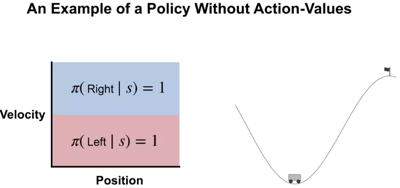
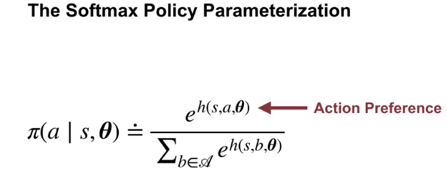
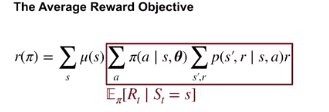
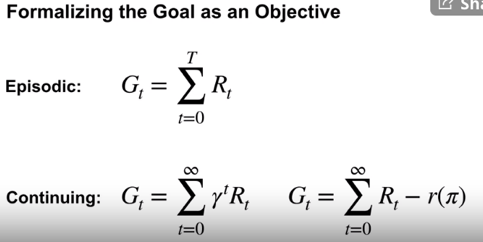
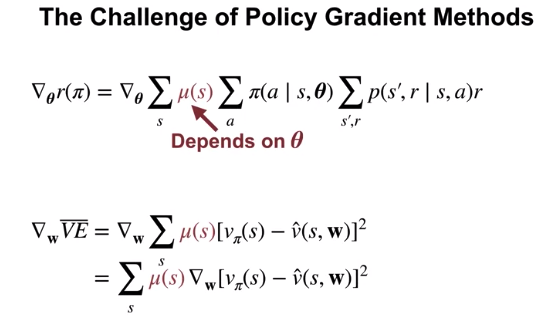
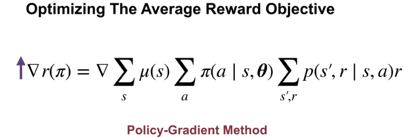
# Week4 Notes

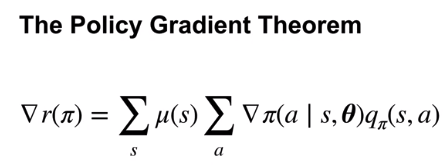
**Lesson 1: Learning Parameterized Policies**

* Understand how to define policies as parameterized functions
  + 
* Define one class of parameterized policies based on the softmax function
* Understand the advantages of using parameterized policies over action-value based methods
  + Flexibility of stochastic policies
  + Stochastic policies might be better than deterministic under fn approximation

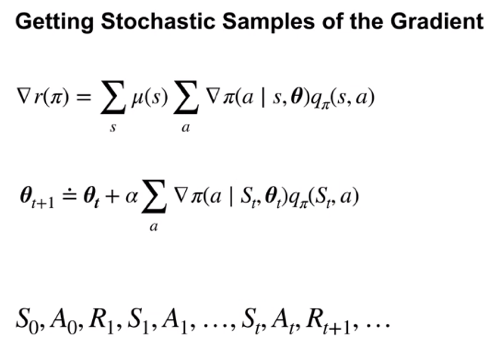
**Lesson 2: Policy Gradient for Continuing Tasks**

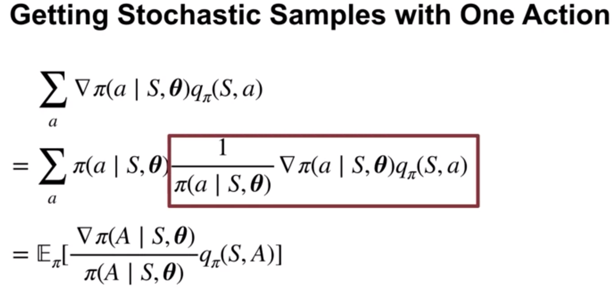
* Describe the objective for policy gradient algorithms
* 

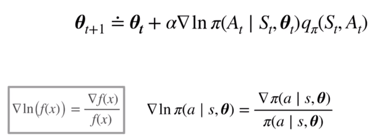
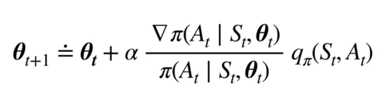


