Q-LEARNING AS A MONOTONE SCHEME

Lingyi Yang
Mathematical Institute
University of Oxford
yangl@maths.ox.ac.uk

ABSTRACT

Stability issues with reinforcement learning methods persist. To better understand some of these stability and convergence issues involving deep reinforcement learning methods, we examine a simple linear quadratic example. We interpret the convergence criterion of exact Q-learning in the sense of a monotone scheme and discuss consequences of function approximation on monotonicity properties.

1 INTRODUCTION

Sutton & Barto (2018) coined the combination of bootstrapping, off-policy learning, and function approximations as "the deadly triad" due to stability issues. Let x denote the state, and u the action. A simple system for analysis is the linear quadratic (LQ) regulator where the dynamics is linear and the objective function is quadratic

$$x_{t+1} = Ax_t + Bu_t, \quad \min\left\{\sum_t x_t^\intercal Q x_t + u_t^\intercal P u_t\right\}.$$

Recht (2019) observed unstable behaviour on this system using a vanilla policy gradient method. Agarwal et al. (2021) showed policy gradients converge when the value function is increasing/monotone pointwise. We know that in general, the monotonicity of a numerical method can have a large impact on its convergence (Godunov & Bohachevsky, 1959; Crandall & Majda, 1980). We say an explicit numerical method with updates $u_i^{n+1} = S(\{u_j^n\}_{j\in\mathcal{I}})$, $S : \mathbb{R}^k \to \mathbb{R}$ is monotone if the operator S is monotone, i.e. if $u \geq v \Rightarrow Su \geq Sv$. Linear schemes are monotone only if all coefficients of u_j^n are non-negative (see the discussion on positive coefficient discretisation by Forsyth & Labahn (2007)). Barles & Souganidis (1991) proved that a stable, consistent, and monotone scheme converges (as the mesh size tends to zero) to the viscosity solution (Crandall & Lions, 1983; Crandall et al., 1992). We look at the impact of monotonicity in the context of continuous LQ problems, and the implications for Q-learning if we view it as a discretised numerical method.

2 1D DETERMINISTIC LINEAR QUADRATIC PROBLEM

Consider a continuous, infinite-horizon, discounted control problem with linear state dynamics

$$\frac{dX_t^{x,u}}{dt} = b(X_t^{x,u}, u_t), \quad b(X_t^{x,u}, u_t) = \alpha X_t^{x,u} + u_t, \quad X^{x,u}(0) = x,$$

where $\{X^{x,u}\}_t$ is the process starting at x and following the policy u. Let the cost function be $J(u_s; X_s) = \int_0^\infty e^{-\beta s} f(X_s^{x,u}, u_s) ds$, where $f(X_t^{x,u}, u_t) = (X_t^{x,u})^2 + u_t^2$, and define the value function $V(x) = \inf_u J(u; x)$. The Hamilton–Jacobi–Bellman (HJB) equation is

$$-\beta V(x) + \inf_u \left\{\partial_x V(x) \cdot b(x, u) + f(x, u)\right\} = 0. \tag{1}$$

To solve the HJB (1) numerically, we can rearrange the formula to get a fixed point problem

$$V^{n+1} = \frac{\beta + \gamma}{\gamma} V^n + \frac{1}{\gamma} \min_u \left\{\partial_x V^n \cdot (\alpha x + u) + x^2 + u^2\right\}. \tag{2}$$

If we choose a finite difference scheme for the derivative depending on the sign of $\alpha x_i + u$, then we can obtain an upwind method that ensures monotonicity. This subtle variation of finite difference schemes for different parts of the domain has an enormous impact on the stability in estimating value function and policies through value/policy iteration. A brief introduction to the continuous control set-up and the stability analysis of (2) can be found in Appendix A.

2.1 Q-LEARNING

Q-learning arises as a fixed point iteration to the Bellman optimality equation in discrete time

$$Q^{n+1}(x_t, u_t) = (1-\alpha)Q^n(x_t, u_t) + \alpha\left[f(x_t, u_t) + \gamma \min_{\hat{u}} Q^n(x_{t+1}, \hat{u})\right],$$

where $f(x, u)$ denotes the instantaneous reward function (Appendix B). We see the coefficients are non-negative for $0 \leq \alpha \leq 1$; within this range, the update step is monotone. This aligns with the usual range for the step size α in Q-learning. We explore larger values outside this range analogous to over-relaxation in optimisation (Saad, 2003). In our experiments, Q-learning is seen to be stable against the theoretical limits for a deterministic LQ problem when $0 \leq \alpha \leq 1$ (Figure 1). We converge to the correct value function, and the differences in policy are due to discretisation error. Since monotonicity is a sufficient condition for convergence, having α outside of this range does not necessitate the method breaking. In Figure 5 we see that we still converge to the theoretical values for $\alpha = 1.3$. Instability occurs when α is sufficiently large, $\alpha = 1.8$ will do (Figure 2).

