Primer on POMDPs and Infomax Control

Copyright ©Javier R. Movellan

Please cite as
Movellan J. R. (2009) Primer on POMDPs and Infomax Control. MPLab Tutorials, University of California San Diego
1 MDP Finite Horizon Problems

For more detailed specification of the notation standards see the Appendix. We use a Matlab-style convention to denote sequences. Under this convention \( x_{t:T} = (x_t, \cdots, x_T) \). We use capital letters for random variables and small letters for specific values taken from those variables. A process is a collection of variables indexed by \( t = 1, \cdots, T \), where \( T \) is called the horizon, or terminal time. The processes of interest are:

- **System Process**: \( X = \{X_1, \cdots, X_T\} \). Where the system, or state variables \( X_t \) take values in \( \{1, \cdots, n_x\} \).
- **Action Process**: \( U = \{U_1, \cdots, U_T\} \) where the actions \( U_t \) take values in \( \{1, \cdots, n_u\} \).
- **Control Process**: \( C = \{C_1, \cdots, C_T\} \). Where each \( C_t : H_t \to U_t \) maps states into actions.
- **Reward Process**: \( R = \{R_1, \cdots, R_T\} \). Where \( R_t : (X_t, U_t) \to \mathbb{R} \) maps state, action combinations into real valued numbers.
- **Return Process**: \( \bar{R} = \{\bar{R}_1, \cdots, \bar{R}_T\} \). Where \( \bar{R}_t \overset{\text{def}}{=} \sum_{\tau=t}^{T} \gamma^{\tau-t} R_\tau, \) and \( \gamma \in [0, 1] \) is called the discount factor.

**Value function for a given controller \( c \)**: It provides the expected return given that we visit state \( x_t \) at time \( t \) and use controller \( c \) to map states into actions.

\[
V^c_t(x_t) = E[\bar{R}_t | x_t, c] \tag{1}
\]

**Optimal value function**: It provides the expected return given that we visit state \( x_t \) at time \( t \), optimized over possible controllers

\[
V_t(x_t) = \max_c E[\bar{R}_t | x_t, c] \tag{2}
\]

**State/Action value function under controller \( c \)**:

\[
V^c_t(x_t, u_t) = E[\bar{R}_t | x_t, u_t, c] \tag{3}
\]

Note we are overloading the symbol for function \( V^c \) and identifying the function by the number of arguments.

**Optimal state/action value Function**:

\[
V_t(x_t, u_t) = \max_c E[\bar{R}_t | x_t, u_t, c] \tag{4}
\]

- **Generative Model** (See Figure 1).
  - \((X_t, C_t)\) generate \( U_t \).
  - \((X_t, U_t)\) generate \((R_t, X_{t+1})\).
- **System matrices**: \( a = \{a^1, \cdots, a^n\} \), where \( a^n \) is an \( n_x \times n_x \) matrix with

\[
a^u_{i,j} = p(X_{t+1} = j | X_t = i, U_t = u) \tag{5}
\]

- **Reward vectors**: \( r^u_i \) is an \( n_x \) dimensional vector such that

\[
r^u_{i,x} = E[R_t | X_t = i, U_t = u] \tag{6}
\]

**Remark 1.1.** Alternative Conventions: Some documents (e.g., Thrun Probabilistic Robotics) use the convention that \( C_t \) maps \( X_t \) into \( U_{t+1} \). This does not affect the main results presented here, except for the shift in notation.
Our goal is to find a controller $c$ that maximizes return. While this appears as a difficult optimization problem it has rich structure that allows solving it in a relatively efficient manner. Key to this solution is the Optimality Theorem, described below. To prove that theorem we will first need a lemma that tells us a condition under which the maximum of a sum of functions equal the sum of the maxima.

**Lemma 1.1 (Max Sum Lemma).** Let $w_i \geq 0$ for $i = 1, \cdots, n$ and let $\hat{x}$ maximize $f_i(x)$ for $i = 1, \cdots, n$, i.e.,

$$f_i(\hat{x}) = \max_x f_i(x)$$

for $i = 1, \cdots, n$. Then

$$\max_x \sum_i w_i f_i(x) = \sum_i w_i f_i(\hat{x})$$

**Proof.** Since $w_i \geq 0$ it follows that

$$\max_x \sum_i w_i f_i(x) \leq \sum_i \max_x w_i f_i(x) = \sum_i w_i f_i(\hat{x})$$

Moreover, by definition of the maximum operation, it follow that for any value $u$

$$\max_x \sum_i w_i f_i(x) \geq \sum_i w_i f_i(u)$$

Thus, choosing $u = \hat{x}$,

$$\max_x \sum_i w_i f_i(x) \geq \sum_i w_i f_i(\hat{x})$$

\[\blacksquare\]

**Theorem 1.1 (Optimality Theorem).** Let $\hat{c}_{t+1:T}$ be a controller that maximizes $E[R_{t+1} \mid x_{t+1}, c_{t+1:T}]$ for all $x_{t+1}$, i.e.,

$$E[\tilde{R}_{t+1} \mid x_{t+1}, \hat{c}_{t+1:T}] = \max_{c_{t+1:T}} E[\tilde{R}_{t+1} \mid x_{t+1}, c_{t+1:T}] = V_{t+1}(x_{t+1})$$

for $x_{t+1} = 1, \cdots, n_x$. Let $w_i \geq 0$ for $i = 1, \cdots, n_x$. Then

$$\max_{c_{t+1:T}} \sum_{i=1}^{n_x} w_i E[\tilde{R}_{t+1} \mid X_{t+1} = i, c_{t+1:T}] = \sum_{i=1}^{n_x} w_i E[\tilde{R}_{t+1} \mid X_{t+1} = i, \hat{c}_{t+1:T}]$$
Proof. It follows from the Max Sum Lemma using
\[
x(t) \overset{\text{def}}{=} c_{t+1:T}
\]
\[
f_i(x) \overset{\text{def}}{=} E[\tilde{R}_{t+1} \mid X_{t+1} = i, c_{t+1:T}]
\]
(14)
(15)

Corollary 1.1 (Bellman Equation for Optimal State/Value Function).
\[
V_i(x_t, u_t) = E[R_t \mid x_t, u_t] + \gamma E[V(x_{t+1}) \mid x_t, u_t]
\]
(16)

