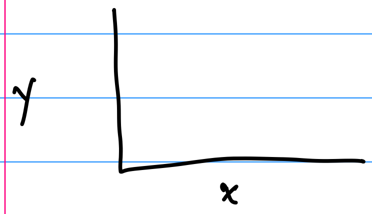
Project Ideas
HW 4 Friday




```
xs = [ ]
ys = [ ]
x = 0
for i in 1:10000
    reset!(env)
    while !terminated(env)
        s = observe(env)
        a = explore(s)
        r = act!(env, a)
        x += 1
        update Q
    if i % 100 == 0
        push!(xs, x)
        push!(ys, eval(env, Q))
plot(xs, ys)
```

learning episode

```
function eval(env, Q)
    rsum = 0
    for i in 1:1000
        reset!(env)
        while !terminated(env)
            s = observe(env)
            a = argmax(Q(s,a))
            rsum +=   act!(env,a)
    return rsum/1000
```

evaluation episode

Last Time

Neural Networks

$$f_\theta(x) = W_1\sigma(W_2\sigma(W_3x + b_3) + b_2) + b_1$$

$$\theta = (W_1, b_1, W_2, b_2, W_3, b_3)$$

Stochastic Gradient Descent with backprop

This Time

RL with Neural Networks

DQN DPG

Approximate $Q(s,a)$ with $Q_\theta(s,a)$

Review: Q-learning update

$$Q(s,a) = Q(s,a) + \alpha\left(\underline{r + \gamma \max_{a'} Q(s',a')} - Q(s,a)\right)$$

Deep Q learning

x,y

```
loop
    s = observe(env)
    a = explore(s)
    r = act!(env, s)
    s' = observe(env)
    y = r + γ max_a' Q(s', a')
    θ ← θ + α ∇_θ Q(s,a)(y − Q(s,a))
```

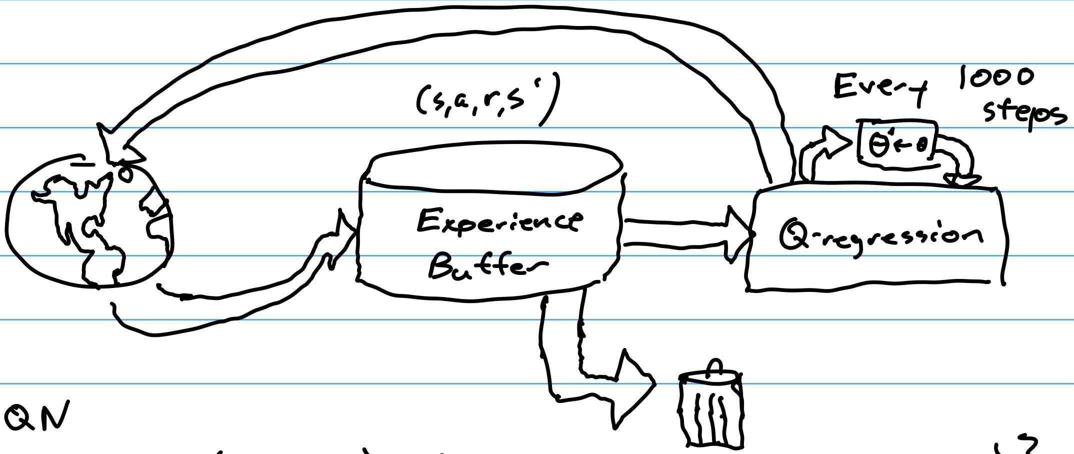
This won't work ... at all.

(s,a,r,s')

1. Samples highly correlated
2. Size-1 batches
3. Moving Target

} use experience buffer

} periodically freeze θ



"Classic" DQN

$$l(s,a,r,s') = \left(r + \gamma \max_{a'} \underline{Q_{\theta'}(s',a')} - Q_\theta(s,a)\right)^2$$

$r + \gamma Q_1(s', \arg\max_{a'} Q_2(s', a'))$

Rainbow

- Double Q-Learning
- Prioritized replay
  - priority proportional to last TD error

(s, a, r, s')

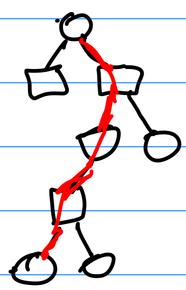
- Dueling networks
  - Value network + advantage network
  - $Q(s,a) = V(s) + A(s,a)$
- Multi-step learning
  - $(r_t + \gamma r_{t+1} + \dots + \gamma^{n-1} r_{t+n-1} + \gamma^n \max_{a'} Q_{\theta'}(s_{t+n}, a') - Q(s,a))^2$
- Distributional RL
  - predict a distribution of returns
- Noisy nets
  - add noise to neural network

Breakout Room

What is the difference between MCTS, SARSA-λ?

$Q(s,a)$ $N(s,a)$

MCTS	SARSA-λ
Rollouts	Q-values
UCB1	ϵ-greedy
N used for exploration	N values used for eligibility
Tree	No tree
Model-Based	Model-Free
Online - decision at current state	Offline - policy



log trick
causality
baselines

DPG

$$\nabla_\theta U(\theta) = E_\tau\left[\sum_{k=1}^{d} \nabla_\theta \log \pi_\theta(a^{(k)} \mid s^{(k)})\, \gamma^{(k-1)} \left(r^{(k)}_{\text{to-go}} - r_{\text{base}}(s^{(k)})\right)\right]$$

$$U(\theta') \approx U(\theta) + \nabla U(\theta)^T(\theta' - \theta)$$

$$g(\theta,\theta') = \tfrac{1}{2}(\theta' - \theta)^T I(\theta' - \theta) = \tfrac{1}{2}\|\theta' - \theta\|_2^2$$

$$\underset{\theta'}{\text{maximize}} \quad U(\theta) + \nabla U(\theta)^T(\theta' - \theta)$$

$$\text{subject to} \quad g(\theta,\theta') \le \epsilon$$

analytic solution

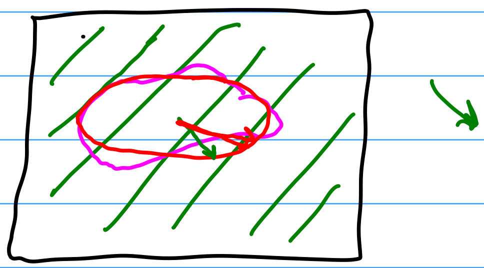
$$\theta' = \theta + u\sqrt{\frac{2\epsilon}{u^T u}} = \theta + \sqrt{2\epsilon}\,\frac{u}{\|u\|}$$

$$u = \nabla U(\theta)$$

Natural Gradient

$$g(\theta,\theta') = D_{KL}(p(\cdot \mid \theta) \,\|\, p(\cdot \mid \theta')) \le \epsilon$$

θ_2



θ_1

$$g(\theta,\theta') = \tfrac{1}{2}(\theta' - \theta)^T F_\theta (\theta' - \theta) \le \epsilon$$

Taylor approximation

$$F_\theta = E_\tau\left[\nabla \log p(\tau \mid \theta)\, \nabla \log p(\tau \mid \theta)^T\right]$$

$$\underset{\theta'}{\text{maximize}} \quad \nabla U(\theta)^T(\theta' - \theta)$$

$$\text{subject to} \quad \tfrac{1}{2}(\theta' - \theta)^T F_\theta (\theta' - \theta) = \epsilon$$

analytical

$$\theta' = \theta + u\sqrt{\frac{2\epsilon}{\nabla U(\theta)^T u}}$$

$$u = F_\theta^{-1} \nabla U(\theta)$$

TRPO
PPO