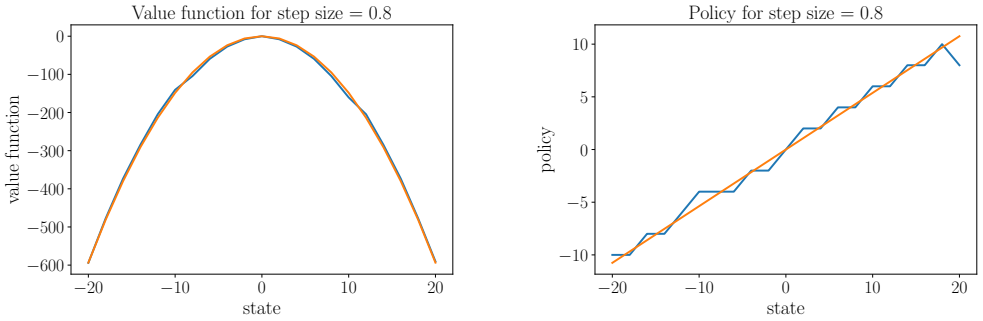

Figure 1: Q-learning: learnt value function and policy (blue) against theoretical (orange) for $\alpha = 0.8$

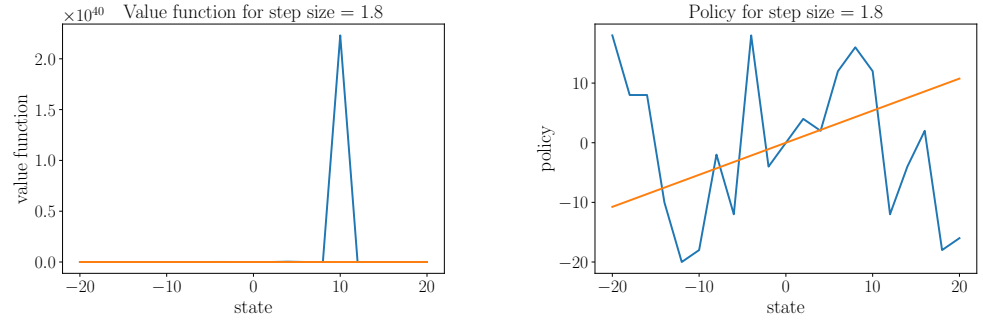

Figure 2: Q-learning: learnt value function and policy (blue) against theoretical (orange) for $\alpha = 1.8$

Ensuring monotonicity with a function approximator is non-trivial. In the LQ case, note $Q^*(x_t, u) = f(x_t, u) + V^*(x_{t+1})$ and f and V can be expressed as a quadratic function in x and u (Appendix A). Therefore a linear function approximator for $Q(x, u)$ with features of terms up to quadratic powers in x and u will be a suitable function class. To be precise, $\tilde{Q}(x, u, w) = X(x, u)^\intercal w$, where $X(x, u)$ are the features that we extract from our state-action pair, and w are the weights. Then (Appendix B)

$$\begin{aligned}\tilde{Q}^{n+1}(x, u, w_{n+1}) = \alpha_n f(x, u)X^\intercal(x, u)X(x, u) + (1 - \alpha_n X^\intercal(x, u)X(x, u))\tilde{Q}^n(x, u, w_n) \\ + \alpha_n X^\intercal(x, u)X(x, u) \max_{\tilde{u}} \tilde{Q}^n(x', \tilde{u}, w_n).\end{aligned}$$

To ensure monotonicity, the features $X(x, u)$ need to be bounded and step sizes are sufficiently small such that $\alpha_n X^\intercal(x, u)X(x, u) < 1$, i.e. $\alpha_n(x, u) < 1/(X^\intercal(x, u)X(x, u))$. This condition is dependent on the state and action, so a potential issue is that we may not sufficiently explore the state space (e.g. if for large values of x, the action u is also large then α needs to be very small). Even a simple, linear function approximator can disrupt monotonicity causing instability so violations in the nonlinear case (neural networks) may explain the stability issues we observe in practice.

ACKNOWLEDGEMENTS

The author would like to thank Prof Samuel N. Cohen and Dr Jaroslav Fowkes for their support and feedback. This work was supported by the EPSRC [EP/L015803/1].

URM STATEMENT

The authors acknowledge that at least one key author of this work meets the URM criteria of ICLR 2024 Tiny Papers Track.

REFERENCES

Alekh Agarwal, Sham M. Kakade, Jason D. Lee, and Gaurav Mahajan. On the Theory of Policy Gradient Methods: Optimality, Approximation, and Distribution Shift. *Journal of Machine Learning Research*, 22(98):1–76, 2021.