Proof.
\[
V_i(x_t, u_t) \overset{\text{def}}{=} \max_{c_{t+1:T}} E[\tilde{R}_t \mid x_t, u_t, c_{t+1:T}]
\]
\[
= E[R_t \mid x_t, u_t] + \max_{c_{t+1:T}} E[\tilde{R}_{t+1} \mid x_t, u_t, c_{t+1:T}]
\]
(17)
(18)
where we used the fact that \( \tilde{R}_t = R_t + \gamma \tilde{R}_{t+1} \) and the fact that \( R_t \) is conditionally independent of \( c_{t+1:T} \) given \( x_t, u_t \). Now note
\[
E[\tilde{R}_{t+1} \mid x_t, u_t, c_{t+1:T}] = \sum_{i=1}^{n_x} p(X_{t+1} = i \mid x_t, u_t) E[\tilde{R}_{t+1} \mid X_{t+1} = i, c_{t+1:T}]
\]
(19)
where we used the facts that
\[
p(X_{t+1} = i \mid x_t, u_t, c_{t+1:T}) = p(X_{t+1} = i \mid x_t, u_t)
\]
(20)
and the fact that
\[
E[\tilde{R}_{t+1} \mid X_{t+1} = i, u_t, c_{t+1:T}] = E[\tilde{R}_{t+1} \mid X_{t+1} = i, c_{t+1:T}]
\]
(21)
Thus, using the Optimality Theorem with \( w_i = p(X_{t+1} = i \mid x_t, u_t) \) it follows that
\[
\max_{c_{t+1:T}} E[\tilde{R}_{t+1} \mid x_t, u_t, c_{t+1:T}] = \sum_{i=1}^{n_x} p(X_{t+1} = i \mid x_t, u_t) E[\tilde{R}_{t+1} \mid X_{t+1} = i, c_{t+1:T}]
\]
\[
= \sum_{x_{t+1}=1}^{n_x} p(x_{t+1} \mid x_t, u_t) V_{t+1}(x_{t+1}) = E[V(X_{t+1}) \mid x_t, u_t]
\]
(22)

Corollary 1.2 (Bellman Equation for Optimal Value Function).
\[
V_t(x_t) = \max_{u_t} V_t(x_t, u_t)
\]
(23)

Proof.
\[
V_t(x_t) \overset{\text{def}}{=} \max_{c_t} \max_{c_{t+T}} E[\tilde{R}_t \mid x_t, c_t, c_{t+1:T}]
\]
\[
= \max_{c_t} \max_{u_t} E[R_t \mid x_t, u_t, \hat{c}_{t+1:T}] = \max_{u_t} V_t(x_t, u_t)
\]
(24)
(25)
where we used the fact that when \( x_t \) is fixed, the controller \( c_t \) determines \( u_t \), thus optimizing with respect to \( c_t(x_t) \) for a fixed \( x_t \) is the same as optimizing with respect to \( u_t \).
Corollary 1.3 (Bellman Equation for a Fixed Controller).

\[ V^c(x_t) = E[R_t + \gamma V^c_{t+1}(X_{t+1}) | x_t, u_t] \]  

(26)

Proof. First consider the case in which the admissible control laws are of the form

\[ U_t = C_t(X_t) \in C_t(X_t) \]  

(27)

where \( C_t(x_t) \) is a set of available actions when visiting state \( x_t \) at time \( t \). This can be seen as a special case of the optimal control problem that happens to have a large negative constant added to the reward function when using inadmissible actions. Thus

\[ V(x_t) = \max_{u_t \in C_t(x_t)} E[R_t + \gamma V_{t+1}(X_{t+1}) | x_t, u_t] \]  

(28)

To get the value of a fixed controller simply restrict the set \( C_t(x_t) = \{c_t(x_t)\} \) and apply (28)

\[ V^c(x_t) = E[R_t + \gamma V^c_{t+1}(X_{t+1}) | x_t, c_t] \]  

(29)

Corollary 1.4 (Optimal Controller). The optimal action at time \( t \) given that we are at state \( x_t \) is as follows

\[ \hat{c}_t(x_t) \arg\max_{c_t} Q_t(x_t, c_t) \]  

(30)

Proof.

\[ \hat{c}_t(x_t) \overset{\text{def}}{=} \arg\max_{u_t} \max_{c_{t+1:T}} E[R_t | x_t, c_{t:T}] = \arg\max_{c_t} Q_t(x_t, c_t) \]  

(31)

Remark 1.2 (Backpropagation Algorithm for MDPs). This suggests a useful method for finding optimal controllers: First we get the \( Q_T, V_T \) functions and the optimal controller \( \hat{c}_T \) for the terminal time \( T \):

\[ V_T(x_T, u_T) = E[R_T | x_T, u_T] \]  

(32)

\[ V_T(x_T) = \max_{u_T} Q_T(x_T, u_T) \]  

(33)

\[ \hat{c}_T(x_T) = \arg\max_{u_T} Q_T(x_T, u_T) \]  

(34)

We can then use the Bellman Equations to find the optimal value functions and optimal controllers

\[ V_{T-1}(x_{T-1}, u_{T-1}) = E[R_{T-1} | x_{T-1}, u_{T-1}] + \gamma \sum_{x_T} p(x_T | x_{T-1}, u_{T-1}) V_T(x_T) \]  

(35)

\[ V_{T-1}(x_{T-1}) = \max_{u_{T-1}} V_{T-1}(x_{T-1}, u_{T-1}) \]  

(36)

\[ \hat{c}_{T-1}(x_{T-1}) = \arg\max_{u_{T-1}} V_{T-1}(x_{T-1}, u_{T-1}) \]  

(37)

The process can be iterated backwards in time down to any desired time \( t \) to find the optimal control law \( \hat{c}_{t:T} \).
v_i^{u,T} = \max_u v_{u}^{i,T}, \quad \text{for } u = 1, \cdots, n_u.

v_{i,T} = \max_u v_{u}^{i,T}, \quad \text{for } u = 1, \cdots, n_u, i = 1, \cdots, n_x.

v_t^{u} = r_t^{u} + a_t^{u}v_{t+1}, \quad \text{for } t = T - 1, \cdots, 1, u = 1 \cdots, n_u.

v_{t,i} = \max_u q_{t,i}^{u}, \quad \text{for } t = T - 1, \cdots, 1, u = 1 \cdots, n_u, i = 1, \cdots, n_x.

\text{Figure 2: The backpropagation dynamic programing algorithm for computing the optimal value functions and optimal controller in MDPs. The } r_t^{u}, v_t^{u}, v_i^{u} \text{ terms are } n_x \text{ dimensional vectors.}

\text{Remark 1.3 (Assumptions). It is useful to clarify the assumptions made to prove the optimality principle and the Bellman Optimality Equations.}

- Assumption 1:
  \[ E[R_t | x_t, c_t; T] = E[R_t | x_t, c_t] \]  
(39)

- Assumption 2:
  \[ p(x_{t+1} | x_t, c_t, c_{t+1}; T) = p(x_{t+1} | x_t, c_t) \]  
(40)

- Assumption 3:
  \[ E[R_{t+1} | x_t, c_t, x_{t+1}, c_{t+1}; T] = E[R_{t+1} | x_{t+1}, c_t, c_{t+1}; T] \]  
(41)

- Assumption 4: Most importantly we assumed that the optimal controller \( \hat{c}_{t+1; T} \) did not impose any constraints on the set of policies \( c_t \) with respect to which we were performing the optimization. This would be violated, if there were an additional penalty or reward that depended directly on \( c_t; T \). For example, this assumption would be violated if we were to force the policies of interest to be stationary. This would amount to putting a large penalty for policies that do not satisfy \( c_1 = c_2 = \cdots c_{T-1} \).

