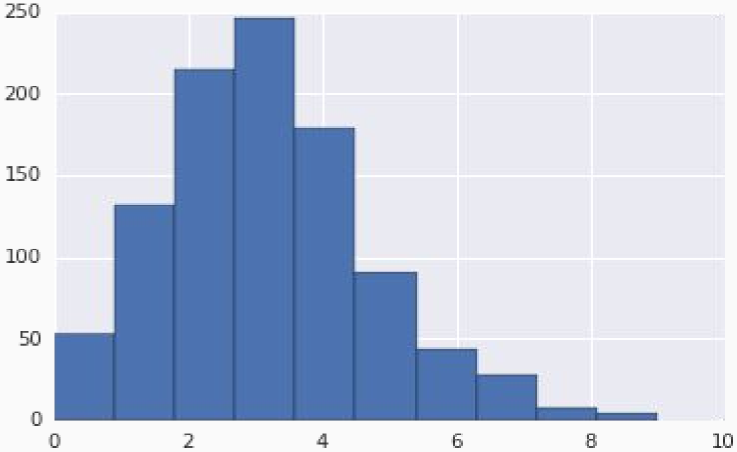
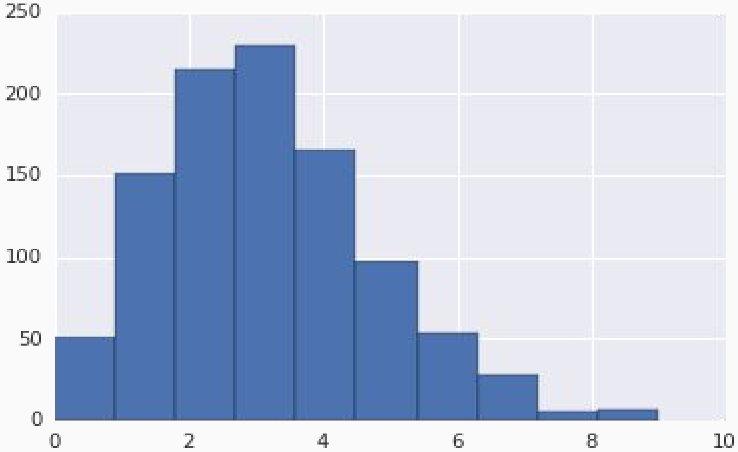
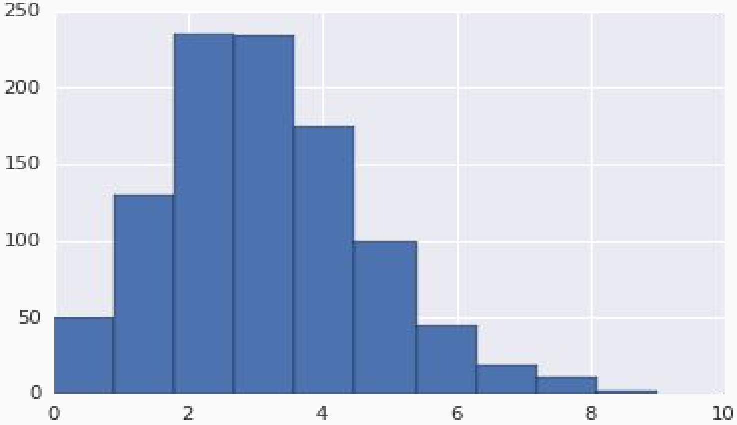
Rental Car Problem

