Markov Decision Processes Lecture Notes 07

Linear Programming

Authors: Benjamin Van Roy *November 3, 2025*

1 The Optimal Discounted Value Function

Consider an MDP $(\mathcal{S}, \mathcal{A}, P)$, a reward function r, and a discount factor γ. We study the linear program

$$\begin{array}{ll} \min_V & \sum_{s\in\mathcal{S}} V(s) \\ \text{s.t.} & V(s) \geq r(s,a) + \gamma \sum_{s'\in\mathcal{S}} P_{ass'} V(s') \qquad \forall s \in \mathcal{S}, a \in \mathcal{A}. \end{array} \tag{1}$$

As we will see, the unique optimal solution is $V_{*,\gamma}$.

1.1 Concise Expression

The linear program (1) can be expressed in a more concise manner in terms of a nonlinear optimization problem. This is because the feasible region identified by one linear constraint per state-action pair can instead be identified by one nonlinear constraint per state. This leads to the optimization problem:

$$\begin{array}{ll} \min_V & \sum_{s\in\mathcal{S}} V(s) \\ \text{s.t.} & V(s) \geq \max_{a\in\mathcal{A}} \left(r(s,a) + \gamma \sum_{s'\in\mathcal{S}} P_{ass'} V(s')\right) \qquad \forall s \in \mathcal{S}. \end{array} \tag{2}$$

The reason (2) is equivalent to (1) is that requiring $V(s)$ to be larger than some action-dependent quantity for each action is equivalent to requiring that it be larger than the maximum over those quantities. Note that the right-hand-side of each constraint in (2) can be rewritten as $(TV)(s)$. Hence, the optimization problem can be expressed very concisely as

$$\begin{array}{ll} \min_V & \sum_{s\in\mathcal{S}} V(s) \\ \text{s.t.} & V \geq TV. \end{array} \tag{3}$$

1.2 Analysis of the Optimal Solution

The following theorem establishes that the unique solution is $V_{*,\gamma}$.

Theorem 1. *Fix a finite-state finite-action MDP $(\mathcal{S}, \mathcal{A}, P)$, a reward function $r : \mathcal{S}\times\mathcal{A} \to \mathbb{R}$, and a discount factor $\gamma \in [0, 1]$. $V_{*,\gamma}$ is the unique optimal solution to the linear program (1).*

Proof. Recall that the feasible set be written as $V \geq TV$. By monotonicity,

$$V \geq TV \geq T^2V \geq \cdots \geq V_{*,\gamma}.$$

Further, since $V_{*,\gamma} = TV_{*,\gamma}$, $V_{*,\gamma}$ is in the feasible set. The result follows. □

1.3 Toy Example

To offer a more concrete understanding of how the linear program leads to an optimal value function, consider the simple MDP illustrated in Figure 1. There are two states $\mathcal{S} = \{1, 2\}$. There are two actions $\mathcal{A} = \{\texttt{stay}, \texttt{move}\}$. The `stay` action self-transitions with probability $1 - \epsilon$, while the `move` action transitions to the other state with probabilty $1 - \epsilon$. The reward is $r(s, a) = 1$ if $s = 1$ and $r(s, a) = 0$ if $s = 2$. Let $\epsilon = 0.1$ and $\gamma = 0.9$. The optimal policy chooses `stay` at state 1 and `move` at state 2.

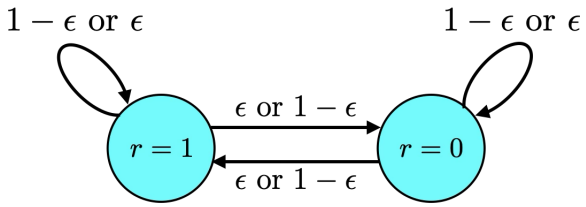

Figure 1: A simple MDP with two states and two actions. In each state, the `stay` action leads to a self-transition with probability $1 - \epsilon$. The reward is 1 when at the first state and 0 when at the second.

