The simplest RL in an MDP

Notation

→ in X_t ; choose A_t

→ get $(X_{t+1}, R_{t+1}) \sim \underline{P(\cdot \mid X_t, A_t)}$

new state reward

IID

likely not known

Q: estimate

$$V^{\pi}(x) = E\left[\sum_{t=0}^{\infty} \gamma^t R_{t+1} \,\middle|\, X_0 = x\right]$$

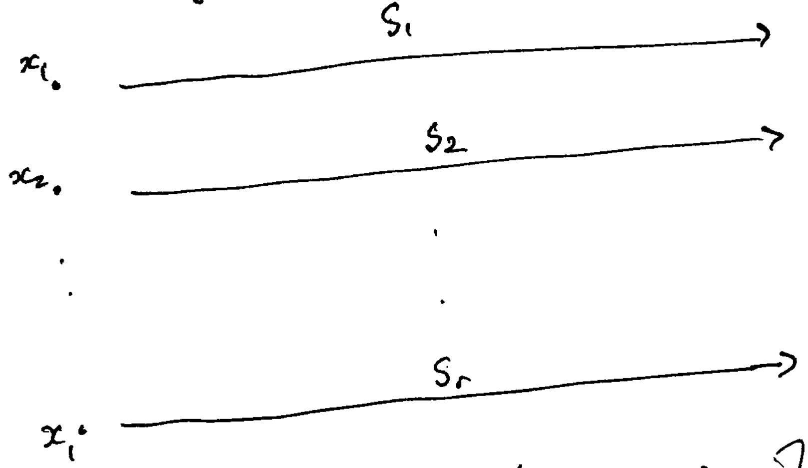
here you act under π.

Simplest

MC estimate

- start at x, run π, observe reward. (this assumes MDP has a terminal state)
- do this many times, average.

and average.



I could form $\frac{1}{\#}\sum$ (Reward of segs starting at x_1)

→ but what if I ~~form~~ get a new seg.?

<u>instead</u>

$$\hat{V}^{\pi}(x) \longrightarrow \hat{V}^{\pi}(x) + \alpha \left[\text{Reward of new seg starting at } x - \hat{V}^{\pi}(x) \right]$$

(Easy) This is unbiased est.

(Easy) But may not be <u>very efficient</u>

Because we have ests of $\hat{V}^{\pi}$(something in x's future)

$$x_1 \xrightarrow{K} x_2 \to x_3 \to x_4 \to \dots$$

a seq.

MC $\hat{V}^\pi(x_1)$ est by sum of discounted rewards along seq.

But we have (say) $\hat{V}^\pi(x_2)$

So another est is
$\hat{V}^\pi(x_1)$ is $R + \gamma \hat{V}^\pi(\cancel{x_2})$.
this has to be unbiased, and <u>might</u> be better

TD(0)

$$\delta_{t+1} = R_{t+1} + \gamma \hat{V}_t(X_{t+1}) - \hat{V}_t(X_t)$$

$$\hat{V}_{t+1}(x) = \hat{V}_t(x) + \alpha\, \delta_{t+1} \mathbb{1}_{[X_t = x]}$$

→ this is temporal difference learning.

→ ~~ea~~ easy to adapt to off-policy seqs.

assume you have

$$(X_t, R_{t+1}, Y_{t+1}) \quad \text{where}$$

$X_t \sim$ some arb. ergodic MC over $\mathcal{X}$.

$$(Y_{t+1}, R_{t+1}) \sim P(\sim \mid X_t, A_t)$$

from the policy

then

$$\delta_{t+1} = R_{t+1} + \gamma \hat{V}(Y_{t+1}) - \hat{V}(X_t)$$

works.

Which is better? **Depends**

example in Szepesvari, p 22 in model that shows small change in model that favors either TD(0), MC, depending on state.

Notice that TD(0), MC, repre
two points on a family of algs.

MC: est by counting reward from x start

est by multiple steps of reward, then pass to est $=$ TD(λ).

TD(0): est by reward from one step, then add est. of new state

TD(λ) — eqns p 24 of Szepesvari.

What if there are too many states?

eg. continuous, etc.

- estimate $\hat{V}(s)$ with some form of function approximator. (Neural network these days)

TD(λ) then updates θ rather than V directly

(Szepesvari, p 21 gives eqns.
→ this can diverge)

But this ...

two fairly natural strategies.

- Build a model of the Q function

$$Q^*(x,a) = E_{\text{(best policy)}}\left[\sum_{t=0}^{\infty} \gamma^t R_{t+1} \,\middle|\, X_0 = x, A_0 = a\right]$$

- particularly nice if I can do this without looking at seqs: observe $X_t, A_t, R_{t+1}, Y_{t+1}$

$$\delta_{t+1} = R_{t+1} + \gamma \max_{a' \in A} Q(Y_{t+1}, a') - Q(X_t, A_t)$$

$$Q(x,a) \to Q(x,a) + \alpha\, \delta_{t+1} \cdot \mathbb{I}_{\{x = X,\, a = A\}}$$

NOTICE : - TD update

- $(Y_{t+1}, R_{t+1}) \sim P(\cdot \mid X_t, A_t)$

- But we don't need a seq (as long as X_t is ergodic.

eg. i.e. you can build a local model, which
is rather nice for many problems

- Alternative: build a model of policy, update.

policy model: parametric probability model $\pi_\theta = p(a_t | s_t, \theta)$

common is some form of neural network predicting means, ~~var~~ log sds of a Gaussian policy.

