Lecture 7 and 8: More policy gradients and LQR

Reinforcement Learning and Decision Making MLSALT7, Lent 2016

Matthew W. Hoffman, Zoubin Ghahramani, Carl Edward Rasmussen

Department of Engineering
University of Cambridge

http://mlg.eng.cam.ac.uk/teaching/mlsalt7/

Variance reduction for policy gradients

The policy gradient

Recall initial definition of the policy gradient from last lecture:

$$\nabla_\theta J(\theta) = \int p_\theta(\tau) \Big[\sum_{t=0}^{T} \gamma^t r(x_t) \Big] \Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t | x_t) \Big] \, d\tau$$

where $\tau = (x_0, \ldots, x_t, a_0, \ldots, a_T)$ is a full trajectory. Given K sample trajectories we can approximate this gradient as follows:

$$\approx \frac{1}{K} \sum_{i=1}^{K} \Big[\sum_{t=0}^{T} \gamma^t r(x_t^i) \Big] \Big[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t^i | x_t^i) \Big]$$

Note the following, for each trajectory:

- we sample from $p_0(x_0)$ and continue…
- using the same θ
- the reason we are using $K \geqslant 1$ is to get a better estimate; the Monte Carlo estimate can be quite noisy!

$$\begin{aligned}\nabla_\theta J(\theta) &= \int p_\theta(\tau)\Big[\sum_{t=0}^{T}\gamma^t\, r(x_t)\Big]\Big[\sum_{t=0}^{T}\nabla\log\pi_\theta(a_t|x_t)\Big]\, d\tau \\ &= \int p_\theta(\tau)\sum_{t=0}^{T}\sum_{n=0}^{T}\gamma^n\, r(x_n)\,\nabla\log\pi_\theta(a_t|x_t)\, d\tau \\ &= \sum_{t=0}^{T}\sum_{n=0}^{T}\underbrace{\int p_\theta(x_n,a_t,x_t)\,\gamma^n\, r(x_n)\,\nabla\log\pi_\theta(a_t|x_t)\, d(x_n,a_t,x_t)}_{A}\end{aligned}$$

Let's consider $n < t$:

$$A = \int p_\theta(x_n,x_t)\,\gamma^n\, r(x_n)\Big[\underbrace{\int \pi_\theta(a_t|x_t)\,\nabla\log\pi_\theta(a_t|x_t)\, da_t}_{\text{expectation of a score is equal to 0}}\Big]\, dx_n\, dx_t = 0$$

Plugging this back in (leaving these terms in would only add variance!):

$$\nabla_\theta J(\theta) = \int p_\theta(\tau)\Big[\sum_{t=0}^{T}\nabla\log\pi_\theta(a_t|x_t)\Big[\sum_{n=t}^{T}\gamma^t\, r(x_t)\Big]\Big]\, d\tau$$

Expectation of a score

The simplification of the expectation of a score (gradient of a log-likelihood) again makes use of $\nabla \log x = \frac{1}{x}\nabla x$

$$\begin{aligned}\int \nabla_\theta \log p(x|\theta)\, p(x|\theta)\, dx &= \int \nabla_\theta p(x|\theta)\, dx \\ &= \nabla_\theta \int p(x|\theta)\, dx \\ &= \nabla_\theta 1 = 0\end{aligned}$$

Variance reduction in general

Consider the general problem of computing an expectation:

$$\mathbb{E}_{p(x)}\big[f(x)\big] \approx \underbrace{\tfrac{1}{N}\sum_{i=1}^{N} f(x^i)}_{F} \quad \text{where } x^i \sim p(x)$$

but F is now a random variable so we can talk about its variance

Problem: F may have high variance; in the setting of gradient descent this additional variance can "bump" us off the descent path

Solution: replace F with a new quantity F' with the same expectation, but lower variance

$$\begin{aligned}\mathbb{E}[F'] &= \mathbb{E}[F] = \mathbb{E}[f(x)],\\ \operatorname{var}[F'] &\leqslant \operatorname{var}[F].\end{aligned}$$

Control variates

Consider an additional function $\phi(x)$ whose expectation $\mu_\phi = \mathbb{E}[\phi(x)]$ we know. We can introduce this function and write

$$\mathbb{E}\big[f(x)\big] = \underbrace{\mathbb{E}\big[f(x) - \phi(x)\big]}_{\text{use Monte Carlo here}} + \underbrace{\mu_\phi}_{\text{we know this}}$$

Nothing ground-breaking, but what about the variance?

$$\mathrm{var}\big[f(x) - \phi(x)\big] = \mathrm{var}\big[f(x)\big] - 2\mathrm{cov}\big[f(x), \phi(x)\big] + \mathrm{var}\big[\phi(x)\big]$$

i.e. we can get a reduction in variance if f and ϕ are strongly correlated

ϕ is our control variate—so-called because it allows us to control the variance of our estimate

