at variable j.
7. $d_t := \mathcal{I}[\text{node } t \text{ applies a split}]$
8. $\pi_t : S_t \to \mathcal{A}_t :=$ The estimated optimal policy

16 Gradient based optimization techniques

In this section, we present gradient based optimization techniques to solve the given optimization problem:

$$\sum_{i=1}^{N}[\max_{\alpha} z(r_t(X_{it},\alpha)+\gamma P_t^{\alpha}V_{t+1}(X_{it}))-(r_t(X_{it},a_1)+\gamma P_t^{a_1}V_{t+1}^{I}(X_{it}))-\eta]\mathcal{I}[X_{it}\in R]\mathcal{I}[X_{it}\notin G_l^t]$$

Further splitting of the indicator term gives us the following:

$$\Pi_{j=1}^{d}\mathcal{I}[a_j\leq(X_{it})_j\leq b_j]$$

If we assume all the terms $(X_{it})_j\in[a_j,b_j]$ for all $j\in[2,d]$ and if we assume to look at the behavior of the function:

$$a_1\to\mathcal{I}[a_1\leq(X_{it})_1\leq b_1]$$

we see that it is 1 for $a_1<(X_{it})_1$ and 0 for $a_1\geq(X_{it})_1$. We can then see that this is a non-differentiable function with the derivatives being 0 for most points in the domain. So proceeding with a gradient-based technique makes no sense in this setting.
How does this look like in the integral setting?

$$\int_{\vec{a}}^{\vec{b}}[\max_{\alpha}(r_t(s,\alpha)+\gamma P_t^{\alpha}V_{t+1}(s))-(r_t(s,a_1)+\gamma P_t^{a_1}V_{t+1}^{I}(s))-\eta]ds+\rho c(R)$$

Taking the derivative of [16] with respect to b_1, we see that we get the following:

$$\int_{a_2}^{b_2}\int_{a_3}^{b_3}...\int_{a_d}^{b_d}[\max_{\alpha}(r_t((b_1,s_2,...,s_d),\alpha)+\gamma P_t^{\alpha}V_{t+1}((b_1,s_2,...,s_d))-$$
$$(r_t((b_1,s_2,...,s_d,a_1)+\gamma P_t^{a_1}V_{t+1}^{I}((b_1,s_2,...,s_d))-\eta]ds=0$$

We can think of the above problem as follows:

Given N points in d-space each of which has a value U_i attached to it. How do we find a box that encloses the set of points with the least sum?

17 Non-convex approximation techniques

Recall the problem we want to find a solution to is given by:

$$(R,a)\mapsto\int_{R-S}\max_{\alpha}[r_t(s,\alpha)+\gamma P_t^{\alpha}V_{t+1}(s)-(r_t(s,a)+\gamma P_t^{a}V_{t+1}(s))]ds-\eta V(R-S)+\rho c(R)$$

In the trajectory integral notation, the problem looks like:

$$(R,a)\mapsto\sum_{i=1}^{N}(\max_{\alpha}[r_t(X_{it},\alpha)+\gamma P_t^{\alpha}V_{t+1}(X_{it})]-(r_t(X_{it},A_{it})+\gamma P_t^{a}V_{t+1}(s))-\eta)\mathcal{I}[X_{it}\in R-S]+\rho c(R)$$

When split what we obtain is the following:

$$(R,a)\mapsto\sum_{i=1}^{N}(\max_{\alpha}[r_t(X_{it},\alpha)+\gamma P_t^{\alpha}V_{t+1}(X_{it})]-(r_t(X_{it},A_{it})+\gamma P_t^{a}V_{t+1}(s))-\eta)\mathcal{I}[X_{it}\notin S]\mathcal{I}[X_{it}\in R]+\rho c(R)$$

We can modify the above by expanding the $\mathcal{I}[X_{it}\in R]$ term as:

$$\mathcal{I}[X_{it}\in R]:=\Pi_{j=1}^{d}\mathcal{I}[(X_{it})_j\in[a_j,b_j]]$$

We then see that the above problem consists of maximizing over a product of indicator functions. Bertismas has theory on how to optimize the objective when it is a product or sum of convex functions.