Guy Barles and Panagiotis E. Souganidis. Convergence of approximation schemes for fully nonlinear second order equations. *Asymptotic Analysis*, 4(3):271–283, 1991. doi: 10.3233/ASY-1991-4305. Publisher: IOS Press.

Michael G. Crandall and Pierre-Louis Lions. Viscosity Solutions of Hamilton–Jacobi Equations. *Transactions of the American Mathematical Society*, 277(1):1–42, 1983. ISSN 00029947. Publisher: American Mathematical Society.

Michael G. Crandall and Andrew Majda. Monotone difference approximations for scalar conservation laws. *Mathematics of Computation*, 34(149):1–21, 1980. ISSN 00255718, 10886842.

Michael G. Crandall, Hitoshi Ishii, and Pierre-Louis Lions. User's guide to viscosity solutions of second order partial differential equations. *Bulletin of the American Mathematical Society*, 27: 1–67, 1992.

Peter A. Forsyth and George Labahn. Numerical methods for controlled Hamilton–Jacobi–Bellman PDEs in finance. *Journal of Computational Finance*, 2007.

Sergei K. Godunov and I. Bohachevsky. Finite difference method for numerical computation of discontinuous solutions of the equations of fluid dynamics. *Matematičeskij sbornik*, 47(89)(3): 271–306, 1959.

Nikolay V. Krylov. *Controlled Diffusion Processes*. Stochastic Modelling and Applied Probability. Springer Berlin Heidelberg, 1980. ISBN 978-3-540-70914-5.

Benjamin Recht. A Tour of Reinforcement Learning: The View from Continuous Control. *Annual Review of Control, Robotics, and Autonomous Systems*, 2(1):253–279, May 2019. ISSN 2573-5144. doi: 10.1146/annurev-control-053018-023825. Publisher: Annual Reviews.

Herbert Robbins and Sutton Monro. A Stochastic Approximation Method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951. ISSN 00034851. Publisher: Institute of Mathematical Statistics.

Yousef Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, 2nd edition, 2003. doi: 10.1137/1.9780898718003.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2nd edition, 2018.

A 1D CONTINUOUS LQ PROBLEM

Consider the typical deterministic, infinite-horizon, discounted control problem with linear state dynamics

$$\frac{dX_t^{x,u}}{dt} = b(X_t^{x,u}, u_t), \quad b(X_t^{x,u}, u_t) = \alpha X_t^{x,u} + u_t, \quad X^{x,u}(0) = x, \tag{3}$$

where $\{X^{x,u}\}_t$ is the process starting at x and following the policy u thereafter. Let the cost function be given by

$$J(u_s; X_s) = \int_0^\infty e^{-\beta s} f(X_s^{x,u}, u_s) ds, \tag{4}$$

where

$$f(X_t^{x,u}, u_t) = (X_t^{x,u})^2 + u_t^2.$$

Define the value function

$$V(x) = \inf_u J(u; x). \tag{5}$$

Lemma 1. *The dynamic programming principle gives us*

$$V(x) = \inf_u \left\{ \int_0^h e^{-\beta s} \big((X_s^{x,u})^2 + u_s^2\big) ds + e^{-\beta h} V(X_h^{x,u}) \right\}.$$

Proof.

$$\begin{aligned}
V(x) &= \inf_u \int_0^\infty e^{-\beta s} f(X_s^{x,u}, u_s) ds \\
&= \inf_u \int_0^h e^{-\beta s} f(X_s^{x,u}, u_s) ds + \inf_u \int_h^\infty e^{-\beta s} f(X_s^{x,u}, u_s) ds \\
&= \inf_u \int_0^h e^{-\beta s} f(X_s^{x,u}, u_s) ds + \inf_{\tilde{u}} \int_0^\infty e^{-\beta(t+h)} f(X_t^{X_h^{x,u}, \tilde{u}}, \tilde{u}_t) dt \\
&= \inf_u \int_0^h e^{-\beta s} f(X_s^{x,u}, u_s) ds + e^{-\beta h} \inf_{\tilde{u}} \int_0^\infty e^{-\beta t} f(X_t^{X_h^{x,u}, \tilde{u}}, \tilde{u}_t) dt \\
&= \inf_u \int_0^h e^{-\beta s} f(X_s^{x,u}, u_s) ds + e^{-\beta h} V(X_h^{x,u}).
\end{aligned}$$

■

See (Krylov, 1980) for a rigorous formulation of this problem detailing the set of admissible controls and growth conditions assumed. From this form of the value function, we can derive the Hamilton–Jacobi–Bellman (HJB) equation.