Control variates for policy gradients (baselines)

In the same way that we eliminated zero-mean terms in the previous slide we can also add terms,

$$\nabla J(\theta) \approx -\frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T} \sum_{k=t}^{T} \nabla \log \pi_\theta(a_t^i | s_t^i) \left[\gamma^k r(s_k^i, a_k^i) - \hat{b}_k(s_k^i, a_k^i) \right]$$

which is called a baseline, i.e. a “baseline reward” to improve on

This can be interpreted as a control variate of the form

$$\phi(x) = \sum_{t=0}^{T} \sum_{k=t}^{T} \nabla \log \pi_\theta(a_t | s_t) \hat{b}_k(s_k, a_k)$$

which so long as b_k is computed using only state/action pairs before time k will have expectation zero

Choice of baseline

There is some analysis in Greensmith et al. providing an optimal baseline under various settings—a bit complicated (and different from the earlier analysis)

However, a common baseline to use is the averaged reward:

$$\hat{b}_k = \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{K} \gamma^t r(s_t^i, a_t^i)$$

in some sense this is intuitive and gives rise to the baseline name:

by combining this with our gradient the reward provides us with an improvement over the average

Actor-critic methods

Another technique involves using the value function as a baseline,

$$V^\pi(s) = \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) | s_0 = s\Big]$$

which is similar to the averaged-reward baseline presented earlier

Actor-critic methods[1] extend this to using compatible function approximation for the value-function (approximate using a linear function of the policy gradient)

The Natural Actor-Critic takes these ideas and applies the natural gradient. Whether this counts as a variance reduction technique is a bit murky.

[1] [? ? ?]

Continuous dynamic programming: LQR

A continuous control problem

We will now consider a discrete-time, continuous state system which evolves according to:

$$\mathbf{x}_{t+1} = \mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{u}_t \tag{Linear}$$

where $\mathbf{u}$ are the actions; the system has costs (negative rewards) given by

$$c(\mathbf{x}, \mathbf{u}) = \mathbf{x}^\top \mathbf{Q}\mathbf{x} + \mathbf{u}^\top \mathbf{R}\mathbf{u} \tag{Quadratic}$$

The resulting algorithm is the Linear Quadratic Regulator (LQR)

- the state evolves linearly (due to $\mathbf{A}$) where deviations can be corrected by $\mathbf{u}$ in a fashion limited by $\mathbf{B}$
- the cost model penalizes deviations of both the state and the actions away from zero—i.e. we want to get to $\mathbf{x} = 0$ and stay there
- the most common "classical control" problem

Value iteration for LQR

Let's first consider the finite-horizon problem:

$$\mathbf{u}^*_{0:T} = \arg\min_{\mathbf{u}_{0:T}} \sum_{t=0}^{T} \gamma^t c(\mathbf{x}_t, \mathbf{u}_t) \quad \text{given } \mathbf{x}_0$$

and note that we've eliminated the expectation because our model is deterministic! We're minimizing because we have a cost!

As a refresher, value iteration can be used for a T horizon problem by optimally solving the problem at time T, then $T-1$ using the solution at time T, …

To do so we can write the value of being in state $\mathbf{x}$ at time T:

$$\begin{aligned} V_T(\mathbf{x}) &= \min_{\mathbf{u}} c(\mathbf{x}, \mathbf{u}) \\ &= \min_{\mathbf{u}} \left[\mathbf{x}^\top \mathbf{Q}\mathbf{x} + \mathbf{u}^\top \mathbf{R}\mathbf{u}\right] \\ &= \mathbf{x}^\top \mathbf{Q}\mathbf{x} && := \mathbf{x}^\top \mathbf{P}_T \mathbf{x} \end{aligned}$$

Often this is just given as a definition with $\mathbf{u}_T$ being undefined (since it's always zero).

First let's recall how we did value iteration in the discrete case:

$$V_{T-1}(x) = \max_{a} \left[r_x + \gamma \sum_{a'} P(x'|a', x) V_T(x') \right]$$

Similarly (although again we lose the integral) we can write the value function for LQR as:

$$\begin{aligned} V_{T-1}(\mathbf{x}) &= \min_{\mathbf{u}} \left[c(\mathbf{x}, \mathbf{u}) + \gamma V_T(\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}) \right] \\ &= \min_{\mathbf{u}} \left[\mathbf{x}^\top \mathbf{Q}\mathbf{x} + \mathbf{u}^\top \mathbf{R}\mathbf{u} + (\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u})^\top \mathbf{P}_T (\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}) \right] \\ &= \mathbf{x}^\top \mathbf{Q}\mathbf{x} + (\mathbf{A}\mathbf{x})^\top \mathbf{P}_T (\mathbf{A}\mathbf{x}) \\ &\quad + \min_{\mathbf{u}} \Big[\underbrace{\mathbf{u}^\top \mathbf{R}\mathbf{u} + (\mathbf{B}\mathbf{u})^\top \mathbf{P}_T (\mathbf{B}\mathbf{u}) + (\mathbf{A}\mathbf{x})^\top \mathbf{P}_T (\mathbf{B}\mathbf{u})}_{A} \Big] \end{aligned}$$