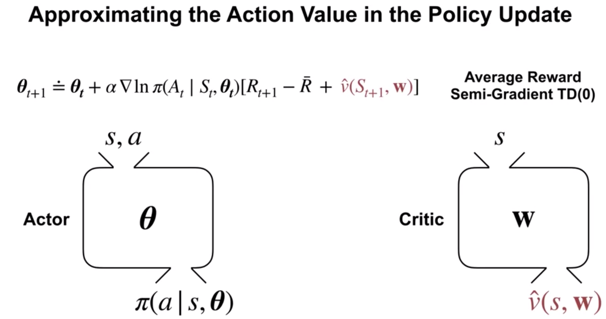
* Describe the results of the policy gradient theorem
  + 
* Understand the importance of the policy gradient theorem

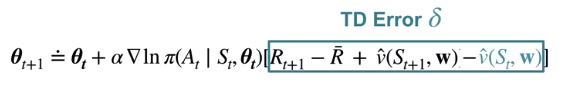
**Lesson 3: Actor-Critic for Continuing Tasks**

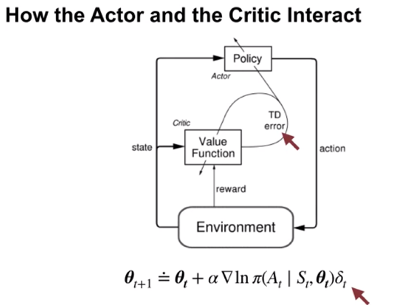
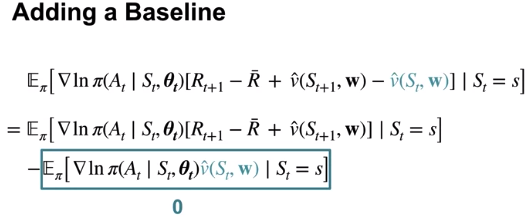
* Derive a sample-based estimate for the gradient of the average reward objective
* 

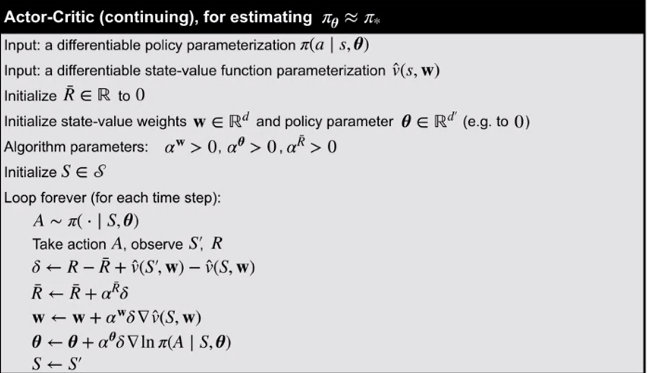




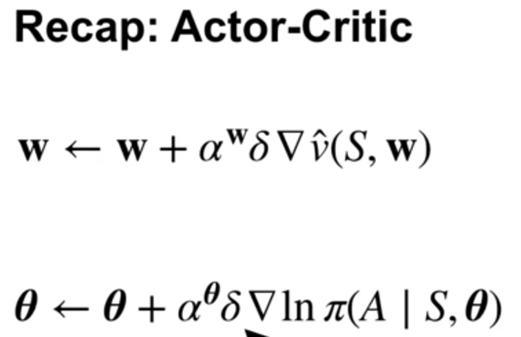
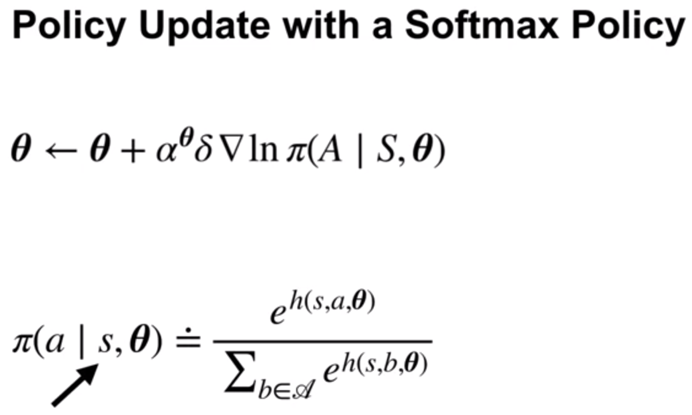
* Describe the actor-critic algorithm for control with function approximation, for continuing tasks
* 