Lemma 2. *The HJB equation of the system given by (3) and (4) is*

$$-\beta V(x) + \inf_u \big\{ \partial_x V(x) \cdot b(x, u) + f(x, u) \big\} = 0. \tag{6}$$

Proof. Let us first consider a constant control $\bar{u}$. By the definition of the value function, we must have

$$V(x) \le \int_0^h e^{-\beta s} f(X_s^{x,\bar{u}}, \bar{u}) ds + e^{-\beta h} V(X_h^{x,\bar{u}}).$$

Now by the chain rule we have

$$V(X_h^{x,\bar{u}}) = V(x) + \int_0^h b(X_t^{x,\bar{u}}, \bar{u}) \partial_x V(X_t^{x,\bar{u}}) dt,$$

therefore

$$V(x) \le \int_0^h e^{-\beta s} f(X_s^{x,\bar{u}}, \bar{u}) ds + e^{-\beta h} \left(V(x) + \int_0^h b(X_t^{x,\bar{u}}, \bar{u}) \partial_x V(X_t^{x,\bar{u}}) dt \right).$$

Upon rearranging, we have

$$\frac{1-e^{-\beta h}}{h}V(x) \leq \frac{1}{h}\int_0^h e^{-\beta s} f(X_s^{x,\bar{u}}, \bar{u})ds + \frac{e^{-\beta h}}{h}\int_0^h b(X_t^{x,\bar{u}}, \bar{u})\partial_x V(X_t^{x,\bar{u}})dt.$$

We consider the limit as h tends to 0. By L'Hôpital's rule

$$\lim_{h\to 0}\frac{1-e^{-\beta h}}{h} = \lim_{h\to 0}\frac{\beta e^{-\beta h}}{1} = \beta,$$

and by applying the Mean Value Theorem, we obtain

$$\beta V(x) \leq f(x, \bar{u}) + b(x, \bar{u})\partial_x V(x).$$

Therefore, for an arbitrary constant cost $\bar{u}$ we have

$$-\beta V(x) + f(x, \bar{u}) + b(x, \bar{u})\partial_x V(x) \geq 0,$$

thus

$$-\beta V(x) + \inf_u \left[f(x, u) + b(x, u)\partial_x V(x)\right] \geq 0.$$

If we apply the above analysis with the optimal control, we will find that equality holds

$$-\beta V(x) + \left[f(x, u^*) + b(x, u^*)\partial_x V(x)\right] = 0,$$

and therefore

$$-\beta V(x) + \inf_u \left[f(x, u) + b(x, u)\partial_x V(x)\right] = 0$$

as required. ■

We can find the optimal feedback control of (6).

Lemma 3. *The optimal control is given by*

$$u^* = -\Gamma x.$$

Proof. Let us propose the following ansatz for (6)

$$V(x) = \Gamma x^2 + 2\kappa x + \lambda.$$

We then find the derivative with respect to x, $dV/dx = 2\Gamma x + 2\kappa$ and substitute into the Hamiltonian to get

$$\partial_x V(x)\cdot b(x, u) + f(x, u) = Qx^2 + Ru^2 + (2\Gamma x + 2\kappa)(Ax + Bu).$$

By taking the partial derivative w.r.t u and setting to zero for the stationary point, we have that

$$2Ru + B(2\Gamma x + 2K) = 0$$

and the optimal control is given by

$$u^* = -\frac{B(\Gamma x + \kappa)}{R}.$$

Evaluate the Hamiltonian at the optimal control

$$\begin{aligned}
\partial_x V(x)\cdot b(x, u) + f(x, u)|_{u^*} &= Qx^2 + R\left(-\frac{B(\Gamma x+\kappa)}{R}\right)^2 + (2\Gamma x + 2\kappa)\left(Ax - \frac{B^2(\Gamma x+\kappa)}{R}\right) \\
&= Qx^2 + \cancel{\frac{B^2(\Gamma x+\kappa)^2}{R}} + 2A\Gamma x^2 + 2A\kappa x - \cancel{\frac{2B^2(\Gamma x+\kappa)^2}{R}} \\
&= \left(Q + 2A\Gamma - \frac{B^2\Gamma^2}{R}\right) + \left(2A\kappa - \frac{2B^2\Gamma\kappa}{R}\right)x - \frac{B^2\kappa^2}{R}.
\end{aligned}$$