For this simple MDP, the linear program (1) can be written concretely as

$$\begin{array}{ll} \min_V & V(1) + V(2) \\ \text{s.t.} & V(1) \geq 1 + \gamma(1-\epsilon)V(1) + \gamma\epsilon V(2) \\ & V(1) \geq 1 + \gamma\epsilon V(1) + \gamma(1-\epsilon)V(2) \\ & V(2) \geq \gamma\epsilon V(1) + \gamma(1-\epsilon)V(2) \\ & V(2) \geq \gamma(1-\epsilon)V(1) + \gamma\epsilon V(2). \end{array} \tag{4}$$

For each state, there is one linear constraint per action. Figure 2 illustrates the feasible region. There is one line per constraint. The shaded grey area is the feasible region. The objective selects the vertex closest to the origin, which is where the blue and red lines meet. These lines correspond to the first and fourth constraints, which are associated with state-action pairs $(1, \texttt{stay})$ and $(2, \texttt{move})$. Since these constraints are binding at the optimal solution $V_{*,\gamma}$, the actions are greedy with respect to $V_{*,\gamma}$. Therefore, it is optimal to select `stay` at state 1 and `move` at state 2.

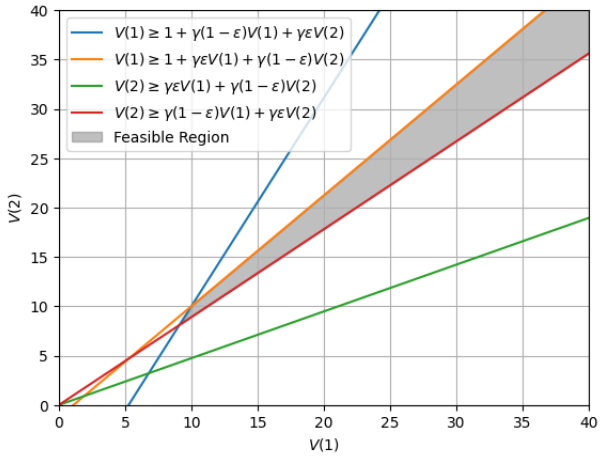

Figure 2: The feasible region for the simple MDP with $\epsilon = 0.1$ and $\gamma = 0.9$.

1.4 Queueing Example

Figure 3 plots an optimal solution to the linear program for the queueing example studied in previous lectures, which allows up to twenty customers to wait in the queue. This recovers the same optimal value function $V_{*,\gamma}$ as we obtained, for example, via policy iteration.

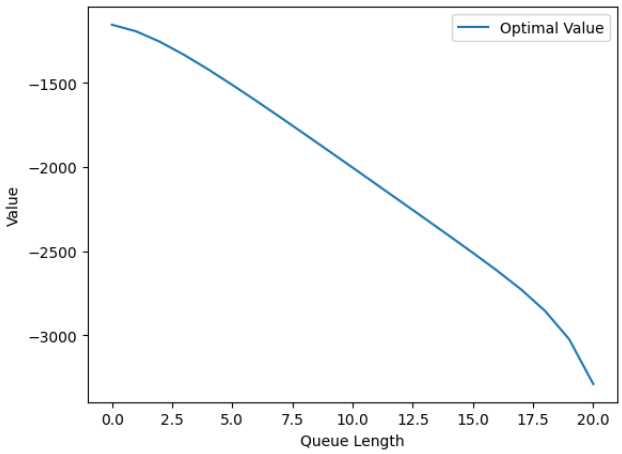

Figure 3: Discounted optimal value function for the queueing system, computed via linear programming.

1.5 State-Relevance Weights

The objective $\sum_{s\in\mathcal{S}} V(s)$ of the linear program (1) sums values across states and thus treats each state identically. Intuitively, one may think that it is more important to get values at some states right more than others. This suggests weighting state-values in this objective. In particular, consider a weighted objective $\sum_{s\in\mathcal{S}} w(s)V(s)$, where $w \in \mathbb{R}^{\mathcal{S}}_{++}$ is a vector of state-relevance weights.