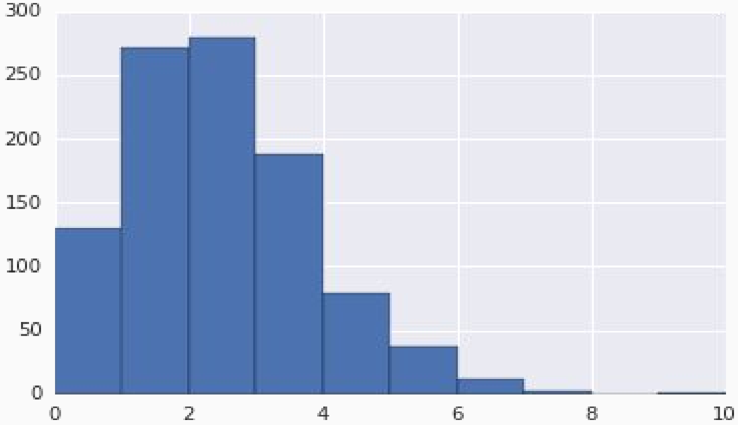
Problem Description

- 2 Car Renting Locations (Maximum car capacity 6)
- Poisson Distribution for arrivals (mean 2 and 3)
- Poisson Distribution for returns (mean 2 and 1)
- 2$ for moving car from one location to another
- Maximum 3 cars can be moved in a day
- 10$ Reward for every rented car







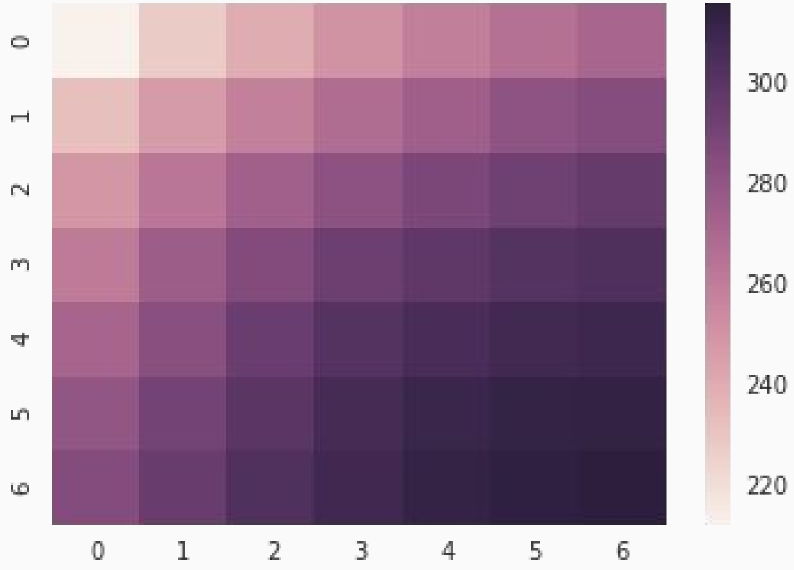


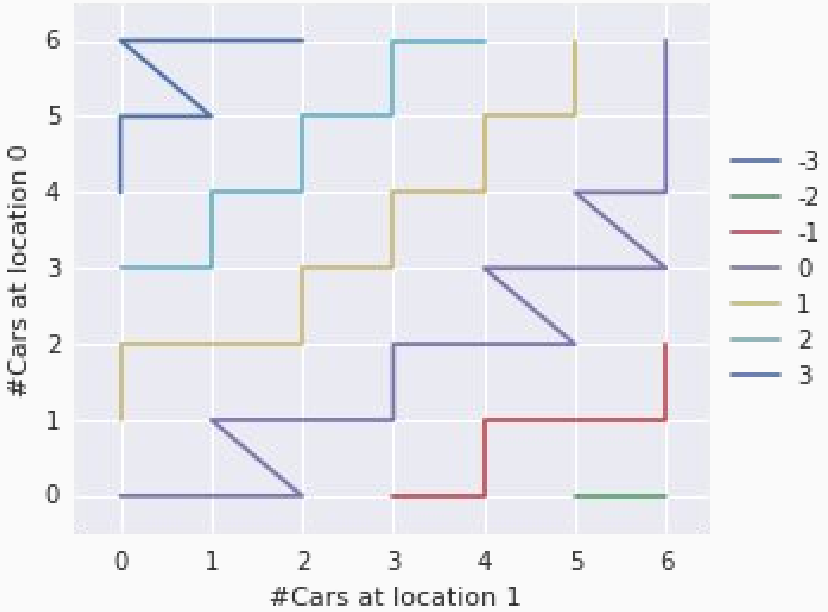
Poisson Distributions for Arrivals and Returns