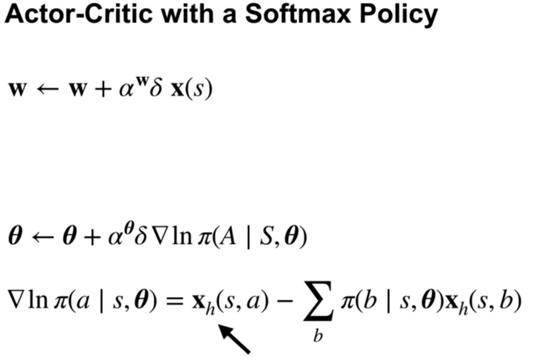


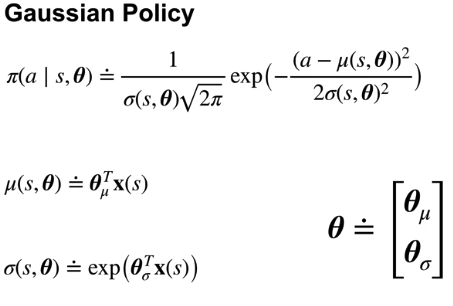
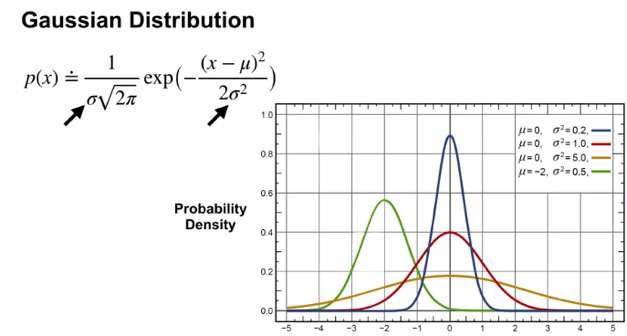
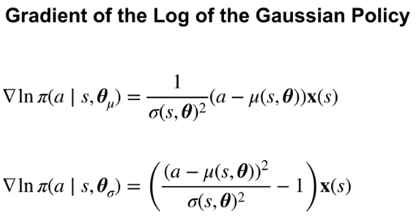
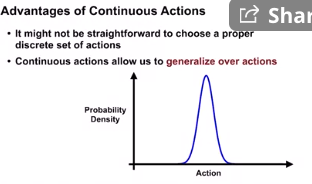




**Lesson 4: Policy Parameterizations**

* Derive the actor-critic update for a softmax policy with linear action preferences
* 
* 



* Implement this algorithm
* Design concrete function approximators for an average reward actor-critic algorithm
* Analyze the performance of an average reward agent
* Derive the actor-critic update for a gaussian policy
* 
* Apply average reward actor-critic with a gaussian policy to a particular task with continuous actions
* 
* 

Assignment

When performing updates to the Actor and Critic, recall their respective updates in the Actor-Critic algorithm video.

We approximate 𝑞𝜋qπ in the Actor update using one-step bootstrapped return(𝑅𝑡+1−𝑅¯+𝑣̂ (𝑆𝑡+1,𝐰)Rt+1−R¯+v^(St+1,w)) subtracted by current state-value(𝑣̂ (𝑆𝑡,𝐰)v^(St,w)), equivalent to TD error 𝛿δ.

𝛿𝑡=𝑅𝑡+1−𝑅¯+𝑣̂ (𝑆𝑡+1,𝐰)−𝑣̂ (𝑆𝑡,𝐰)(1)δt=Rt+1−R¯+v^(St+1,w)−v^(St,w)(1)

**Average Reward update rule**: 𝑅¯←𝑅¯+𝛼𝑅¯𝛿(2)R¯←R¯+αR¯δ(2)

**Critic weight update rule**: 𝐰←𝐰+𝛼𝐰𝛿∇𝑣̂ (𝑠,𝐰)(3)w←w+αwδ∇v^(s,w)(3)

**Actor weight update rule**: 𝜃←𝜃+𝛼𝜃𝛿∇𝑙𝑛𝜋(𝐴|𝑆,𝜃)(4)θ←θ+αθδ∇lnπ(A|S,θ)(4)

However, since we are using linear function approximation and parameterizing a softmax policy, the above update rule can be further simplified using:

∇𝑣̂ (𝑠,𝐰)=𝐱(𝑠)(5)∇v^(s,w)=x(s)(5)

∇𝑙𝑛𝜋(𝐴|𝑆,𝜃)=𝐱ℎ(𝑠,𝑎)−∑𝑏𝜋(𝑏|𝑠,𝜃)𝐱ℎ(𝑠,𝑏)(6)