- update by gradient ascent:

$$\theta_{k+1} = \theta_k + \alpha \nabla_\theta J(\pi_\theta) \Big|_{\theta_k}$$

expected return

$$J(\pi_\theta)\Big|_{\theta_t} = \underbrace{\mathop{\mathbb{E}}_{\tau\sim\pi_\theta}\left[R(\tau)\right]}$$

↳ expected return under π_θ

~~assume~~ yo

- write $\tau = (s_0, a_0, s_1 \;\ldots\; s_{T+1})$ trajectory

$$P(\tau|\theta) = \rho_0(s_0) \cdot \prod_{t=0}^{T} \left\{ P(s_{t+1}|s_t, a_t) \cdot \pi_\theta(a_t|s_t) \right\}$$

prior of starting in 0

- Notice $\nabla_\theta \log f(\theta) = \frac{1}{f(\theta)} \cdot \nabla_\theta f(\theta)$

(logarithmic diff reinvented!)

$$\log P(\tau|\theta) = \log \rho_0 + \sum_{t=0} \left[\log P(s_{t+1}|a_t, s_t) + \log \pi_\theta(a_t|s_t) \right]$$

NO θ!

So $\nabla_\theta \log P(\tau|\theta)$

$$= \sum_{t=0}^{T} \left[\log \pi_\theta(a_t|s_t) \right]$$

So:

$$\nabla_\theta J = \nabla_\theta \int_\tau P(\tau|\theta)\, R(t)$$

$$= \int_\tau \nabla_\theta P(\tau|\theta)\, R(\tau)$$

$$= \int_\tau P\, \nabla_\theta \log P \quad R(\tau)$$

$$= \mathop{E}_{\tau} \left[\sum_{t=0}^{\infty} \nabla_\theta \log \pi(a_t|s_t) \quad R(\tau) \right]$$

But this is reward for whole trajectory

Expectation!

some funny thing the implication

(OpenAI' notes) gets

$$\nabla_\theta J = \mathop{E}_{\tau \sim \pi_\theta}\left[\sum_{t=0}^{T} \nabla_\theta \log \pi \left(\sum_{u=t}^{T} R(s_u, a_u, s_{u+1}) \right) \right]$$

<u>this is a better estimate</u>:

we had $\nabla_\theta J = \mathop{E}_{\tau}\left[f_1(\tau) + f_2(\tau) \right]$

<u>But</u> $\mathop{E}_{\tau}\left[f_1(\tau) \right] = 0$

→ removing f_1 reduces variance – this is not just manipulation for showing off.

an importance strategy.

