Reinforcement Learning Cheat Sheet

Notation

In general, random variables are upper case and the values of the random variable are lower case. Matrices are bold.

Symbol	Meaning
t	a discrete time
S_t	state at time t
A_t	action at time t
R_t	reward at time t
$\mathcal{S}$	set of all non-terminal states
$\mathcal{A}$	set of all actions
$\mathcal{R}$	set of all rewards
$\doteq$	definition equal

Problem setup: Markov decision processes

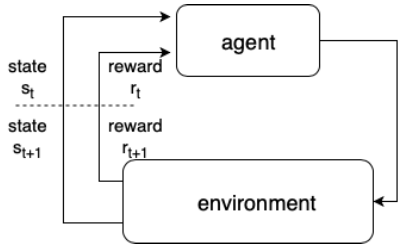

Figure 1: Actor-critic relation

In a Markov decision process (MDP) shown in Figure 1, a game agent interacts with an environment to achieve a certain goal. The interaction happens at every discrete time $t = 1, 2, 3, ...$ The agent observes certain state of the environment $S_t \in \mathcal{S}$, selects some action $A_t \in \mathcal{A}$ and then receives certain reward $R_{t+1} \in \mathcal{R}$. In a finite MDP, $p(s', r|s, a) = Pr\{S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a\}$. The expected reward can be computed by $r(s, a) = \mathbb{E}[R_t|S_{t-1} = s, A_{t-1} = a] = \sum_{r\in\mathcal{R}} r \sum_{s'\in\mathcal{S}} p(s', r|s, a)$.

The goal of a game is typically to maximize the return. The discounted reward can be framed by:

$$G_t \doteq R_{t+1} + \gamma R_{t+1} + ... + \gamma^2 R_{t+2} \doteq \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

$\gamma \in [0, 1]$ is the discount factor.

Actor-critic

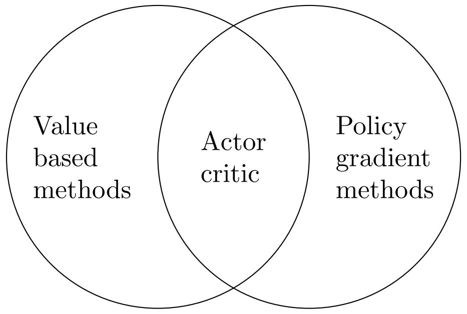

Figure 2: Actor-critic relation

Actor-critic (AC) methods lay between value-based and policy gradient methods as shown in figure 2. They both estimate the policy and state-action functions, whereas value-based methods only estimate state-value functions and have an implicit ϵ-greedy policy, and policy gradient methods do not have value function and only estimates the policy.

Deep Q-learning (DQN)

DQN sits on the basis of many deep RL techniques. It tries to use a deep neural network to estimate the Q-value instead of using a linear method which has been proven to be more effective. The opitmal Q-value should tell us the max reward one can have by taking certain action: $Q^*(s, a) = \mathbb{E}_\pi[G_t|S_t = s, A_t = a]$. The Bellman equation describes that $Q^*(s, a) = \mathbb{E}_\pi[R_t + \gamma \max_{a'} Q^*(s', a'|s, a)]$.

Given a neural network with parameter θ, and the target Q-value $R_t + \gamma \max_{a'} Q^*(s', a'|s, a)$ the update gradient on the loss function can be defined as:

$$\begin{aligned}\nabla_{\theta^-} L(\theta) =& \mathbb{E}[(R_t + \gamma \max_{a'} \\ & Q(s', a'|\theta^-) - Q(s, a|\theta))\nabla Q(s, a|\theta)]\end{aligned}$$

where θ^- is the old network parameter used to estimate the target value. Futhuremore, DQN also incoprates two other techniques to improve the performance, namely experience replay and using a second network to generate the target Q value.

DQN

The benefits of experience replay:

- More data efficiency as one sample might get used more than once.
- Using consecutive samples might not be efficient as consecutive samples might be highly correlated which means that they contain overlapping information. For example, during gameplay, if an agent is in the left half of the environment, the consecutive plays are more likely in the left half as well which might contain overlapping information.
- To make learning sample distribution independent of model parameters. On-policy learning might bias the probability of sample distribution. For instance, if the model thinks the action to move left is greater, then all the states will have greater probabilities of being reached and vice versa. It is easy to see how an agent can get stuck in a local minimum while oscillating between left and right.

Using two networks for the Q value is to reduce variance. When learning, if $Q(s_t, a)$ increases, often $Q(s_{t+1}, a), \forall a \in \mathcal{A}$ increases, which could lead to divergence.