\text{Remark 1.4 (Transition Dependent Rewards). In some problems the reward } R_t \text{ may be a function of } X_t, X_{t+1}, U_t. \text{ Note this does not brake any of the assumptions and so the Bellman equations hold. In cases like this it would probably be a good idea to change the notation so } R_t \text{ would be referred to as } R_{t+1} \text{ to reflect the dependency on variable that is generated at time } t + 1.\)

\text{2 Partially Observable Processes: Finite Horizon}

In addition to the processes defined in the fully observable case we have the following additional processes

- Sensor Process: \( Z = \{Z_1, \cdots, Z_T\} \). Where the sensor measurements \( Z_t \) take values in \( \{1, \cdots, n_z\} \)

- Observation History: \( H_t = (Z_{1:t}, U_{1:t-1}) \)

\footnote{We use the term argmax_x \( f(x) \) in an informal sense to signify any value that globally maximizes \( f \), i.e., if \( \hat{x} = \text{argmax} \ f(x) \) then \( f(\hat{x}) = \max f(x) \).}
• **Control Process:** $C = \{C_1, \ldots, C_t\}$. In this case the controller does not have access to the system variables. Instead at each time step $t$ the controller $C_t : H_t \rightarrow U_t$ maps the available information into actions.

• **Generative Model (See Figure 6).**
  
  - $(H_t, C_t)$ generate $U_t$.
  - $(X_t, U_t)$ generate $(R_t, X_{t+1}, Z_{t+1})$.
  - $(H_t, U_t, Z_{t+1})$ generate $H_{t+1}$.

• **Sensor matrices:** $b^u = \{b^1, \ldots, b^n_u\}$, where $b^u$ is an $n_x \times n_z$ matrix with $b^u_{i,j} = p(Z_t = j \mid X_t = i, U_t = u)$ (42)

Figure 3: Graphical Representation of the a time slice of the process. Arrows represent dependency relationships between variables. An arrow from variable $X$ to variable $Y$ indicates that $X$ is a parent of $Y$. The probability of a random variable is conditionally independent of all the other variables given the parent variables. Dotted figures indicate unobservable variables, continuous figures indicate observable variables.

2.1 **Equivalence with Fully Observable Case**

The goal in a finite horizon POMDP problem is to find control policies $c$ that map the observable history into actions in an optimal manner. Here optimality is defined with respect to a reward process $R_t(X_t, U_t)$

$$V^c_t(h_t) = E[R_t \mid h_t, c]$$ (43)

It turns this optimization problem is an MDP problem with respect to the information state. Note $H_t$ satisfies the necessary assumptions

• **Assumption 1:**
  $$E[R_t \mid h_t, c_t] = E[R_t \mid h_t, c]$$ (44)

• **Assumption 2:**
  $$p(o_{t+1} \mid h_t, c_t, o_{t+1:t}) = p(o_{t+1} \mid h_t, c_t)$$ (45)

• **Assumption 3:**
  $$E[R_{t+1} \mid h_t, c_t, h_{t+1}, c_{t+1:T}] = E[R_{t+1} \mid h_{t+1}, c_{t+1:T}]$$ (46)
2.2 Sufficient Statistics

While it is useful to know that POMPDs are MDPs with respect to the information state, the problem is that the number of information grows exponentially with the horizon $T$. For example if at each time we can get a binary sensory value $Z_t$ and a binary actuator value $U_t$ then by time $T$ there is total of $4^T$ possible states. Fortunately in some cases one may find a sufficient statistic $S_t$ of $H_t$ that does not lose the relevant information about $H_t$. To be usable, such a statistic would need to satisfy the conditions of the following theorem.

**Theorem 2.1 (Sufficient Statistics).** Let $S_t$ be a random variable satisfying the following conditions:

- **Assumption 1:** It is a function of $H_t$
  \[ S_t = f^1_t(H_t) \]  
  (47)

- **Assumption 2:** It is recursive:
  \[ S_{t+1} = f^2_t(S_t, U_t, Z_{t+1}) \]  
  (48)

- **Assumption 3:** The expected reward given $h_t, u_t$ is function of $s_t, u_t$
  \[ E[R_t | h_t, u_t] = f^3_t(s_t, u_t) \]  
  (49)

  where \( s_t \overset{\text{def}}{=} f^1_1(h_t) \)

- **Assumption 4:** The distribution of sensor measurements given $h_t, u_t$ is a function of $s_t, u_t$
  \[ p(z_{t+1} | h_t, u_t) = f^4_t(z_{t+1}, s_t, u_t) \]  
  (50)

  where \( s_t \overset{\text{def}}{=} f^1_t(h_t) \)

Then the optimal controller given $H_t$ is the same as the optimal controller given $S_t$.

**Proof.** First we show that at terminal time $T$ we can recover $V_T(h_t, u_t)$ using a function $\tilde{V}_T$ of $s_t, u_t$. For a fixed $h_T$ let $s_T \overset{\text{def}}{=} f^1_T(h_T)$ and note

\[ V_T(h_T, u_t) = E[R_T | h_T, u_T] = f^3_T(s_T, u_T) \]  
(51)

Let

\[ \tilde{V}_T(s_T, u_T) \overset{\text{def}}{=} f^3_T(s_T, u_T) = V_T(h_T, u_t) \]  
(52)

The same argument can be used to show that $V_T(h_t)$ and the optimal action $\hat{c}_T(h_t)$ are functions of $s_t$

\[ \tilde{V}_T(s_t) \overset{\text{def}}{=} \max_{u_T} \tilde{V}_T(s_t, u_T) = V_T(h_T) \]  
(53)

\[ \hat{c}_T(s_t) \overset{\text{def}}{=} \hat{c}_T(h_t) = \arg\max_{u_T} \tilde{V}_T(s_t, u_t) \]  
(54)

Now assume for time $t + 1$, the optimal value of an information state $h_t$ can be recovered from the statistic $s_{t+1} \overset{\text{def}}{=} f^1_{t+1}(h_{t+1})$, i.e., there is a function $\tilde{V}_{t+1}$ such that

\[ \tilde{V}_{t+1}(s_{t+1}) \overset{\text{def}}{=} V_{t+1}(h_{t+1}) \]  
(55)