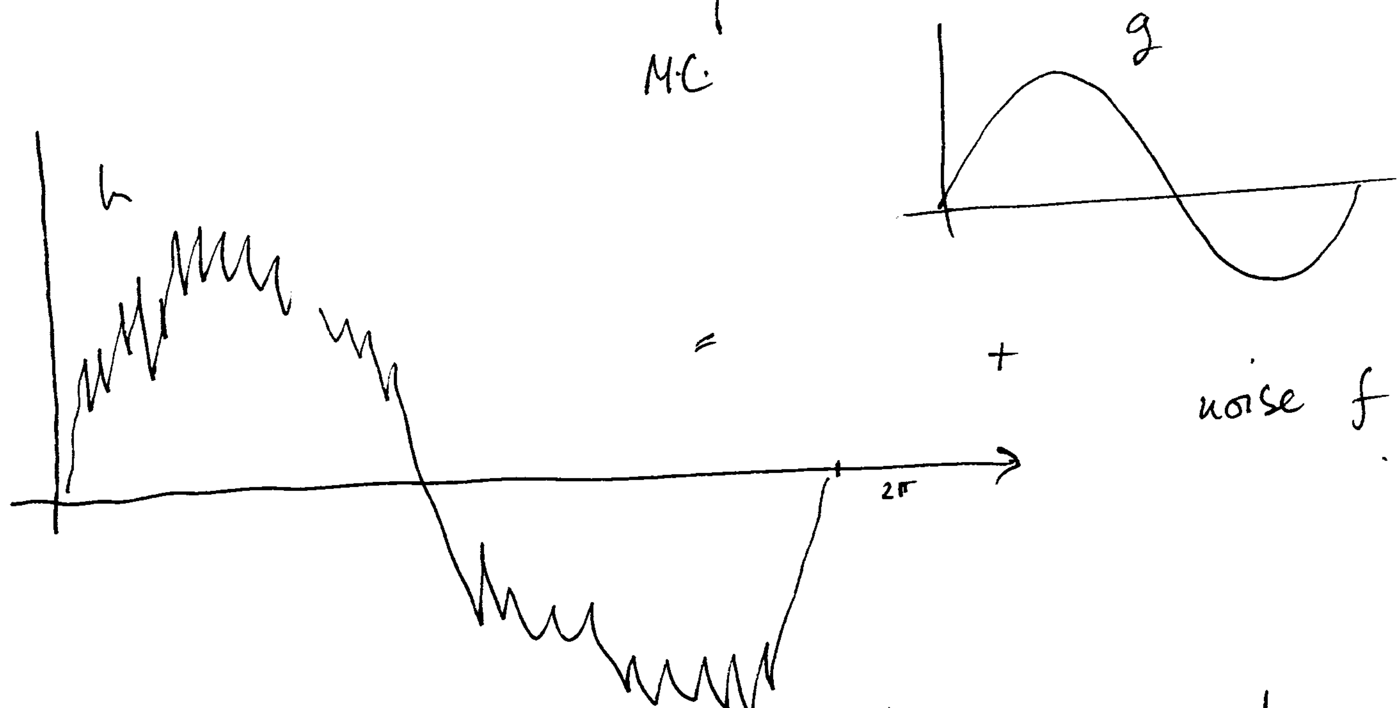
assume I want to estimate $E_p[h] =$

$$E_p[\, f + g \,]$$

f: hard to integrate

g: easy to integrate.

<u>Always</u> use $\underbrace{E_p[f]}_{\text{M.C.}}$ + [easy est of g]



→ notice that h adds large terms to sum, which then <u>cancel</u>

(or should, but don't precisely)

<u>Now</u>

$$\mathop{E}_{a_t \sim \pi}\left[\nabla_\theta \log \pi(a_t|s_t) \cdot b(s_t)\right]$$

$$= \left\{ \int \left[\underbrace{P(a_t|s_t)}_{\pi(a_t|s_t)} \cdot \nabla_\theta \log \pi(a_t|s_t)\right] \right\} b(s_t) .$$

<u>But</u> $\int P \nabla_\theta \log P = \int \nabla_\theta P = \nabla_\theta \int P = 0$

$$\therefore \quad \mathop{E}_{a_t \sim \pi}\left[\nabla_\theta \log \pi(a_t|s_t)\, b(s_t)\right] = 0$$

often referred to as a <u>baseline</u>

this means

$$\mathop{E}_{\tau}\left[\nabla_\theta \log \pi(a_t|s_t) \cdot b(s_t)\right] = 0$$

Natural choice of baseline

$$V^{\pi}(s_t) \leftarrow \text{est as above}$$

$$\nabla_{\theta} J(\pi_{\theta}) = \underset{\tau}{E}\left[\sum_{t=0}^{T} \nabla_{\theta} \log \pi(a_t|s_t)\left(\sum_{u=t}^{T} R(s_u, a_u, s_{u+1}) - V^{\pi}(s_u)\right.\right.$$

other base lines

$Q^{\pi}(s_t, a_t)$ works.

$Q^{\pi}(s_t, a_t) - V^{\pi}(s_t)$ works

Which is best?

Don't know

What does all [illegible]

- BE CAREFUL
  - the reason there are lots of methods is they mostly DON'T WORK RELIABLY.

- best cases seem to be ones you could do with LQR, but the linearization is a nuisance.
  - velocity control of vehicle
  - stopping at right place from a single view of stop sign

- Important features: important
  - local reward is quite
  - you can generate a ton of examples from simulator
  - var control is easy.

However

You can est Var[r] , Cov[ ]
from MC replicates to get variance reduction.

This suggests a simple way to get improved var in Policy gradient, cause you know $\int$ baseline.

This idea extends :

- under some circs, you don't need to know r
  (g must be easy to integrate, and a covariance cond)

- you can use multiple control variates.

Control variates method

we want to know

$$I_f = \int f\, P\, dx = E_p[f]$$

assume we have g, st $\underbrace{E_p[g] = \gamma}_{\text{KNOWN}}$

then

$$\frac{1}{N}\sum_i \left[f(x_i) + c(g(x_i) - \gamma)\right] = I_f + \xi$$

where $E[\xi] = 0$

$$\mathrm{Var}(\xi) = \mathrm{Var}\left[\text{est of } f \text{ from } \frac{1}{N}\sum_i f(x_i)\right]$$
$$+ c^2\,\mathrm{Var}\left[\text{est of } \gamma \text{ from } \frac{1}{N}\sum g(x_i)\right]$$
$$+ 2c\,\mathrm{Cov}\left[\quad , \quad\right]$$

<u>whence</u> best c $\left(= -\dfrac{\mathrm{Cov}}{\mathrm{Var}[\gamma]}\right)$

usually unknown.