Substitute this into the HJB

$$\beta(\Gamma x^2 + \kappa x + \lambda) - \left(\left(Q + 2A\Gamma - \frac{B^2\Gamma^2}{R}\right) + \left(2A\kappa - \frac{2B^2\Gamma\kappa}{R}\right)x - \frac{B^2\kappa^2}{R}\right) = 0$$

and set each coefficient to zero

$$0 = \frac{B^2\Gamma^2}{R} + \Gamma(\beta - 2A) - Q, \tag{7}$$

$$0 = 2\kappa(\beta + \frac{B^2\Gamma}{R} - A), \tag{8}$$

$$0 = \beta\lambda + \frac{B^2\kappa^2}{R}. \tag{9}$$

From (8), we see that either $\kappa = 0$ or $\Gamma = \frac{R(A-\beta)}{B^2}$. The latter case would not generally satisfy (7). Thus we must have $\kappa = 0$. Substituting this into (9), we also get that $\lambda = 0$. Thus the only non-zero coefficient of $V(x)$ is Γ, which satisfies the quadratic (7). We choose the root that ensures we have a stable solution (typically positive definite).

For our problem (3) we have that $A = \alpha$, $B = R = Q = 1$, hence Γ must satisfy

$$\Gamma^2 + (\beta - 2\alpha)\Gamma - 1 = 0, \tag{10}$$

which has two roots Γ^+ and Γ^-. We choose the root that would result in a positive eigenvalue. Our optimal control is then $u^* = -\Gamma x$. ■

If we want to solve the HJB (6) numerically, we can rearrange the formula to get a fixed point method

$$V^{n+1} = \frac{\beta + \gamma}{\gamma} V^n + \frac{1}{\gamma} \min_u \left\{ \partial_x V^n \cdot (\alpha x + u) + x^2 + u^2 \right\}. \tag{11}$$

If we naïvely tried to solve this directly by approximating the derivative with finite differences, for example, central difference, then we obtain the following numerical method

$$V_i^{n+1} = \frac{\beta + \gamma}{\gamma} V_i^n + \frac{1}{\gamma} \min_u \left\{ \frac{V_{i+1}^n - V_{i-1}^n}{2\Delta x} \cdot (\alpha x_i + u) + x_i^2 + u^2 \right\}, \tag{12}$$

where $V_i = V(x_i)$ and $x_i = x_{i-1} + \Delta x$, and we can take forward and backward differencing at the boundary.

This method is not monotone in general as the coefficients of V_{i+1}^n and V_{i-1}^n are the opposite signs to each other.

To obtain a monotone method we look at an upwind scheme. Let us approximate the derivative w.r.t. x with forward differences or backward differences depending on the sign of $\alpha x + u$

$$\partial_x V \approx \frac{V_{i+1} - V_i}{\Delta x} \quad \text{if} \quad \alpha x + u > 0$$

$$\partial_x V \approx \frac{V_i - V_{i-1}}{\Delta x} \quad \text{if} \quad \alpha x + u < 0.$$

For $\alpha x + u > 0$,

$$\begin{aligned} V_i^{n+1} &= \frac{\beta + \gamma}{\gamma} V_i^n + \frac{1}{\gamma} \min_u \left\{ \frac{V_{i+1}^n - V_i^n}{\Delta x} (\alpha x_i + u) + x_i^2 + u^2 \right\} \\ &= \left(\frac{\beta + \gamma}{\gamma} - \frac{\alpha x_i + u^*}{\gamma \Delta x} \right) V_i^n + \frac{\alpha x_i + u^*}{\gamma \Delta x} V_{i+1}^n + \frac{x_i^2 + (u^*)^2}{\gamma}, \end{aligned}$$

where u^* is the argmin of the Hamiltonian. Now the coefficient of V_{i+1}^n is positive, but the coefficient of V_i^n will only be positive if we have

$$\Delta x > \frac{\alpha x_i + u^*}{\beta + \gamma}. \tag{13}$$

Hence taking a large value for γ enables us to use finer meshes.

Similarly when $\alpha x + u < 0$,

$$V_i^{n+1} = \frac{\beta+\gamma}{\gamma} V_i^n + \frac{1}{\gamma} \min_u \left\{ \frac{V_i^n - V_{i-1}^n}{\Delta x}(\alpha x_i + u) + x_i^2 + u^2 \right\}$$
$$= \left(\frac{\beta+\gamma}{\gamma} + \frac{\alpha x_i + u^*}{\gamma \Delta x} \right) V_i^n - \frac{\alpha x_i + u^*}{\gamma \Delta x} V_{i-1}^n + \frac{x_i^2 + (u^*)^2}{\gamma}.$$

Since $\alpha x + u < 0$, the coefficient of V_{i-1}^n is positive, but the coefficient of V_i^n will only be positive if we have

$$\Delta x > -\frac{\alpha x_i + u^*}{\beta + \gamma}. \tag{14}$$

Note that by updating our value function as

$$V^{n+1} = \frac{\beta+\gamma}{\gamma} V^n + \frac{1}{\gamma} \min_u \{ \partial_x V^n \cdot (ax + u) + (x^2 + u^2) \},$$

this is precisely the value iteration updates. We only do one cycle of policy evaluation before choosing a new policy by taking u to be the argmin of the Hamiltonian (policy improvement). A monotone numerical scheme for the LQ problem with value iteration updates is described in Algorithm 1.