Approach

- State: (#Cars at Location0, #Cars at Location1)
- Action: #Cars moved at night
- Algorithm: Policy Iteration

Optimal Policy & State Values

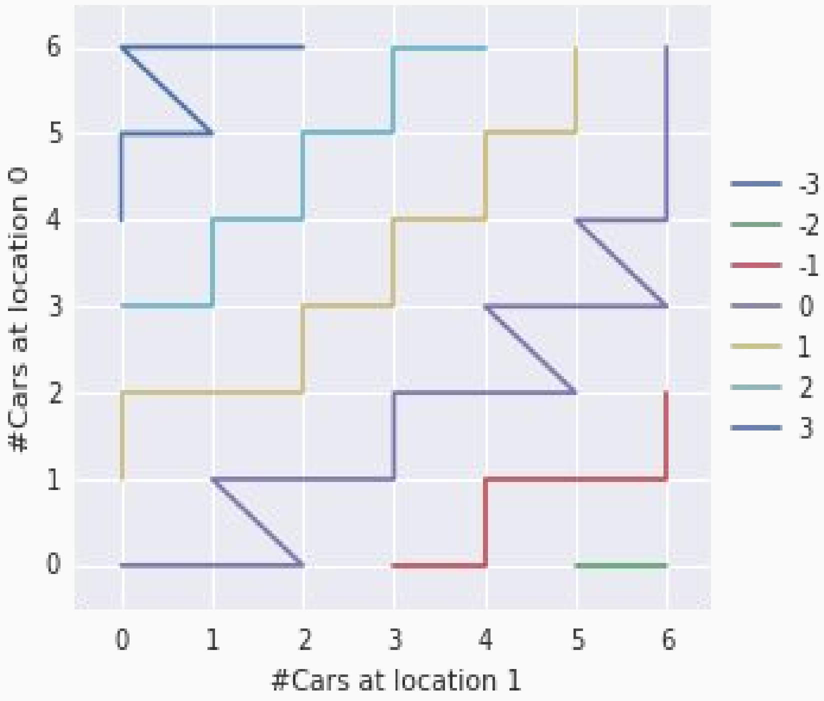




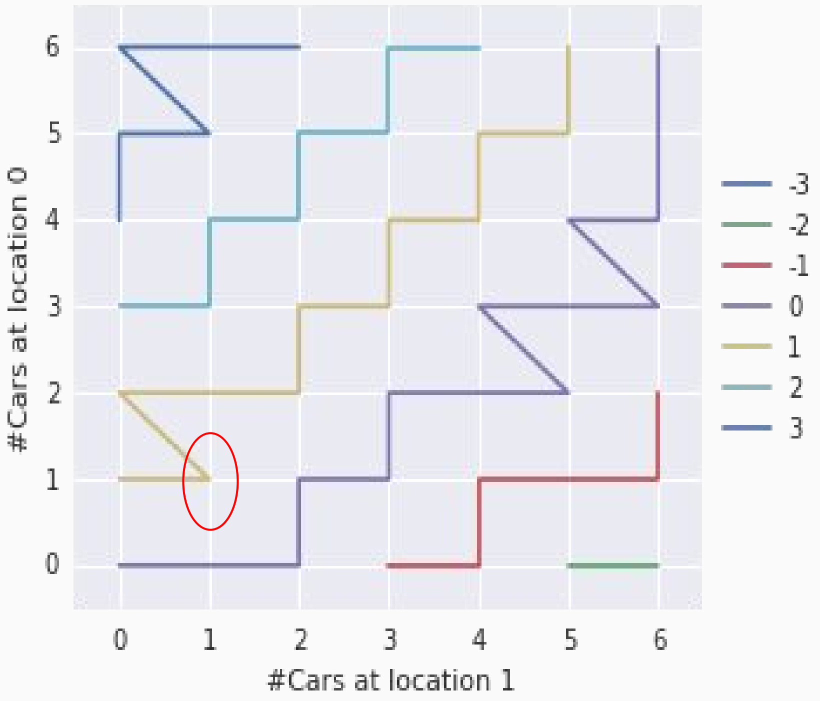
Modified Version

- Additional Cost of $3 for parking if #Cars exceeds 3
- 1 car can be moved for free from Location 0 to 1

What do you expect ?



