Dynamic Programming Lecture #17

Outline:

- Stochastic fixed point
- Q-learning

“Stochastic” Fixed Point

- Objective: Find fixed point
$$x = E_\theta \{G(x, \theta)\}$$
Probabilities of θ can depend on x
- Example: Unknown mean
$$G(x, \theta) = \theta \Rightarrow x = E\{\theta\}$$
- Example: Stochastic gradient
$$G(x, \theta) = x - \nabla f(x) + \theta$$
$$x = E_\theta \{x - \nabla f(x) + \theta\}$$
If θ is zero-mean,
$$0 = -\nabla f(x)$$
- Problem: Can only measure “noisy” samples $G(x, \theta)$
- Noisy fixed-point iteration attempt:
$$x^+ = G(x, \theta)$$
- Averaging attempt:
  - Take several samples
  $$\bar{G}(x) = \frac{1}{K} \sum_{k=1}^{K} G(x, \theta_k)$$
  - Apply deterministic algorithm on $\bar{G}(x)$, e.g.,
  $$x^+ = \bar{G}(x)$$

Robbins-Monro & Stochastic Approximation

- Iterative algorithm:

$$x^+ = (1-\gamma)x + \gamma G(x,\theta) = x + \gamma(G(x,\theta) - x)$$

- Main issue: How to choose iteration-dependent step size, γ, to neutralize effect of θ?
- Example: Unknown mean

$$x_{t+1} = x_t + \gamma_t(\theta_t - x_t)$$

For $\gamma_t = \frac{1}{t+1}$,

$$x_{t+1} = x_t + \frac{1}{t+1}(\theta_t - x_t)$$

- This is a recursive form of a running average:

$$\begin{aligned} x_{t+1} &= \frac{\theta_0 + \ldots + \theta_t}{t+1} \\ &= \frac{\theta_0 + \ldots + \theta_{t-1}}{t}\frac{t}{t+1} + \frac{\theta_t}{t+1} \\ &= x_t\left(1 - \frac{1}{t+1}\right) + \frac{1}{t+1}\theta_t \\ &= x_t + \frac{1}{t+1}(\theta_t - x_t) \end{aligned}$$

- Temporal difference learning is precisely an iterative algorithm for stochastic approximation

Q-Factor DP

- Define Q-factor:

$$Q(i,u) = g(i,u) + \alpha \sum_{s} p_{ij}(u) J^*(j)$$

NOTE: Q is function of state AND control.

- Bellman equation:

$$J^*(i) = \min_{u} Q(i,u)$$

- Implication: For any i and u

$$g(i,u) + \alpha \sum_{j} p_{ij}(u) J^*(j) = g(i,u) + \alpha \sum_{j} p_{ij}(u) \min_{v} Q(j,v)$$

$$\Rightarrow$$

$$Q(i,u) = g(i,u) + \alpha \cdot E_j \left\{ \min_{v} Q(j,v) \right\}$$

"Q-factor Bellman equation"

- What's the difference?

$$E_w \min_{u}(\cdot) \quad \text{vs} \quad \min_{u} E_w(\cdot)$$

$$Q\text{-factor Bellman} \quad \text{vs} \quad J \text{ Bellman}$$

- Q learning:

$$Q^+(i,u) = Q(i,u) + \gamma \left(g(i,u) + \alpha \min_v Q(j,v) - Q(i,u) \right)$$

Q-Factor Contraction

- Q-factor Bellman equation:

$$Q(i,u) = g(i,u) + \alpha \sum_s p_{ij}(u) J^*(j)$$
$$\Rightarrow$$
$$Q(i,u) = g(i,u) + \alpha E_j \left\{ \min_v Q(j,v) \right\}$$

  or

$$Q = \bar{G}Q$$

- FACT: $\bar{G}$ is a contraction in the $\max$-norm.
- Proof: Suppose

$$Q_B(i,u) - c \leq Q_A(i,u) \leq Q_B(i,u) + c$$

  Then

$$\begin{aligned} (\bar{G}Q_A)(i,u) &= g(i,u) + \alpha \sum_{j=1}^{n} p_{ij}(u) \min_v Q_A(j,v) \\ &\leq g(i,u) + \alpha \sum_{j=1}^{n} p_{ij}(u) (\min_v Q_B(j,v) + c) \\ &= (\bar{G}Q_B)(i,u) + \alpha c \end{aligned}$$

  Likewise

$$(\bar{G}Q_B)(i,u) - \alpha c \leq (\bar{G}Q_A)(i,u)$$

Q Learning

- Q Bellman equation looks like stochastic fixed point

$$x = E_\theta \{G(x, \theta)\}$$

where

  - $x \sim Q(i, u)$
  - $\theta \sim j$ (i.e., next state)

- Apply stochastic iterations:

$$\begin{aligned} Q^+(i, u) &= Q(i, u) + \gamma \left(g(i, u) + \alpha \min_v Q(j, v) - Q(i, u) \right) \\ &= Q(i, u) + \gamma \left((\bar{G}Q)(i, u) - Q(i, u) + w \right) \end{aligned}$$

where

$$w = \min_v Q(j, v) - \sum_{j=1}^{n} p_{ij}(u) \min_v Q(j, v)$$

Note

$$E\{w|Q\} = 0$$

- *Almost* looks like stochastic approximation, but only one (i, u) pair is updated per iteration.

- THEOREM: *Asynchronous* Q-learning results in bounded iterations that converge to the unique equilibrium, Q^*.

Q Learning Issues

- Convergence requires infinite visits to every (i, u) pair
- No policy is specified!
  - Define "softmax"
    $$\sigma_i(v; T) = \frac{e^{v_i/T}}{e^{v_1/T} + ... + e^{v_m/T}}$$
    e.g., for $m = 2$,
    $$\sigma(v) = \begin{pmatrix} e^{v_1/T}/(e^{v_1/T} + e^{v_2/T}) \\ e^{v_2/T}/(e^{v_1/T} + e^{v_2/T}) \end{pmatrix}$$
  - Note that $\sum_i \sigma_i(v) = 1$.
  - Choosing a component according to a distribution of $\sigma_i(v)$ looks like choosing maximum of v with high probability.
  - Parameter T represents "temperature". Recovers $\max$ as $T \to \infty$
  - Suitable policy to accompany Q-learning:
    $$\mu(i; Q) = \texttt{rand}[\sigma(Q(i, \cdot); T)]$$
    Combines "exploration" with "exploitation".
- What about curse of dimensionality?
  - Impose a structured form of Q:
    $$Q(i, u) = \Phi(i, u; r)$$
    where r is a vector of parameters (e.g., basis coefficients, neural net weights, etc.)
  - New Q learning: Update coefficients as done in temporal difference learning
  - No convergence results.