Algorithm 1: Value Iteration for LQ

1. Initialisation
Discretise the state and action spaces
Parameters: a small threshold $\theta > 0$ determining accuracy of estimation (convergence criterion), a maximum iteration count N
Initialize $V(x)$ as zeros
Initialize $\Delta > \theta$.
2. Iteration
repeat
  for *each* $x \in \mathcal{X}$ **do**
    $\hat{V}(x) \leftarrow V(x)$,
    $u^* \leftarrow$ argmin Hamiltonian at x
    $V(x) \leftarrow \frac{\beta+\gamma}{\gamma}\hat{V}(x) + \frac{1}{\gamma}H(x, u^*, \hat{V})$,
    where $H(x, u^*, \hat{V})$ is the Hamiltonian evaluated at x, with control u^* and using the suitable differencing for a monotone scheme.
  end
  $\Delta \leftarrow \max|\hat{V} - V|$.
until $\Delta < \theta$ *or* N;

For a policy iteration-like update, we need to have a fixed policy that we evaluate the value function on until convergence before we make a policy improvement step. The pseudocode is given in Algorithm 2

Let us be more precise on finding the control. To recap, for the value iteration, we are updating at each iteration with the rule

$$V^{n+1} = -\frac{\beta-\gamma}{\gamma} V^n + \frac{1}{\gamma} \min_u \left\{ \partial_x V^n \cdot (\alpha x + u) + (x^2 + u^2) \right\}.$$

In order to obtain a monotone scheme, we must apply forward differencing on $\partial_x V^n$ if $\alpha x + u > 0$ or backward differencing otherwise. However, for value iteration, we are also minimising over all u, which means that there are a few cases we can fall into. Let us consider finding the correct action for each discretised state x_i.

We have the regions $R_1 = \{u : \alpha x_i + u \geq 0\}$ and $R_1 = \{u : \alpha x_i + u < 0\}$. Let

$$H^n(u) = \begin{cases} h_1^n(u) & \text{if } u \in R_1 \\ h_2^n(u) & \text{if } u \in R_2 \end{cases}$$

where

$$h_1^n = \frac{V_{i+1}^n - V_i^n}{\Delta x}(\alpha x_i + u) + x_i^2 + u^2,$$

Algorithm 2: Policy Iteration for LQ

1. Initialisation
Discretise the state and action spaces
Parameters: two small thresholds $\theta_v > 0$ and $\theta_u > 0$ determining accuracy of estimation (convergence criterion), maximum iteration counts N_v, N_u
Initialize $u(x) \in \mathcal{U}$ arbitrarily for all $x \in X$ (let us say equal to 1).
Initialize $\Delta > \theta$.
2. Policy Evaluation
repeat
  for *each* $x \in \mathcal{X}$ **do**
    $\hat{V}(x) \leftarrow V(x)$,
    $V(x) \leftarrow \frac{\beta+\gamma}{\gamma}\hat{V}(x) + \frac{1}{\gamma}H(x, u, \hat{V})$,
    where $H(x, u, \hat{V})$ is the Hamiltonian evaluated at x, with the current control u and using the suitable differencing for a monotone scheme.
  end
  $\Delta \leftarrow \max|\hat{V} - V|$.
until $\Delta < \theta_v$;
3. Policy Improvement
for *each* $x \in \mathcal{X}$ **do**
  $\hat{u}(x) \leftarrow u(x)$
  $u(x) \leftarrow$ argmin of Hamiltonian at x
end
If $\max|\hat{u} - u| < \theta_u$, then stop and return $V \approx V^*$ and $u \approx u^*$; else go back to Step 2.

and

$$h_2^n = \frac{V_i^n - V_{i-1}^n}{\Delta x}(\alpha x_i + u) + x_i^2 + u^2.$$

Our problem is now to find

$$y = \min_u H^n(u).$$

The smoothness of $H(u)$ depends on the smoothness of the value function. In the case of the LQ problem, the value function is smooth, therefore $H(u)$ is smooth except on the boundary of R_1 and R_2, which in this case is a linear boundary, $u = -\alpha x_i$.

We can solve the above problem numerically, by finding

$$y_1^* = \min_u h_1(u), \quad u \in R_1$$

and

$$y_2^* = \min_u h_2(u), \quad u \in R_2$$

then taking the minimum over these 2 values

$$y = \min\{y_1, y_2\}$$

to get the desired control for value iteration.