Original Problem



Modified Version

Observations

- Free transportation of 1 car causes action of moving car from location 0 - 1 to be selected more.
- But additional parking cost didn’t have any impact.

Problem with current approach ?

- Infeasible for large State Space

Can you identify the bottleneck ?

Policy iteration (using iterative policy evaluation)

1. Initialization
   $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$

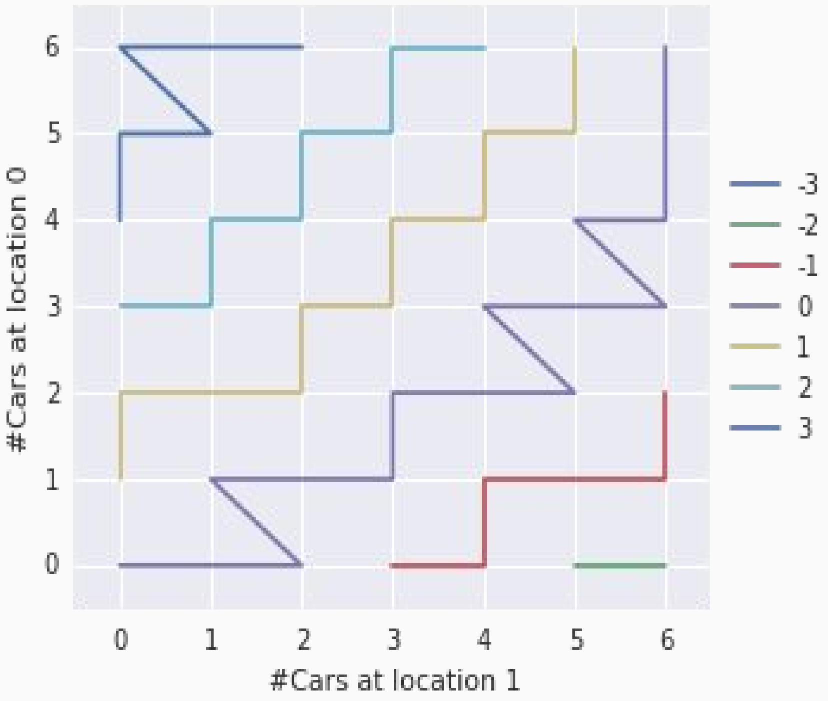
2. Policy Evaluation
   Repeat
   $\quad \Delta \leftarrow 0$
   $\quad$ For each $s \in \mathcal{S}$:
   $\quad\quad v \leftarrow V(s)$
   $\quad\quad V(s) \leftarrow \sum_{s',r} p(s',r|s,\pi(s))\left[r + \gamma V(s')\right]$
   $\quad\quad \Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$ (a small positive number)

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
   $\quad$ *old-action* $\leftarrow \pi(s)$
   $\quad \pi(s) \leftarrow \operatorname{argmax}_a \sum_{s',r} p(s',r|s,a)\left[r + \gamma V(s')\right]$
   $\quad$ If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*
   If *policy-stable*, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