It is easy to verify that the proof of Theorem 1 goes through for any (positive) state relevance weights. Hence, the optimal solution to a weighted optimization problem

$$\begin{array}{ll} \min_V & \sum_{s\in\mathcal{S}} w(s)V(s) \\ \text{s.t.} & V \geq TV, \end{array} \tag{5}$$

is again $V_{*,\gamma}$. As such, state-relevance weights do not change the optimal solution.

Figure 2 facilitates interpretation of this property in our toy MDP. If we draw a negative weight vector $-w$ but originating at the optimal vertex of the feasible region, we see that the vector points outside the feasible region. Hence, there is no vector V within the feasible region that obtains a smaller weighted-objective value.

1.6 Tetris

Farias and Van Roy [2006] applied a variation of the linear program to a version of Tetris. In this version, tetrominoes are positioned by rotation and translation as they fall onto a wall made up of previous ones. Each tetromino is made up of four equally-sized bricks, and the Tetris board is a two-dimensional grid, ten-bricks wide and twenty-bricks high. Each tetromino takes on one of seven possible shapes. A point is received for each row constructed without any holes, and the corresponding row is cleared. The game terminates once the height of the wall exceeds 20. The objective is to maximize the expected number of points accumulated over the course of the game. A representative mid-game board configuration is illustrated in Figure **??**.

To model this game as an MDP, we take the state S_t to encodes the wall configuration and the shape of the falling tetromino. The action A_t encodes the rotation and translation applied to the falling tetromino. It is natural to consider the reward associated with a state-action pair to be the number of points received as a consequence of the action. However, Farias and Van Roy [2006] obtained better results by taking the reward to be the height of the current wall, and a reward of $-20/(1-\gamma)$ upon termination. They took the objective to be discounted expected return with this reward function, with discount factor $\gamma = 0.9$. With this formulation, an optimal policy maximizes the number of rows cleared prior to termination with a greater emphasis on the immediate future, due to discounting.

Figure 4: Tetris board.

In principle, we could solve the linear program (1) to obtain an optimal policy for Tetris. However, this would not be computationally feasible, as this linear program would present too many constraints, and its objective would require summing over too many values.

In a spirit similar to approximate policy iteration, we consider a parameterized value function. In particular, let

$$\tilde{V}_\theta(s) = \sum_{k=1}^{K} \phi_k(s), \tag{6}$$

where $K = 22$ and $\phi_1, \ldots, \phi_K$ are fixed functions defined as follows:

- $\phi_1, \ldots, \phi_{10}$ map the state to the height of each of the ten columns,
- $\phi_{11}, \ldots, \phi_{19}$ map the state to the absolute difference between heights of successive columns,
- ϕ_{20} maps the state to the maximum column height,
- ϕ_{21} maps the state to the number of 'holes' in the wall,
- ϕ_{22} is equal to one at every state.

While use of a parameterized approximation allows us to express the value function via a vector $\theta \in \mathbb{R}^{22}$, the objective $\sum_{s\in\mathcal{S}} \tilde{V}_\theta(s)$ still requires summing over too many terms. However, if we apply state relevance weights $w \in \mathbb{R}^{\mathcal{S}}_{++}$ that sum to one, we can approximate the weighted objective $\sum_{s\in\mathcal{S}} w(s)\tilde{V}_\theta(s)$ via Monte Carlo sampling. In particular, sample M states $s_1, \ldots, s_M$ iid from w, which can be interpreted a probability distribution since components are positive and sum to one. Then, use an approximate objective $\sum_{m=1}^{M} w(m)\tilde{V}_\theta(s_m)$.

The aforementioned Monte Carlo approach gives rise to an objective that we can easily compute. But another issue is that, for the game of Tetris, the linear program presents too many constraints. We address this by approximating the feasible region. In particular, we retaining only constraints associated with the sampled states $s_1, \ldots, s_M$ and relax the remaining constraints.