We will now show that if this assumption holds, then the optimal value and the optimal actions at time $t$ for every information state $h_t$ can be computed from their
Thus the sufficient statistic of value functions and optimal actions for each possible information state \( h_t \) is a function of \( s_t \), its sufficient statistic, i.e., \( s_t = f_t^1(h_t) \). Using Bellman’s equation we get

\[
V_t(h_t, u_t) = E[R_t \mid h_t, u_t] + \gamma \sum_{h_{t+1}} p(h_{t+1} \mid h_t, u_t) V_{t+1}(h_{t+1})
\]  \hspace{1cm} (56)

where

\[
p(h_{t+1} \mid h_t, u_t) = \sum_{z_{t+1}} p(h_{t+1}, z_{t+1} \mid h_t, u_t, z_{t+1})
= \sum_{z_{t+1}} p(z_{t+1} \mid h_t, u_t) p(h_{t+1} \mid h_t, u_t)
\]  \hspace{1cm} (57)

Note \( h_t, u_t, z_{t+1} \) determine the information history at time \( t+1 \), i.e.

\[
p(h_{t+1} \mid h_t, u_t, z_{t+1}) = \delta(h_{t+1}, f_t^1(h_t, u_t, z_{t+1}))
\]  \hspace{1cm} (58)

\[
p(h_{t+1} \mid h_t, u_t) = \sum_{z_{t+1}} p(z_{t+1} \mid h_t, u_t) \delta(h_{t+1}, f_t^1(h_t, u_t, z_{t+1}))
\]  \hspace{1cm} (59)

Thus

\[
V_t(h_t, u_t) = E[R_t \mid h_t, u_t] + \gamma \sum_{z_{t+1}} p(z_{t+1} \mid h_t, u_t) V_{t+1}(f_t^1(h_t, u_t, z_{t+1}))
\]  \hspace{1cm} (60)

\[
= E[R_t \mid s_t, u_t] + \gamma \sum_{z_{t+1}} p(z_{t+1} \mid s_t, u_t) \tilde{V}_{t+1}(f_t^2(s_t, u_t, z_t))
\]  \hspace{1cm} (61)

which is a function of \( s_t, u_t \), i.e.

\[
\tilde{V}_t(s_t, u_t) \overset{\text{def}}{=} V_t(h_t, u_t) = E[R_t \mid s_t, u_t] + \gamma \sum_{z_{t+1}} p(z_{t+1} \mid s_t, u_t) \tilde{V}_{t+1}(f_t^2(s_t, u_t, z_t))
\]  \hspace{1cm} (62)

and

\[
\tilde{V}_t(s_t) \overset{\text{def}}{=} V_t(h_t) = \max_{u_t} \tilde{Q}_t(s_t, u_t)
\]  \hspace{1cm} (63)

\[
\tilde{c}_t(s_t) \overset{\text{def}}{=} \tilde{c}_t(h_t) = \max_{u_t} \tilde{Q}_t(s_t, u_t)
\]  \hspace{1cm} (64)

Thus, starting at \( T \) and moving backwards in time, we can compute all the necessary value functions and optimal actions for each possible information state \( h_t \) using just the sufficient statistic of \( h_t \).

**Theorem 2.2 (Controller Given the Posterior State Distribution).** The optimal controller given the information history \( h_t \) is equivalent to the optimal controller given the posterior distribution \( p(x_t \mid h_t) \).

**Proof.** We just need to show that the posterior distribution satisfies the assumption of the Sufficient Statistics Theorem.
• The posterior distribution is a function of the information history. This is obviously true since
\[ p(x_t \mid h_t) = f_1^t(h_t) \]  
(66)

• The posterior distribution is a recursive function. For a fixed \( h_t, u_t, z_{t+1} \) let \( h_{t+1} \overset{\text{def}}{=} f_2^t(h_t, u_t, z_{t+1}) \). Thus
\[ p(x_{t+1} \mid h_{t+1}) = p(x_{t+1} \mid h_t, u_t, z_{t+1}) = \frac{p(x_{t+1}, z_{t+1} \mid h_t, u_t)}{p(z_{t+1} \mid h_t, u_t)} \]  
(67)

where
\[ p(x_{t+1}, z_{t+1} \mid h_t, u_t) = \sum_{x_t} p(x_t, x_{t+1}, z_{t+1} \mid h_t, u_t) \]
\[ = \sum_{x_t} p(x_t \mid h_t)p(x_{t+1} \mid x_t, u_t)p(z_{t+1} \mid x_{t+1}, u_t) \]
\[ = \sum_{x_t} p(x_t \mid h_t)a_{x_t, x_{t+1}}b_{u_t}^{z_{t+1}} \]  
(68)

and
\[ p(z_{t+1} \mid h_t, u_t) = \sum_{x_{t+1}} p(x_{t+1}, z_{t+1} \mid h_t, u_t) \]
\[ = \sum_{x_t} p(x_t \mid h_t) \sum_{x_{t+1}} a_{x_t, x_{t+1}}b_{u_t}^{z_{t+1}} \]  
(69)

Thus
\[ p(x_{t+1} \mid h_{t+1}) = \frac{p(x_t \mid h_t) \sum_{x_{t+1}} a_{x_t, x_{t+1}}b_{u_t}^{z_{t+1}}}{\sum_{x_t} p(x_t \mid h_t) \sum_{x_{t+1}} a_{x_t, x_{t+1}}b_{u_t}^{z_{t+1}}} \]  
(70)

is a function of \( p(x_t \mid h_t), u_t, z_{t+1} \)

• The expected reward is a function of the posterior distribution. This is obviously true since
\[ E[R_t \mid h_t, u_t] = \sum p(x_t \mid h_t)R(x_t, u_t) \]  
(71)

• The distribution of sensor measurements is a function of the posterior distributions. This is evident in equation (69).

\[ \blacksquare \]