$$\begin{aligned}\mathbf{u}^* = \arg\min_{\mathbf{u}} A &= \arg\min_{\mathbf{u}} \mathbf{u}^\top (\underbrace{\mathbf{R} + \mathbf{B}^\top \mathbf{P}_T \mathbf{B}}_{\boldsymbol{W}})\mathbf{u} + \underbrace{(\mathbf{A}\boldsymbol{x})^\top \mathbf{P}_T \mathbf{B}}_{\boldsymbol{w}^\top}\,\mathbf{u} \\ &= -\boldsymbol{W}^{-1}\boldsymbol{w}\end{aligned}$$

and the minimum value is

$$\begin{aligned}\min_{\mathbf{u}} A &= -\boldsymbol{w}^\top \boldsymbol{W}^{-1}\boldsymbol{w} \\ &= -\boldsymbol{x}^\top \mathbf{A}^\top \mathbf{P}_T \mathbf{B}(\mathbf{R} + \mathbf{B}^\top \mathbf{P}_T \mathbf{B})^{-1}\mathbf{B}^\top \mathbf{P}_T^\top \mathbf{A}\boldsymbol{x}\end{aligned}$$

Plugging this back into the value function we have:

$$V_{T-1}(\boldsymbol{x}) = \boldsymbol{x}^\top (\underbrace{\mathbf{Q} + \mathbf{A}^\top (\mathbf{P}_T - \mathbf{P}_T \mathbf{B}(\mathbf{R} + \mathbf{B}^\top \mathbf{P}_T \mathbf{B})^{-1}\mathbf{B}^\top \mathbf{P}_T^\top)\mathbf{A}}_{\mathbf{P}_{T-1}})\boldsymbol{x}$$

i.e. we now have a recursive definition for the value function which is parameterized by the quadratic term $\mathbf{P}_t$

Infinite-horizon LQR

In the same way that we defined the infinite-horizon value function for discrete models, we can iterate

$$\mathbf{P}_{i+1} = \mathbf{Q} + \mathbf{A}^\top(\mathbf{P}_i - \mathbf{P}_i\mathbf{B}(\mathbf{R} + \mathbf{B}^\top\mathbf{P}_i\mathbf{B})^{-1}\mathbf{B}^\top\mathbf{P}_i^\top)\mathbf{A}$$

which will converge to the optimal value function,

$$V(\mathbf{x}) = \mathbf{x}^\top\mathbf{P}\mathbf{x}.$$

The optimal policy can be found similarly:

$$\begin{aligned}\mathbf{u}^* &= \arg\min_{\mathbf{u}} \mathbf{u}^\top(\underbrace{\mathbf{R} + \mathbf{B}^\top\mathbf{P}_T\mathbf{B}}_{\mathbf{W}})\mathbf{u} + \underbrace{(\mathbf{A}\mathbf{x})^\top\mathbf{P}_T\mathbf{B}}_{\mathbf{w}^\top}\mathbf{u} \\ &= -\mathbf{W}^{-1}\mathbf{w} = -\mathbf{K}\mathbf{x} \quad \text{where } \mathbf{K} = (\mathbf{R} + \mathbf{B}^\top\mathbf{P}\mathbf{B})^{-1}\mathbf{B}^\top\mathbf{P}\mathbf{A}\end{aligned}$$

Note this can also be found non-iteratively using what is known as the algebraic Riccati equation

Stochastic LQR

We can also consider noisy transitions,

$$\boldsymbol{x}_{t+1} = \boldsymbol{A}\boldsymbol{x}_t + \mathbf{B}\boldsymbol{u}_t + \mathcal{N}(0, \boldsymbol{W}) \qquad \text{(Linear-Gaussian)}$$

which can be solved similarly, Similarly (although again we lose the integral) we can write the value function for LQR as:

$$V_{T-1}(\boldsymbol{x}) = \min_{\boldsymbol{u}} \left[c(\boldsymbol{x}, \boldsymbol{u}) + \gamma \mathbb{E}[V_T(\boldsymbol{x}')] \right]$$

The value function has an additional term due to the noise; but it is uncontrollable (minimizing $\boldsymbol{u}$ cannot affect it). The policy is still:

$$\boldsymbol{u}^* = K\boldsymbol{x}$$

Linear-Quadratic-Gaussian control (LQG)

We can also consider hidden-state models:

$$\mathbf{y}_t = \mathbf{C}\mathbf{x} + \mathcal{N}(0, \mathbf{V})$$

Again a similar approach can be used, but where the controller acts on the states $\hat{\mathbf{x}}$ predicted by a Kalman filter.