Linear Feature Encoding for Reinforcement Learning
Supplemental Materials

Zhao Song, Ronald Parr†, Xuejun Liao, Lawrence Carin
Department of Electrical and Computer Engineering
† Department of Computer Science
Duke University, Durham, NC 27708, USA

1 Proof of Lemma 2

Proof: We start from the linear model solution and proceed as follows:

\[ w = (I - \gamma P^\pi_\Phi)^{-1} r_\Phi \]
\[ = (I - \gamma (\Phi^T \Phi)^{-1} \Phi^T P^\pi_\Phi)^{-1} (\Phi^T \Phi)^{-1} \Phi^T R \]
\[ = (\Phi^T \Phi - \gamma (\Phi^T P^\pi_\Phi)^{-1} \Phi^T R = w^*_\Phi. \]

where the penultimate substitutes the definition of \( r_\Phi \) and \( P^\pi_\Phi \) in (3a) and (3b) of the main text, respectively.

2 Proof of Theorem 3

Proof: The Bellman error in the context of linear value functions can be represented as

\[ BE(\hat{Q}^\pi(s, a)) = R(s, a) + \gamma \sum_{s', a'} P^\pi(s', a'|s, a) \Phi(s', a') w^*_\Phi - \Phi(s, a) w^*_\Phi \]  \hspace{1cm} (A1)

We proceed to represent (A1) in its corresponding matrix form as

\[ BE(\hat{Q}^\pi) = R + \gamma P^\pi \Phi w^*_\Phi - \Phi w^*_\Phi \]  \hspace{1cm} (A2)

Plugging (5) of the main text into (A2), we have

\[ BE(\hat{Q}^\pi) = R + \gamma P^\pi \Phi w^*_\Phi - \Phi w^*_\Phi \]
\[ = (\Delta_R + \Phi r_\Phi) + \gamma (\Delta^\pi_\Phi + \Phi P^\pi_\Phi) w^*_\Phi - \Phi w^*_\Phi \]
\[ = \Delta_R + \gamma \Delta^\pi_\Phi w^*_\Phi + \Phi r_\Phi - \Phi (I - \gamma P^\pi_\Phi) w^*_\Phi \]
\[ = \Delta_R + \gamma \Delta^\pi_\Phi w^*_\Phi + \Phi r_\Phi - \Phi (I - \gamma P^\pi_\Phi) w^*_\Phi \]
\[ = \Delta_R + \gamma \Delta^\pi_\Phi w^*_\Phi. \]

The penultimate step follows from Lemma 2, and the last follows equation (4b) of the main text.

3 Proof of Theorem 7

Proof: Equation (6) of the main text implies that there exist perfect linear predictors of the reward and the expected next state, given \( \Phi = AE_\pi \). Specifically, we pick \( P^\pi_\Phi = D^*_\pi E_\pi \) and \( r_\Phi = D^*_\pi \).

Next, we have

\[ \Delta^\pi_\Phi = P^\pi \Phi - \Phi P^\pi_\Phi = P^\pi \Phi - AE_\pi D^*_\pi E_\pi \]
\[ = P^\pi \Phi - P^\pi AE_\pi = P^\pi \Phi - P^\pi \Phi = 0 \]

\[ \Delta_R = R - \Phi r_\Phi = R - AE_\pi D_\pi^T = R - R = 0 \]

From Theorem 3, this implies zero Bellman error. \( \square \)

### 4 Proof of Theorem 8

**Proof:** Consider an MDP for which the \( Q \) and \( P^\pi \) are *not* linear in \( A \). This would be the typical case in which one would wish to use a neural network or other non-linear approximation method. \( P^\pi \) can be deterministic so that \( P^\pi_A \) is a matrix of raw encodings of actual states, not mixtures. Assume \( k = l \) and pick \( E = P^\pi \), i.e., pick a vacuous encoder. (For this example we will ignore the reward because predicting the reward does not change anything.) This implies a vacuous decoder \( D = I \). When combined, these predict \( P^\pi_A \). However, \( Q \) is not linear in \( A \) by assumption and therefore is not linear in \( \Phi = E(A) \) since elements of \( E(A) \) are also elements of \( A \). Therefore, a linear value function using features \( E(A) \) may have nonzero Bellman error. \( \square \)

### 5 Additional Results

After learning a policy \( \pi \), we can evaluate \( V_\pi \) exactly since there are just 203 states. Subsequently, we have

\[
\text{Actual return} = \sum_s V_\pi(s) b_0(s),
\]

where \( b_0 \) corresponds to a uniform distribution. Figure A1 shows the actual returns for different algorithms, where the “optimal” curve is obtained by solving the MDP.

Figure A1: Actual return as a function of the number of training episodes, in the Blackjack problem.