2.3 Backpropagation Algorithm for POMDPs

We will follow the convention that \( X_t, U_t \) cause the state \( X_{t+1} \). The state \( X_{t+1} \) and, possibly, the action \( U_t \) cause the observation \( Z_{t+1} \). Let \( n_x, n_u, n_z \) be the number of possible states, actions, and observations. Let \( H_t \) be the observable history up to time \( t \), i.e.,
\[ H_t = (Z_{1:t}, U_{1:t-1}) \]  
(72)

and \( Q_t \) represent the posterior probability of the states given the observations available up to that time, i.e.,
\[ q_{i,t} = p(X_t = i \mid h_t) \]  
(73)
Hereafter we refer to $Q_t$ as the information state at time $t$. Given an information state $q_t$ our goal is to find a controller that maps information states for all time steps $q$ into actions so as to optimize the accumulation of reward over a finite period of time. The value of an information state $q_t$ given a controller $c$, is defined as follows

$$V^c_t(q_t) = E[\bar{R}_t | q_t, c]$$ \hspace{1cm} (74)

where

$$\bar{R}_t \overset{\text{def}}{=} \sum_{\tau=t}^{T} \gamma^{\tau-t} R_\tau$$ \hspace{1cm} (75)

and $R_t$ is a function of $X_t, U_t$. The case in which $R_t$ is also a function of $X_{t-1}$ can be handled as described in Remark 1.4. Let

$$a_{ij}^u \overset{\text{def}}{=} p(X_{t+1} = j | X_t = i, U_t = u)$$ \hspace{1cm} (76)

$$b_{ij}^u \overset{\text{def}}{=} p(Z_{t+1} = j | X_{t+1} = i, U_t = u)$$ \hspace{1cm} (77)

Let $b^*_z$ be a diagonal matrix whose $i$th diagonal elements is $b_{i,z}^u$, i.e,

$$b^*_z \overset{\text{def}}{=} \text{diag}(b^u_{i,z})$$ \hspace{1cm} (79)

Let

$$r^u_{i,t} = R_t(i, u)$$ \hspace{1cm} (80)

Let

$$h_j(q_t, u, z) \overset{\text{def}}{=} p(X_{t+1} = j, Z_{t+1} = z | q_t, U_t = u)$$ \hspace{1cm} (81)

Thus

$$h_j(q_t, u, z) = \sum_{i=1}^{n_x} q_{i,t} a^u_{i,j} b^u_{j,z}$$ \hspace{1cm} (82)

or in vector form

$$h(q, u, z) = q^u a^u b^u_z$$ \hspace{1cm} (83)

Let

$$f_j(q_t, u, z) \overset{\text{def}}{=} p(X_{t+1} = j | q_t, U_t = u, Z_{t+1} = z)$$ \hspace{1cm} (84)

Thus

$$f_j(q_t, u, z) = \frac{h_j(q_t, u, z)}{p(Z_{t+1} = z | q_t, U_t = u)}$$ \hspace{1cm} (85)

where

$$p(Z_{t+1} = z | q_t, U_t = u) = \sum_{j=1}^{n_x} h_j(q_t, u, z)$$ \hspace{1cm} (86)

We now note that the transition probability between information states is a sum of delta functions

$$p(q_{t+1} | q_t, u_t) = \sum_{z_{t+1}} p(z_{t+1} | q_t, u_t) \delta(q_{t+1}, f(q_t, u_t, z_{t+1}))$$ \hspace{1cm} (87)
where $\delta$ is the Kronecker delta function
\[ \delta(u, v) = \begin{cases} 1 & \text{if } u = v \\ 0 & \text{else} \end{cases} \] (88)
and
\[ f(q, u, z) = \left( f_1(q, u, z), \cdots, f_n(q, u, z) \right) \] (89)
Let $V_t(q_t, u_t)$ the optimal value of the belief $q_t$ and action $u_t$ at time $t$, i.e.,
\[ V_t(q_t, u_t) = q_t^{r^u_t} + \gamma \sum_{q_{t+1}} p(q_{t+1} | q_t, u_t) V_{t+1}(q_{t+1}) \] (90)
where $V_{t+1}$ is the optimal value function at time $t+1$, and $\gamma$ is the discount factor.
Thus
\[ V_t(q_t, u_t) = q_t^{r^u_t} + \gamma \sum_{q_{t+1}} p(q_{t+1} | q_t, u_t) \delta(q_{t+1}, f(q_t, u_t, z)) V_{t+1}(q_{t+1}) \] (91)
\[ = q_t^{r^u_t} + \gamma \sum_{z_{t+1}} p(z_{t+1} | q_t, u_t) V_{t+1}(f(q_t, u_t, z)) \] (92)
\[ V_{t+1}(q_t, u_t) = q_t^{r^u_t} + \gamma \sum_{z_{t+1}} p(z_{t+1} | q_t, u_t) V_{t+1}(h(q_t, u_t, z)) \] (93)
We will now assume (and later prove) that there is a matrix $w_{t+1}$ such that
\[ V_{t+1}(q_t, u_t) = \max(q_t^{r^u_t}) \] (94)
Let $m_{t+1}$ be the number of columns of $w_{t+1}$. Note $q_t^{r^u_t}$ is an $m_{t+1}$ dimensional row vector. The max operator simply chooses an element of this vector that is not smaller than any other element of the vector.
In such case, for all $\alpha \in \mathbb{R}$
\[ V_{t+1}(\alpha q_t) = \alpha V_{t+1}(q_t) \] (95)
Thus
\[ V_t(q_t, u_t) = q_t^{r^u_t} + \gamma \sum_{z_{t+1}} V_{t+1}(p(z_{t+1} | q_t, u_t) f(q_t, u_t, z)) \] (96)
\[ = q_t^{r^u_t} + \gamma \sum_{z_{t+1}} V_{t+1}(h(q_t, u_t, z)) \] (97)
which under assumption (92) simplifies as follows
\[ V_t(q_t, u_t) = q_t^{r^u_t} + \gamma \sum_{z_{t+1}} \max(h(q_t, u_t, z)) w_{t+1} \] (98)
\[ = q_t^{r^u_t} + \gamma \sum_{z_{t+1}} \max(q_t^{a^n b^n z_{t+1}}) \] (99)
and using Lemma 4.2 (sum of max is max of cross sums)
\[ V_t(q_t, u_t) = q_t^{r^u_t} + \gamma \max \left( q_t^{\bigoplus_{z_{t+1}} a^n b^n z_{t+1}} \right) \] (100)
\[
\begin{align*}
\mathbf{w}_{T+1} &= (0, \cdots, 0)' \in \mathbb{R}^{n_z} \\
\text{For } t &= T, T-1, \cdots, 1 \\
k_t^{u,z} &\overset{\text{def}}{=} a^u \cdot \text{diag}(b_{t,z}), \text{ for } u = 1, \cdots n_u \text{ and } z = 1 \cdots n_z \\
n_t^{u} &\overset{\text{def}}{=} \gamma t^{u,z} \cdot w_{t+1}, \text{ for } u = 1, \cdots n_u \text{ and } z = 1 \cdots n_z \\
\mathbf{w}_t &\overset{\text{def}}{=} r_t^{u} \oplus w_{t-1}^{u} \oplus \cdots \oplus w_{t}^{u,n_z}, \text{ for } u = 1, \cdots n_u \\
\mathbf{w}_t &\overset{\text{def}}{=} (w_1^u, \cdots, w_n_u^u) \\
V_t(q, u) &= \max_u q' w_t^u, \text{ for } u = 1, \cdots n_u \\
V_t(q) &= \max_u V_t(q, u); \\
\end{align*}
\]