For policy iteration, when we are doing policy evaluation, as the policy is fixed, we just need to check the sign of $\alpha x + u$. However, we need to also ensure that monotonicity is maintained in policy improvement. The policy improvement step can be described as

$$u^* = \arg\min_u H(u).$$

In this case we can again solve the two minimisation problem separately to find y_1^* and y_2^* and choose u based on which of these have the lower value.

In Figures 3 and 4, we see the result of applying the downwind iteration and upwind iteration respectively. We see clearly the instability arising in the downwind case, and how important it is to choose between forward and backward difference so that we ensure monotonicity.

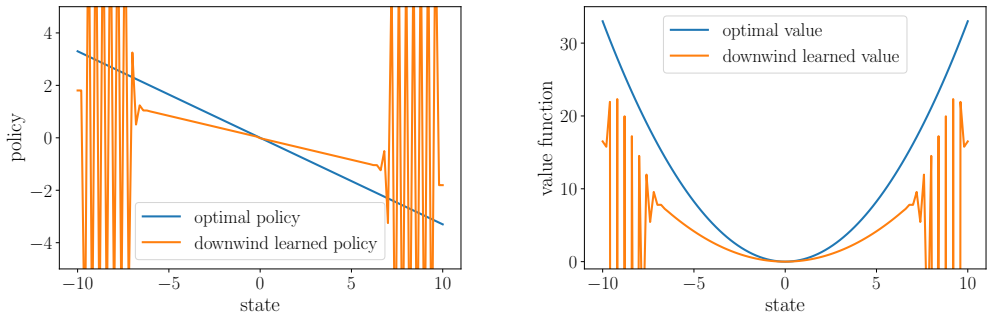

Figure 3: An intermediate policy and value function for a downwind method (the policy and value function have not converged yet). Instability forms and becomes amplified with further iterations.

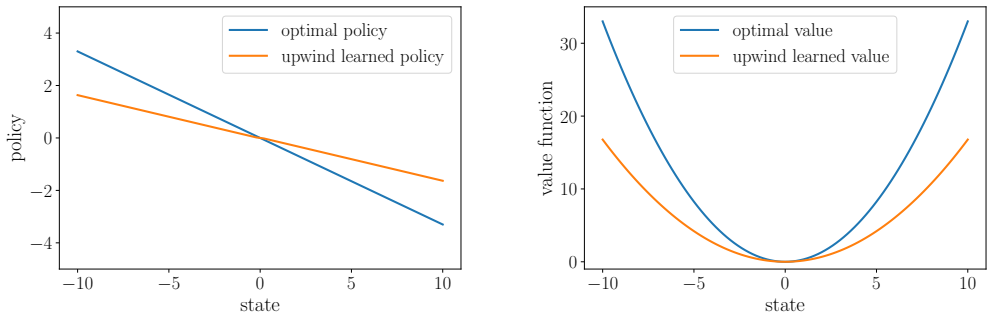

Figure 4: An intermediate policy and value function for an upwind method. Note that whilst the policy has not converged yet, there are no instabilities in this case.

B Q-LEARNING (DISCRETE SETTING)

Let the transition be $x_{t+1} = b(x_t, u_t)$ and let $f(x, u)$ denote the reward function. The state value function under policy π is defined as

$$V^{\pi}(x) = \sum_{t=0}^{\infty} \gamma^t f(x_t, \pi(x_t)), \quad x_0 = x.$$

The Bellman optimality equations are

$$V^*(x_t) = \max_{u_t} \left[f(x_t, u_t) + \gamma V^*(x_{t+1})|_{x_{t+1}=b(x_t,u_t)} \right],$$

$$\begin{aligned} Q^*(x_t, u_t) &= f(x_t, u_t) + \gamma V^*(x_{t+1})|_{x_{t+1}=b(x_t,u_t)} \\ &= f(x_t, u_t) + \gamma \min_{\hat{a}} Q(x_{t+1}, \hat{a})|_{x_{t+1}=b(x_t,u_t)}. \end{aligned}$$

Thus the first order condition is given by

$$\frac{\partial Q^*(s_{t+1}, \hat{a})}{\partial \hat{a}} = 0$$

or

$$\frac{\partial f}{\partial u_t} + \gamma \frac{\partial V^*}{\partial x_{t+1}} \frac{\partial x_{t+1}}{\partial u_t} = 0.$$

Monotonicity gives a sufficient condition for stability but it is not necessary. We see that Q-learning is also stable for $\alpha = 1.3$ case, as seen in Figure 5.

Note that, when we have a constant step size $\alpha > 1$, not only can we no longer guarantee monotonicity, but the square summability condition of the step size α (Robbins & Monro, 1951) would also be violated.