In our setting we would approximate the indicator using a sequence of convex functions and then solve the corr. optimization problem. Can we do this? In particular is there an arbitarily close convex approximation for the indicator function?
This idea does not work; Assume f to be convex and approximates the $0-1$ function z, where we have $z(x) = 0 \forall x \leq 0$ and $z(x) = 1 \forall x > 0$.
Then we have that $f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y)$. In the following if we have $x = 0$ and $y = 1$, we get:

$$f(0\lambda + (1-\lambda)y) \leq \lambda f(0) + (1-\lambda) f(1)$$

The above shows us that:

$$f(1-\lambda) \leq (1-\lambda)$$

for $\lambda \in [0, 1]$ This prevents f from being close to z.

18 Sorting algorithm in IDTR

We recall the sorting algorithm from the IDTR paper that is used to compute the optimal regions to split at and the optimal actions. They claim that the objective looks like the following in the supplementary material of the original IDTR paper:

$$(\hat{R}_{tl}, \hat{a}_{tl}) := \arg_{R \in \mathcal{R}_t} \min_{a \in \mathcal{A}_t} \frac{1}{n} \sum_{i=1}^{n} U_{ia} I(X_i \in R) - \eta(2 - V(R))$$

However, the notes from the original paper say otherwise; here the equation which we are minimizing looks like the following:

$$(\hat{R}_{tl}, \hat{a}_{tl}) := \arg_{R \in \mathcal{R}_t} \min_{a \in \mathcal{A}_t} \frac{1}{n} \sum_{i=1}^{n} [I(X_i \in G, X_i \in R) C_{i,a} + I(X_i \in G, X_i \notin R) C'_{i,a}] + \eta\rho(R)$$

Can we simplify the expression in 18 to make it look like the expression in 18? Yes, we must be able to. The simplification looks as follows:

$$(\hat{R}_{tl}, \hat{a}_{tl}) := \arg_{R \in \mathcal{R}_t} \min_{a \in \mathcal{A}_t} \frac{1}{n} \sum_{i=1}^{n} I(X_i \in G)[I(X_i \in R) C_{i,a} + [1 - I(X_i \in R)] C'_{i,a}] + \eta\rho(R)$$

We can further simplify the above as follows:

$$(\hat{R}_{tl}, \hat{a}_{tl}) := \arg_{R \in \mathcal{R}_t} \min_{a \in \mathcal{A}_t} \frac{1}{n} \sum_{i=1}^{n} I(X_i \in G)[I(X_i \in R)(C_{i,a} - C'_{i,a}) + C'_{i,a}] + \eta\rho(R)$$

The above can further be written as:

$$(\hat{R}_{tl}, \hat{a}_{tl}) := \arg_{R \in \mathcal{R}_t} \min_{a \in \mathcal{A}_t} \frac{1}{n} \sum_{X_i \in G} [I(X_i \in R) U_{i,a} + C'_{i,a}] + \eta\rho(R)]$$

In this case, what we see is that the final objective is given by:

$$(\hat{R}_{tl}, \hat{a}_{tl}) := \arg_{R \in \mathcal{R}_t} \min_{a \in \mathcal{A}_t} \frac{1}{n} \left[\left[\sum_{X_i \in G} U_{i,a} I(X_i \in R) \right] + C'_{i,a} |G| + \eta\rho(R) \right]$$

Above we see that the step that is the most computationally expensive is the one in which we compute the arg min over the total set of conditions to find the optimal region and action over all possible regions and actions. Under the assumption that the transition and reward kernels are close in time t

References

[1] S. Alowais, S. Alghamdi, N. Alsuhebany, T. Alqahtani, A. Alshaya, S. Almohareb, A. Aldairem, M. Alrashed, K. Saleh, H. Badreldin, M. Al Yami, S. Al Harbi, and A. Albekairy, *Revolutionizing healthcare: the role of artificial intelligence in clinical practice*, BMC Medical Education, 23 (2023).