Note the cross sum operator \( \oplus \) and its properties are described in the Appendix. Thus
\[
V_t(q) = \max_u V_t(q, u) = \max_u (q' w_t^u) = \max(q' w_t) \tag{101}
\]
where \( w_t \) is a column-wise concatenation of the \( w_t^u \) matrices, i.e.,
\[
\mathbf{w}_t \overset{\text{def}}{=} \begin{pmatrix} w_1^u & \cdots & w_n_u^u \end{pmatrix} \tag{102}
\]
Thus we have shown that if assumption (92) is true for time \( t+1 \) then it must also be true for time \( t \). In addition equations (100) and (102) tell us how to construct the weight matrix \( w_t \) given the weight matrix \( w_{t+1} \). Now note that for \( t = T \)
\[
\begin{align*}
V_T(q, u) &= q' r_T^u \tag{103} \\
V_T(q) &= \max_u V_T(q, u) = \max(q' w_T) \tag{104} \\
\mathbf{w}_T &\overset{\text{def}}{=} (r_T^1, \cdots, r_T^{n_u}) \tag{105}
\end{align*}
\]
thus this shows, by induction, that assumption (92) is correct and provides a method to compute the value function starting at \( T \) and going all the way back to \( t \).

Note for time \( T \) \( \mathbf{w}_T = (r_T^1, \cdots, r_T^{n_u}) \) has \( n_u \) columns. For time \( t \) we have \( n_u \) \( w_t^u \) matrices. For each \( w_t^u \) we cross sum \( n_z \) matrices each with as many columns as the number of columns in \( w_t \) this gives a total of \( n_z n_t^{u+1} \) columns, where \( n_t^{u+1} \) represents the number of columns in \( w_{t+1} \). For each \( w_t^u \) we cross sum \( r_t^u \), which is a vector so it does not increase the number of columns. \( w_t \) concatenates the \( w_t^u \) matrices. Thus the \( w_t^u \) matrix has \( n_t = (n_z n_t^{u+1}) \cdot n_u \). Suppose \( n_u = 2, n_z = 2 \), then \( n_T = 2, n_{T-1} = (2^2) \cdot 2 = 8, n_{T-2} = (8^2) \cdot 2 = 128 \), etc.

### 3 Infomax Control

In some problems of interest the goal is to act in a manner that provides the most information about the state of the world. In such cases it is useful to use the
entropy of the posterior distribution as a component of the reward function. The Renyi entropy of order \( \alpha \geq 0 \) is defined as follows (I need to check the paper Blind Source Separation Using Renyi’s mutual information)

\[
H_\alpha(p) = \frac{1}{1-\alpha} \log(\sum_x p^\alpha(x))
\] (106)

It can be shown that in the limit as \( \alpha \to 1 \) \( H_\alpha \) converges to the Shannon Entropy

\[
\lim_{\alpha \to 1} H_\alpha(p) = \sum_x p(x) \log p(x)
\] (107)

Moreover

\[
H_\infty(p) = \lim_{\alpha \to \infty} H_\alpha(p) = -\log(\max_x p(x))
\] (108)

We can adapt the backpropagation algorithm to solve infomax problems by using

\[
\exp(-H_\infty(q_t)) = \max_x q_t(x)
\] (109)

as a component of the reward function. To this end we let the instantaneous reward associated with action \( u \) when in information state \( q \) is a follows

\[
\left( \sum_x q_t R_t(x, u) \right) + \lambda \max_x q_t(x)
\] (110)

for a parameter \( \lambda \geq 0 \) that controls the relative importance of the entropy term. In this case (91) takes teh following form

\[
V_t(q_t, u_t) = q_t' r_t + \max \lambda I + \gamma \sum_{z_{t+1}} p(z_{t+1} | q_t, u_t) V_{t+1}(f(q_t, u_t, z_{t+1}))
\] (111)

We assume (and later prove) that there is a matrix \( w_{t+1} \) such that

\[
V_{t+1}(q) = \max(q' w_{t+1})
\] (112)

Following the same steps as in (94) to (98) we get

\[
V_t(q, u) = q' r_t + \max q' \lambda I + \gamma \max(q' \bigoplus_{z=1}^{n_x} a^u b^u z w_{t+1}) = \max(q' w^u_t)
\] (113)

where \( I \) is an \( n_x \times n_x \) identity matrix and

\[
w^u_t = \left( \bigoplus_{z=1}^{n_x} a^u b^u z w_{t+1} \right) \oplus r_t \oplus \lambda I
\] (114)

Thus

\[
V_t(q) = \max_u V_t(q, u) = \max_u \max(q' w^u_t) = \max(q' w_t)
\] (115)

where \( w_t \) is the column-wise concatenation of the \( w^u_t \) matrices

\[
w_t = (w^u_t \cdots w^{n_x}_t)
\] (116)

All is left is to show that assumption (92) is true for time \( T \). Note

\[
V_T(q, u) = q' r_T + \lambda \max(q')
\] (117)

Thus

\[
V_T(q) = \max(q' r_t) + \max(q' \lambda I) = \max(q' w_T)
\] (118)
\[ w_{T+1} = (0, \cdots, 0)' \in \mathbb{R}^{n_x} \]

For \( t = T, T - 1, \cdots, 1 \)

\[ k_{t}^{u,z} \text{ def } = a^{u} \text{ diag}(b_{z}) \text{ for } u = 1, \cdots n_u \text{ and } z = 1 \cdots n_z \]

\[ w_{t}^{u,z} \text{ def } = \gamma b_{t}^{u,z} \cdot w_{t+1} \text{ for } u = 1, \cdots n_u \text{ and } z = 1 \cdots n_z \]

\[ w_{t}^{u} \text{ def } = \lambda I \oplus r_{t}^{u} \oplus w_{t}^{u,1} \oplus \cdots \oplus w_{t}^{u,n_z} \text{ for } u = 1, \cdots n_u \]

\[ w_{t} = \left( w_{t}^{1}, \cdots, w_{t}^{n_u} \right) \]

\[ V_{t}(q, u) = \max_{q'} w_{t}^{u}, \text{ for } u = 1, \cdots n_u \]

\[ V_{t}(q) = \max_{u} V_{t}(q, u); \]

Figure 5: The backpropagation dynamic programing algorithm for Infomax Control in POMDPs.

where

\[ w_{T} = r_{T} \oplus \lambda I \] (119)

\[ r_{T} = (r_{T}^{1}, \cdots, r_{T}^{n_u}) \] (120)

Note for time \( T \) we have \( n_u \) matrices \( w_{T}^{u} \). Each \( w_{T}^{u} \) has \( n_u \times n_x \) columns. Thus \( w_{T} \) has \( n_u \times n_x \) columns. For time \( t \) we have \( n_u \) matrices \( w_{t}^{u} \). For each \( w_{t}^{u} \), we cross sum \( n_z \) matrices each with as many columns as the number of columns in \( w_{t} \) this gives a total of \( n_{t+1} \) columns, where \( n_{t+1} \) represents the number of columns in \( w_{t+1} \). For each \( w_{t}^{u} \) we cross the \( I_{n_x} \) thus the number of columns for each is \( (n_{t+1})n_x \). For each \( w_{t}^{u} \) we cross sum \( r_{t}^{u} \), which is a vector so it does not increase the number of columns. \( w_{t} \) concatenates the \( w_{t}^{u} \) matrices.