ODEs

1st Order Linear	Use integrating factor, $I = e^{\int P(x)dx}$
Separable:	$\int P(y)dy/dx = \int Q(x)$
HomogEnEous:	$dy/dx = f(x,y) = f(xt,yt)$ sub $y = xV$ solve, then sub $V = y/x$
Exact:	If $M(x,y) + N(x,y)dy/dx = 0$ and $M_y = N_x$ i.e. $\langle M,N\rangle = \nabla F$ then $\int_x M + \int_y N = F$
Order Reduction	Let $v = dy/dx$ then check other types *If purely a function of y,* $\frac{dv}{dx} = v\frac{dv}{dy}$
Variation of Parameters:	When $y'' + a_1y' + a_2y = F(x)$ F contains $\ln x$, $\sec x$, $\tan x$, $\div$
Bernoulli	$y' + P(x)y = Q(x)y^n$ $\div y^n$ $y^{-n}y' + P(x)y^{1-n} = Q(x)$ *Let* $U(x) = y^{1-n}(x)$ $\frac{dU}{dx} = (1-n)y^{-n}\frac{dy}{dx}$ $\frac{1}{1-n}\frac{du}{dx} + P(x)U(x) = Q(x)$ *solve as a 1st order*
Cauchy-Euler	$x^ny^n + a_1x^{n-1}y^{n-1} + \cdots + a_{n-1}y^{n-2} + a_ny = 0$ guess $y = x^r$
3 Cases:	
1) Distinct real roots	$y = ax^{r_1} + bx^{r_2}$
2) Repeated real roots	$y = Ax^r + y_2$ *Guess* $y_2 = x^ru(x)$ *Solve for* $u(x)$ *and choose one* $(A = 1, C = 0)$
3) Distinct complex roots	$y = B_1x^a\cos(b\ln x) + B_2x^a\sin(b\ln x)$

Series Solution

$y'' + p(x)y' + q(x)y = 0$

Useful when $p(x), q(x)$ not constant

Guess $y = \sum_{n=0}^{\infty} a_n(x - x_0)^n$

e^x	$\sum_{n=0}^{\infty} x^n/n!$
$\sin x$	$\sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)!}x^{2n+1}$
$\cos x$	$\sum_{n=0}^{\infty} \frac{(-1)^n}{(2n)!}x^{2n}$

Systems

$\vec{x}' = A\vec{x}$	
A is diagonalizable	$\vec{x}(t) = a_1e^{\lambda_1 t}\vec{v_1} + \cdots + a_ne^{\lambda_n t}\vec{v_n}$
A is not diagonalizable	$\vec{x}(t) = a_1e^{\lambda_1 t}\vec{v_1} + a_2e^{\lambda t}(\vec{w} + t\vec{v})$ where $(A - \lambda I)\vec{w} = \vec{v}$ $\vec{v}$ is an Eigenvector w/ value λ i.e. $\vec{w}$ is a generalized Eigenvector
$\vec{x}' = A\vec{x} + \vec{B}$	Solve y_h $\vec{x_1} = e^{\lambda_1 t}\vec{v_1}, \vec{x_2} = e^{\lambda_2 t}\vec{v_2}$ $\vec{X} = [\vec{x_1}, \vec{x_2}]$ $\vec{X}\vec{u}' = \vec{B}$ $y_p = \vec{X}\vec{u}$ $y = y_h + y_p$

Matrix Exponentiation

$A^n = SD^nS^{-1}$

D is the diagonalization of A

Laplace Transforms

$L[f](s) = \int_0^{\infty} e^{-sx}f(x)dx$

$f(t) = t^n, n \geq 0$	$F(s) = \frac{n!}{s^{n+1}}, s > 0$
$f(t) = e^{at}, a$ *constant*	$F(s) = \frac{1}{s-a}, s > a$
$f(t) = \sin bt, b$ *constant*	$F(s) = \frac{b}{s^2+b^2}, s > 0$
$f(t) = \cos bt, b$ *constant*	$F(s) = \frac{s}{s^2+b^2}, s > 0$
$f(t) = t^{-1/2}$	$F(s) = \frac{\pi}{s^{1/2}}, s > 0$
$f(t) = \delta(t-a)$	$F(s) = e^{-as}$
f'	$L[f'] = sL[f] - f(0)$
f''	$L[f''] = s^2L[f] - sf(0) - f'(0)$
$L[e^{at}f(t)]$	$L[f](s-a)$
$L[u_a(t)f(t-a)]$	$L[f]e^{-as}$

Gaussian Integral

$$\int_{-\infty}^{+\infty} e^{-1/2(\vec{x}^TA\vec{x})} = \frac{\sqrt{2\pi}^n}{\sqrt{\det A}}$$