[2] G. Bacci, A. Ingólfsdóttir, K. Larsen, and R. Reynouard, *Active learning of markov decision processes using baum-welch algorithm (extended)*, arXiv preprint arXiv:2110.03014, (2021).

[3] J. Bajwa, U. Munir, A. Nori, and B. Williams, *Artificial intelligence in healthcare: transforming the practice of medicine*, Future Healthcare Journal, 8 (2021), pp. e188–e194.

[4] O. Bastani, Y. Pu, and A. Solar-Lezama, *Verifiable reinforcement learning via policy extraction*, in Proceedings of the 32nd International Conference on Neural Information Processing Systems, NIPS'18, Red Hook, NY, USA, 2018, Curran Associates Inc., p. 2499–2509.

[5] B. Chakraborty and S. A. Murphy, *Dynamic treatment regimes*, Annu. Rev. Stat. Appl., 1 (2014), pp. 447–464.

[6] T. H. Davenport and R. Kalakota, *The potential for artificial intelligence in healthcare*, Future Healthcare Journal, 6 (2019), pp. 94 – 98.

[7] D. Ernst, P. Geurts, and L. Wehenkel, *Tree-based batch mode reinforcement learning*, Journal of Machine Learning Research, 6 (2005).

[8] J. Grand-Clément, C. Chan, V. Goyal, and E. Chuang, *Interpretable machine learning for resource allocation with application to ventilator triage*, arXiv preprint arXiv:2110.10994, (2021).

[9] ——, *Interpretable machine learning for resource allocation with application to ventilator triage*, arXiv preprint arXiv:2110.10994, (2021).

[10] U. D. Gupta, E. Talvitie, and M. Bowling, *Policy tree: adaptive representation for policy gradient*, in Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, AAAI'15, AAAI Press, 2015, p. 2547–2553.

[11] D. Hein, S. Udluft, and T. A. Runkler, *Interpretable policies for reinforcement learning by genetic programming*, Engineering Applications of Artificial Intelligence, 76 (2018), pp. 158–169.

[12] Y. Huang, R. Cao, and A. Rahmani, *Reinforcement learning for sepsis treatment: A continuous action space solution*, in Proceedings of the 7th Machine Learning for Healthcare Conference, Z. Lipton, R. Ranganath, M. Sendak, M. Sjoding, and S. Yeung, eds., vol. 182 of Proceedings of Machine Learning Research, PMLR, 05–06 Aug 2022, pp. 631–647.

[13] N. N. Khanna, M. A. Maindarkar, V. Viswanathan, J. F. E. Fernandes, S. Paul, M. Bhagawati, P. Ahluwalia, Z. Ruzsa, A. Sharma, R. Kolluri, I. M. Singh, J. R. Laird, M. Fatemi, A. Alizad, L. Saba, V. Agarwal, A. Sharma, J. S. Teji, M. Al-Maini, V. Rathore, S. Naidu, K. Liblik, A. M. Johri, M. Turk, L. Mohanty, D. W. Sobel, M. Miner, K. Viskovic, G. Tsoulfas, A. D. Protogerou, G. D. Kitas, M. M. Fouda, S. Chaturvedi, M. K. Kalra, and J. S. Suri, *Economics of artificial intelligence in healthcare: Diagnosis vs. treatment*, Healthcare (Basel), 10 (2022), p. 2493.

[14] P. W. Lavori, R. Dawson, and D. Shera, *A multiple imputation strategy for clinical trials with truncation of patient data*, Stat. Med., 14 (1995), pp. 1913–1925.