Thus the \( w_{t}^{u} \) matrix has \( n_t = (n_{t+1})n_x n_u \). Suppose \( n_u = 2, n_z = 2 \), then \( n_T = 4, n_{T-1} = (4^2) \ast 4 = 64, n_{T-2} = (64^2) \ast 4 = 16384 \), etc.

3.1 A Counterintuitive Example

Consider the following case: There are two internal states, two actions and two observations. The state transition probability is the identity matrix

\[ a^{u} = I \text{ for } u = 1, 2 \] (121)

The first action provides no information about the state

\[ b_{1,z} = \left( \begin{array}{c} 0.5 \\ 0.5 \end{array} \right) \text{ for } z = 1, 2 \] (122)

where \( I \) is the identity matrix. The second action provides information about the state

\[ b_{2}^{1} = \left( \begin{array}{c} 0.9 \\ 0.1 \end{array} \right) \] (123)

\[ b_{2}^{2} = \left( \begin{array}{c} 0.1 \\ 0.9 \end{array} \right) \] (124)
If we choose action 2 and we get observation 1, then this indicates that the system is likely to be in state 1. If we choose action 2 and get observation 2, then it is likely to be in state 2. At time $t$ the controller has a belief state $q_t$ then it chooses one of the two actions. The total reward is the max value of the components of $q_t$ plus the max value of the components of $q_{t+1}$. This is related to the limit Renyi entropy with parameter $\alpha$
to $\infty$. It also corresponds to the the probability of correctly guessing the state at time $t$ plus the probability of correctly guessing the state at time $t + 1$. Using (126) we get

$$V_1(q_t, u) = \max q_t + \sum_{z_{t+1}} p(z_{t+1} | q_t, u_t) \max_{x_{t+1}} p(x_{t+1} | q_t, z_{t+1})$$

$$= \max q_t + \sum_{z_{t+1}} \max_{x_{t+1}} p(z_{t+1} | q_t, u_t) p(x_{t+1} | q_t, z_{t+1})$$

$$= \max q_t + \sum_{z_{t+1}} \max_{x_{t+1}} p(x_{t+1}, z_{t+1} | q_t, u_t) \tag{127}$$

Note for $U_t = 1$

$$p(Z_{t+1} = 1, X_{t+1} = 1 | q_t, U_t = 1) = q_{11}a_{11}b_{11}^1 + q_{21}a_{21}b_{12}^1 = q_{11}0.5$$

$$p(Z_{t+1} = 2, X_{t+1} = 1 | q_t, U_t = 1) = q_{11}a_{11}b_{12}^1 + q_{21}a_{21}b_{12}^1 = q_{11}0.5$$

$$p(Z_{t+1} = 1, X_{t+1} = 2 | q_t, U_t = 1) = q_{12}a_{12}b_{21}^1 + q_{22}a_{22}b_{22}^1 = q_{22}0.5$$

$$p(Z_{t+1} = 2, X_{t+1} = 2 | q_t, U_t = 1) = q_{12}a_{12}b_{22}^1 + q_{22}a_{22}b_{22}^1 = q_{22}0.5$$

Thus

$$V(q_t, 1) = (\max q_t) + 0.5 \max\{q_{1t}, q_{2t}\} + 0.5 \max\{q_{1t}, q_{2t}\} = 2 \max q_t \tag{132}$$

Thus if we choose the uninformative action, the probability of being correct at time $t$ equals the probability of being correct at time $t + 1$. The total reward is simply twice the probability of being correct given the prior belief $q_t$. For $U_t = 2$

$$p(Z_{t+1} = 1, X_{t+1} = 1 | q_t, U_t = 2) = q_{11}a_{11}b_{11}^2 + q_{21}a_{21}b_{12}^2 = q_{11}0.9$$

$$p(Z_{t+1} = 2, X_{t+1} = 1 | q_t, U_t = 2) = q_{11}a_{11}b_{12}^2 + q_{21}a_{21}b_{12}^2 = q_{11}0.1$$

$$p(Z_{t+1} = 1, X_{t+1} = 2 | q_t, U_t = 2) = q_{12}a_{12}b_{21}^2 + q_{22}a_{22}b_{21}^2 = q_{22}0.1$$

$$p(Z_{t+1} = 2, X_{t+1} = 2 | q_t, U_t = 2) = q_{12}a_{12}b_{22}^2 + q_{22}a_{22}b_{22}^2 = q_{22}0.9$$

Thus

$$V(q_t, 1) = \max q_t + \max\{q_{1t}0.9, q_{2t}0.1\} + 0.5 \max\{q_{1t}, q_{2t}\} \tag{137}$$

Consider the case in which we start with uninformative priors i.e., $q_t = (0.5, 0.5)^t$. In this case

$$V(q_t, 1) = 2 \times 0.5 = 1 \tag{138}$$

$$V(q_t, 2) = 0.5 + 0.5 \max\{0.9, 0.1\} + 0.5 \max\{0.1, 0.9\} \tag{139}$$

$$= 0.5 + 0.9 = 1.4 > V(q_t, 1) \tag{140}$$

Thus, not surprisingly, in this case the optimal strategy is to choose the most informative action. Consider now the case for which $q_t = (0.9, 0.1)$. In this case

$$V(q_t, 1) = 2 \times 0.9 = 1.8 \tag{141}$$

$$V(q_t, 2) = 0.9 + \max\{0.81, 0.01\} + \max\{0.09, 0.09\} \tag{142}$$

$$= 0.9 + 0.81 + 0.09 = 1.8 = V(q_t, 1) \tag{143}$$

Thus, surprisingly, in this case it does not matter which action we take, the informative action is as good as the uninformative action.
3.2 Point Based Approximations

A problem with the algorithm described above is that the weight matrix $w_t$ can grow very large. In particular if $w_{t+1}$ has $m_{t+1}$ rows, then $w_t$ will have $n_u \times m_{t+1}$. In practice many of these columns may be irrelevant since they may never be picked by the max operator. Note if a column of $w_t$ is never picked up by the max operator, it may be pruned out without any loss of information thus potentially mitigating the growth of $w_t$. In addition, a common approach is to focus on a finite set of information states $q$ rather than in all possible states. Such an approach goes by the name of point based POMDP approximations.