Rather than literally specify state relevance weights w, we sampled states by simulating the game and using every 90th state visited over the course of play. We began with a simulation under a simple heuristic policy arrived at by hand-selecting parameters $\theta \in \mathbb{R}^{22}$ and applying greedy actions with respect to $\tilde{V}_\theta$. We also tried "bootstrapping" this process. In particular, after solving the approximate linear program to obtain new parameters $\theta \in \mathbb{R}^{22}$, we would simulate again and solve another linear program. This process can be repeated any number of times.

Figure 5 plots the average score attained by greedy policies. The horizontal axis indicates the number of bootstrapping iterations applied to arrive at the policy. There are two curves. One presents averages over many runs. The other presents results from the best run. Each point in each curve is the result of simulating many games under a fixed policy and averaging scores.

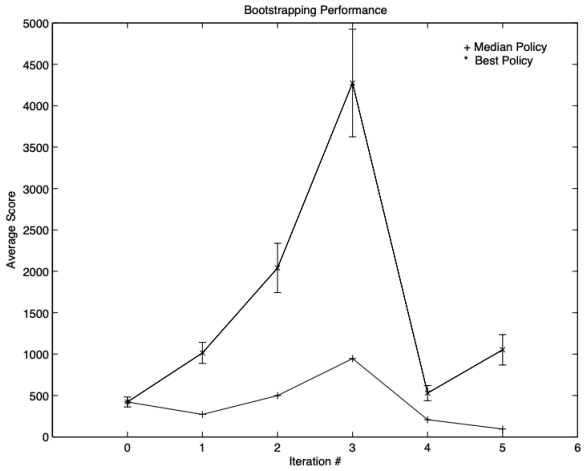

Figure 5: Results from applying the linear programming approach to Tetris.

2 Gain and Bias

Consider a unichain MDP $(\mathcal{S}, \mathcal{A}, P)$. Recall that the discounted value relates to gain and bias via a Laurent expansion:

$$V_{*,\gamma}(s) = \frac{1}{1-\gamma}\lambda_* + V_*(s) + O(1-\gamma).$$

Based on this, we can intuit a linear program that produces the optimal gain λ_*. To do this, we first rewrite (1), replacing $V(s)$ with $\frac{1}{1-\gamma}\lambda + V(s) + O(1-\gamma)$):

$$\begin{array}{ll} \min_{\lambda,V} & \sum_{s\in\mathcal{S}}(\frac{1}{1-\gamma}\lambda + V(s) + O(1-\gamma)) \\ \text{s.t.} & \frac{1}{1-\gamma}\lambda + V(s) + O(1-\gamma) \geq r(s,a) + \gamma\sum_{s'\in\mathcal{S}} P_{ass'}(\frac{1}{1-\gamma}\lambda + V(s) + O(1-\gamma)) \qquad \forall s \in \mathcal{S}, a \in \mathcal{A}. \end{array}$$

Letting γ approach 1 and rearranging terms, this linear program becomes

$$\begin{array}{ll} \min_{\lambda,V} & \lambda \\ \text{s.t.} & V(s) \geq r(s,a) + \sum_{s'\in\mathcal{S}} P_{ass'}V(s) - \lambda \qquad \forall s \in \mathcal{S}, a \in \mathcal{A}. \end{array}$$

We can write this concisely as

$$\begin{array}{ll} \min_{\lambda,V} & \lambda \\ \text{s.t.} & V \geq TV - \lambda. \end{array} \tag{7}$$

The optimal objective is λ_*, and each optimal solution takes the form $(\lambda_*, V_* + \eta\mathbf{1})$ for some $\eta \in \mathbb{R}$.

3 The Primal

Duality theory establishes that for each linear program, there is a dual with the same optimal objective value. For example, the so-called standard form linear program and its dual are given by:

$$\begin{array}{lll} \max_{x\in\mathbb{R}^N} & c^\top x & \\ \text{s.t.} & Ax = b & \qquad \min_{y\in\mathbb{R}^M} \quad b^\top y \\ & x \geq 0 & \qquad \text{s.t.} \quad A^\top y \geq c \end{array} \tag{8}$$

They duality theorem indicates that the maximal value delivered by the primal is equal to the minimal value delivered by the dual. By interpreting linear programs we have considered so far in this lecture as such duals, we can derive interesting primals. The primal linear programs involve maximizing expected return over state distributions.