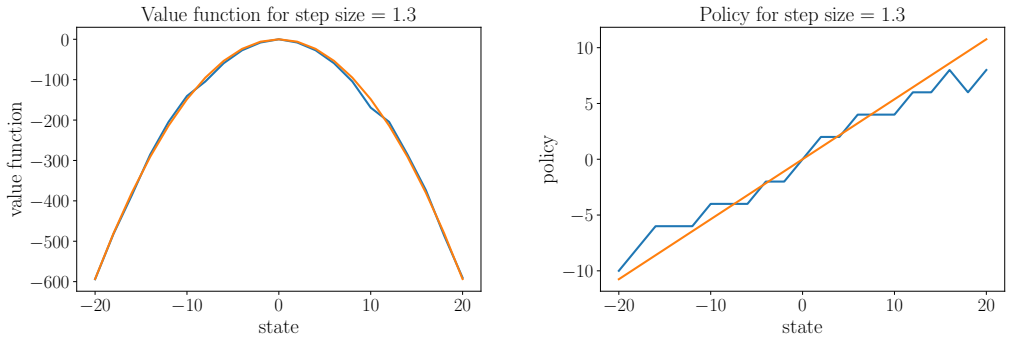

Figure 5: Learnt value function and policy (blue) against theoretical (orange) for $\alpha = 1.3$

For 'table-lookup' methods, as long as all states are updated infinitely often and step sizes satisfy square summability conditions (Robbins & Monro, 1951), then we have convergence for policy evaluation. This is not guaranteed when we have a general function approximator. The problem with having exact table representation is that they are slow to converge, and the number of states/state-action pairs suffer from the curse of dimensionality. Using a function approximator, we postulate that as long as the function approximation preserves monotonicity, then Q-learning should still converge.

When we are approximating $Q(x,u)$ by $\tilde{Q}(x,u,w)$, we want to minimise the expected error

$$\frac{1}{2}E\big[(Q(x,u)-\tilde{Q}(x,u,w))^2\big].$$

We use a semi-gradient descent method and take update steps in the form of

$$w_{n+1} = w_n + \alpha_n\big(Q(x,u)-\tilde{Q}(x,u,w_n)\big)\nabla_{w_n}\tilde{Q}(x,u,w_n).$$

For reinforcement learning/control problems, we do not have access to the true value function $Q(x,u)$ and therefore use an approximation in its place. An approximate method derived from Q-learning uses

$$Q(x,u) \approx f(x,u) + \gamma \max_{\bar{u}} \tilde{Q}(x',\bar{u},w).$$

Since

$$Q^*(x_t,u) = f(x_t,u) + V^*(x_{t+1})$$

and we know in the LQ case, both f and V can be expressed as a quadratic function in x and u (Appendix A), a linear function approximator for $Q(x,u)$ with features of terms up to quadratic powers in x and u will be a suitable function class for this approximation.

This linear approximator for $Q(x,u)$ is represented as

$$\tilde{Q}(x,u,w) = X(x,u)^\intercal w, \tag{15}$$

where $X(x,u)$ are the (e.g. quadratic) features that we extract from our state-action pair, and w are the weights. Then

$$\begin{aligned}
\tilde{Q}^{n+1}(x,u,w_{n+1}) &= X^\intercal(x,u)\big(w_n + \alpha_n\big(Q(x,u)-\tilde{Q}^n(x,u,w_n)\big)X(x,u)\big)\\
&\approx \tilde{Q}^n(x,u,w_n)\\
&\quad + \alpha_n X^\intercal(x,u)\big(f(x,u)+\gamma\max_{\tilde{u}}\tilde{Q}^n(x',\tilde{u},w_n)-\tilde{Q}^n(x,u,w_n)\big)X(x,u)\big)\\
&= \alpha_n f(x,u)X^\intercal(x,u)X(x,u) + (1-\alpha_n X^\intercal(x,u)X(x,u))\tilde{Q}^n(x,u,w_n)\\
&\quad + \alpha_n X^\intercal(x,u)X(x,u)\max_{\tilde{u}}\tilde{Q}^n(x',\tilde{u},w_n).
\end{aligned} \tag{16}$$

Monotonicity implies that the action-value function $\tilde{Q}^{n+1}(x,u,w_{n+1})$ must be non-decreasing with respect to the action-value function at the other points. Hence to ensure monotonicity we need to ensure that the features $X(x,u)$ are bounded and we take sufficiently small step size such that $\alpha_n X^\intercal(x,u)X(x,u) < 1$, i.e. $\alpha_n(x,u) < 1/(X^\intercal(x,u)X(x,u))$, the step size is dependent on the state and action. A potential problem is that we may not sufficiently explore the state space with a step size that is dependent on the state-action space (e.g. if for large values of x, the action u is also large then α needs to be very small).