3.3 Policy Gradient Methods

In policy gradient methods the controller is parameterized and we attempt to find values of the parameter that optimize a utility function. Here we will focus on the finite horizon case but the approach can be easily generalized for infinite horizon problems. Let $\theta$ represent the parameters of a controller i.e. for each $\theta$ there is a probability distribution that maps belief states into actions. Let

$$\Phi(\theta) = E[r(Q, Z, U)]$$

where $Q = Q_{1:T}$ is the belief process is determined by the observation process $Z = Z_{1:T}$ and the action process $U = U_{1:T}$. Note we let the reward $r$ depend on the observable processes in a manner that may or may not be additive across time steps. Our goal is to find values of $\theta$ that maximize $\Phi$. Note

$$\Phi(\theta) = \int p(q, z, u) r(q, z, u) dq \, dz \, du$$

where $q = q_{1:T}, z = z_{1:T}, u = u_{1:T}$. We will employ stochastic gradient descent methods to maximize $\Phi$. Note

$$\nabla_\theta \Phi = \int r(q, z, u) \nabla_\theta p(q, z, u) dq \, dz \, du$$

$$= \int r(q, z, u) p(q, z, u) \nabla_\theta \log p(q, z, u) dq \, dz \, du \quad (146)$$

Thus we can approximate the gradient by getting $n$ samples from the observable processes $\{(q^{(i)}, z^{(i)}, u^{(i)}) : i = 1, \ldots, n\}$

$$\nabla_\theta \Phi \approx \frac{1}{n} \sum_{i=1}^{n} r(q^{(i)}, z^{(i)}, u^{(i)}) \nabla_\theta \log p(q^{(i)}, z^{(i)}, u^{(i)}) \quad (147)$$

Note for any sequence $(q, z, u)$ of belief, observations and actions

$$p(q, z, u | \theta) = p(q_0) p(z_1 | q_0) p(q_1 | q_0, z_1) p(u_1 | q_1, \theta)$$

$$p(z_2 | q_1, u_1) p(q_2 | q_1, u_1, z_2) p(u_2 | q_2, \theta) \cdots \quad (148)$$

Thus

$$\nabla_\theta \log p(q, z, u) = \sum_{t=1}^{T} \nabla_\theta \log p(u_t | q_t, \theta) \quad (150)$$

3.3.1 Soft-Max controller

A convenient way to parameterize the controller is to use a softmax function

$$p(U = i | q \theta) = \frac{e^{\theta' \phi(q)}}{\sum_k e^{\theta' \phi(q)}} \quad (151)$$
For policy gradient methods the controller is parameterized by $\theta$ and gradient methods are used to find values of $\theta$ that maximize the long term utility.

where $\phi(q)$ is a vector function of $q$ that represents key features of the belief state. Here we represent $\theta$ as a matrix. The element $\theta_{ij}$ of this matrix represents the support of feature $\phi_j(q)$ for response $i$. The term $\theta_{i \cdot}$ is the $i$ column of the matrix $\theta$. Let $q$ be fixed and define

\[
p_k = p(U = k \mid q \theta)
\]
\[
\phi = \phi(q)
\]

Note

\[
\frac{\partial p_k}{\partial \theta_{ij}} = p_k (\delta_{jk} - p_k) \phi_j
\]

Thus

\[
\frac{\partial \log p_k}{\partial \theta_{ij}} = (\delta_{jk} - p_k) \phi_j
\]

4 Appendix

4.1 Terminology

Optimal controllers are presented in terms of maximization of a reward function. Equivalently they could be presented as minimization of costs, by simply setting the cost function equal the reward with opposite sign. Below is a list of useful words and their equivalents

- Cost = - Value = - Reward = - Utility = - Payoff
- The goal is to minimize Costs, or equivalent to maximize Value, Reward, Utility.
- We will use the terms Return and Performance to signify Cost or Value.
- Step = Stage
- One Step Cost = Running Cost
- Terminal cost = Bequest cost
- Policy = control law = controller = control
• Optimal n-step to go cost = optimal 1 step cost + optimal (n-1) step to go cost
• n-step to go cost given policy = 1 step cost given policy + (n-1) step to go cost given policy

4.2 Cross Sums

Definition 4.1. Let \( \mathbf{a} \) be an \( r \) row, \( c \) column matrix and \( \mathbf{b} \) an \( r \) row, \( d \) column matrix. Then \( \mathbf{a} \oplus \mathbf{b} \) is an \( r \) row by \( c \times d \) column matrix, defined as follows

\[
\mathbf{a} \oplus \mathbf{b} = \begin{pmatrix}
(a_1 + b_1), & \cdots, & (a_1 + b_d), \\
\vdots & \ddots & \vdots \\
(a_c + b_1), & \cdots, & (a_c + b_d)
\end{pmatrix}
\]  

where \( a_i, b_j \) are the \( i \)th column of \( \mathbf{a} \) and \( j \)th column of \( \mathbf{b} \).

Definition 4.2. Let \( \mathbf{a}_1, \cdots, \mathbf{a}_n \) be a set of matrices with \( r \) rows and \( c_i \) columns.

\[
\bigoplus_{i=1}^{n} \mathbf{a}_i \stackrel{\text{def}}{=} \mathbf{a}_1 \oplus \mathbf{a}_2 \oplus \cdots \oplus \mathbf{a}_n
\]  

Remark 4.1. Note \( \bigoplus_{i=1}^{n} \mathbf{a}_i \) is a matrix with \( r \) rows and \( \prod_{i=1}^{n} c_i \) columns.

Lemma 4.1 (Sum of Max is Max of Cross Sum). Let \( x, y \) be row vectors. Let \( \text{max}(x) \) be an element of \( x \) that is no smaller than any of the other elements of \( x \). Then

\[
\text{max}(x) + \text{max}(y) = \text{max}(x \oplus y)
\]  

Lemma 4.2. Let \( x \) be an \( r \)-dimensional row vector. Let \( \mathbf{a}_1 \cdots \mathbf{a}_n \) be matrices with \( r \) rows. Then

\[
\sum_{i=1}^{n} \text{max}(x \mathbf{a}_i) = \text{max} \left( x \bigoplus_{i=1}^{n} \mathbf{a}_i \right)
\]