[15] A. Likmeta, A. M. Metelli, A. Tirinzoni, R. Giol, M. Restelli, and D. Romano, *Combining reinforcement learning with rule-based controllers for transparent and general decision-making in autonomous driving*, Robotics Auton. Syst., 131 (2020), p. 103568.

[16] Y. Liu, B. Logan, N. Liu, Z. Xu, J. Tang, and Y. Wang, *Deep reinforcement learning for dynamic treatment regimes on medical registry data*, in 2017 IEEE International Conference on Healthcare Informatics (ICHI), 2017, pp. 380–385.

[17] ——, *Deep reinforcement learning for dynamic treatment regimes on medical registry data*, in 2017 IEEE international conference on healthcare informatics (ICHI), IEEE, 2017, pp. 380–385.

[18] F. Maes, R. Fonteneau, L. Wehenkel, and D. Ernst, *Policy search in a space of simple closed-form formulas: Towards interpretability of reinforcement learning*, in Discovery Science, J.-G. Ganascia, P. Lenca, and J.-M. Petit, eds., Berlin, Heidelberg, 2012, Springer Berlin Heidelberg, pp. 37–51.

[19] A. Orfanoudaki, S. Saghafian, K. Song, H. A. Chakkera, and C. Cook, *Algorithm, human, or the centaur: How to enhance clinical care?*, (2022).

[20] M. Qian and S. Murphy, *Performance guarantees for individualized treatment rules*, Annals of statistics, 39 (2011), pp. 1180–1210.

[21] J. M. Robins, *Optimal structural nested models for optimal sequential decisions*, in Proceedings of the Second Seattle Symposium in Biostatistics: analysis of correlated data, Springer, 2004, pp. 189–326.

[22] S. Ross, G. Gordon, and D. Bagnell, *A reduction of imitation learning and structured prediction to no-regret online learning*, in Proceedings of the fourteenth international conference on artificial intelligence and statistics, JMLR Workshop and Conference Proceedings, 2011, pp. 627–635.

[23] S. Saghafian and L. Idan, *Effective generative ai: The human-algorithm centaur*, arXiv preprint arXiv:2406.10942, (2024).

[24] ——, *Effective generative ai: The human-algorithm centaur*, arXiv preprint arXiv:2406.10942, (2024).

[25] A. Silva, T. Killian, I. D. J. Rodriguez, S.-H. Son, and M. Gombolay, *Optimization methods for interpretable differentiable decision trees in reinforcement learning*, arXiv preprint arXiv:1903.09338, (2019).

[26] J. Tarbouriech, S. Shekhar, M. Pirotta, M. Ghavamzadeh, and A. Lazaric, *Active model estimation in markov decision processes*, in Conference on Uncertainty in Artificial Intelligence, PMLR, 2020, pp. 1019–1028.

[27] N. Topin, S. Milani, F. Fang, and M. Veloso, *Iterative bounding mdps: Learning interpretable policies via non-interpretable methods*, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, 2021, pp. 9923–9931.

[28] D. Vos and S. Verwer, *Optimal decision tree policies for markov decision processes*, arXiv preprint arXiv:2301.13185, (2023).

[29] X. Wu, R. Li, Z. He, T. Yu, and C. Cheng, *A value-based deep reinforcement learning model with human expertise in optimal treatment of sepsis*, npj Digital Medicine, 6 (2023).

[30] T. Xu, H. Zhu, and I. C. Paschalidis, *Learning parametric policies and transition probability models of markov decision processes from data*, European Journal of Control, 57 (2021), pp. 68–75.

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

[32] Y. Zhang, E. B. Laber, M. Davidian, and A. A. Tsiatis, *Interpretable dynamic treatment regimes*, Journal of the American Statistical Association, 113 (2018), pp. 1541–1549.

[33] ——, *Interpretable dynamic treatment regimes*, J. Am. Stat. Assoc., 113 (2018), pp. 1541–1549.