3.1 Average Expected Return

For the case of expected average return, we make the following associations between (7) and the standard form dual in (8):

$$y \leftrightarrow \begin{bmatrix} \lambda \\ V \end{bmatrix} \qquad b \leftrightarrow \begin{bmatrix} 1 \\ \mathbf{0} \end{bmatrix} \qquad A_a^\top \leftrightarrow \begin{bmatrix} \mathbf{1} & (I - P_a) \end{bmatrix} \qquad c_a \leftrightarrow r(\cdot, a),$$

taking the matrix $A^\top$ and vector c to be stacking of matrices $(A_a^\top : a \in \mathcal{A})$ and $(c_a : a \in \mathcal{A})$, respectively. Then, the primal can be written as

$$\begin{array}{lll} \max_\mu & \sum_{s\in\mathcal{S}} \sum_{a\in\mathcal{A}} \mu(s,a) r(s,a) & \\ \text{s.t.} & \sum_{a'\in\mathcal{A}} \mu(s',a') = \sum_{s\in\mathcal{S}} \sum_{a\in\mathcal{A}} \mu(s,a) P_{ass'} & \forall s' \in \mathcal{S} \\ & \sum_{s\in\mathcal{S}} \sum_{a\in\mathcal{A}} \mu(s,a) = 1 & \\ & \mu(s,a) \geq 0 & \forall s \in \mathcal{S}, a \in \mathcal{A}. \end{array} \tag{9}$$

Under the unichain assumption, the constraints ensure that the feasible set comprises of all state-action distribution μ_π that can be attained by stationary policy. The objective is the expected average return under that policy. Hence, the maximal value is λ_*.

3.2 Discounted Expected Return

A similar exercise produces a primal for the discounted version of the linear program. In this case, we make the association

$$y \leftrightarrow V \qquad b \leftrightarrow \mathbf{1} \qquad A_a^\top \leftrightarrow I - \gamma P_a \qquad c_a \leftrightarrow r(\cdot, a).$$

The primal can be written as

$$\begin{array}{lll} \max_\mu & \sum_{s\in\mathcal{S}} \sum_{a\in\mathcal{A}} \mu(s,a) r(s,a) & \\ \text{s.t.} & \sum_{a'\in\mathcal{A}} \mu(s',a') = 1 + \gamma \sum_{s\in\mathcal{S}} \sum_{a\in\mathcal{A}} \mu(s,a) P_{ass'} & \forall s' \in \mathcal{S} \\ & \mu(s,a) \geq 0 & \forall s \in \mathcal{S}, a \in \mathcal{A}. \end{array} \tag{10}$$

To interpret the constraints, consider a change of variables: $\mu(s,a) = \tilde{\mu}(s,a)|\mathcal{S}|/(1-\gamma)$ and $\tilde{P}_{ass'} = (1-\gamma)/|\mathcal{S}| + \gamma P_{ass'}$. The equality constraints become

$$\sum_{a'\in\mathcal{A}} \tilde{\mu}(s',a') = \frac{1-\gamma}{|\mathcal{S}|} + \gamma \sum_{s\in\mathcal{S}} \sum_{a\in\mathcal{A}} \mu(s,a) P_{ass'} \qquad \forall s' \in \mathcal{S}.$$

Together with the nonnegativity constraint, these equality constraints ensure that the feasible set is the set of all state-action distributions of an MDP $(\mathcal{S}, \mathcal{A}, \tilde{P})$ that can be attained by stationary policies. We've shown in a previous homework that these are multiples discounted state-action frequencies. It follows that the objective of (10) is a multiple of the expected discounted return.

References

Vivek F Farias and Benjamin Van Roy. Tetris: A study of randomized constraint sampling. In *Probabilistic and randomized methods for design under uncertainty*, pages 189–201. Springer, 2006.