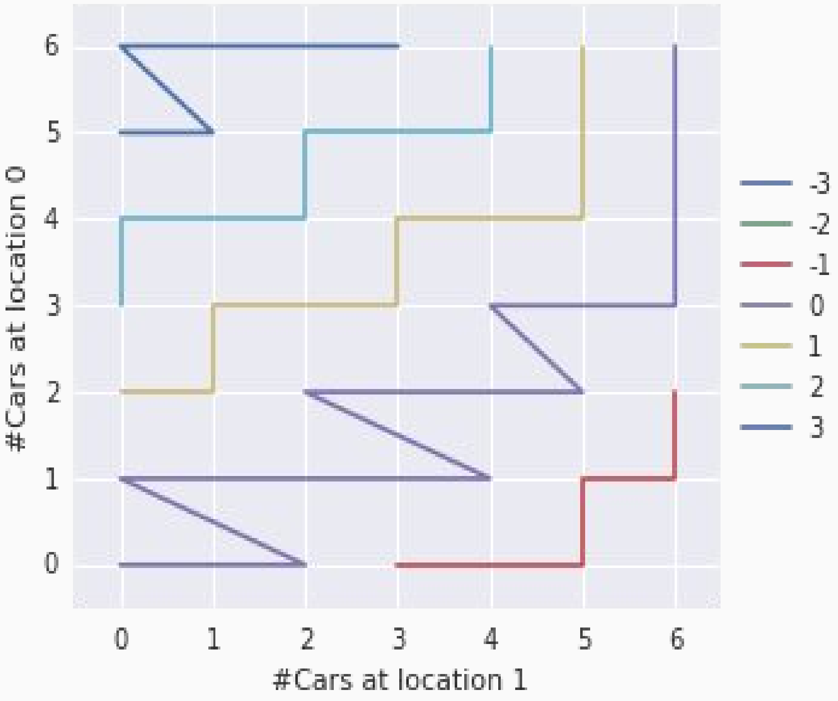
Ideas ??

- State Values seem to be learnable by simple model
- Approximate expected poisson reward by stochastic sampling
- Can we use something else ?

Policies for Poisson and Uniform distributions

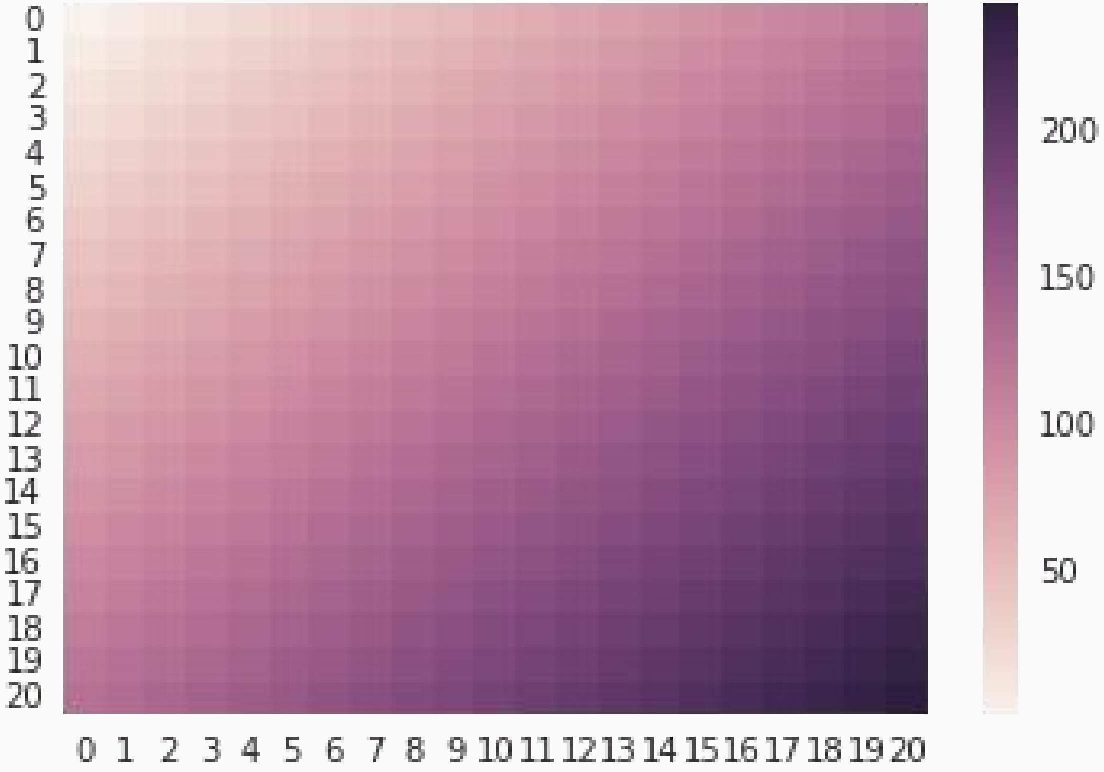


Poisson Distribution



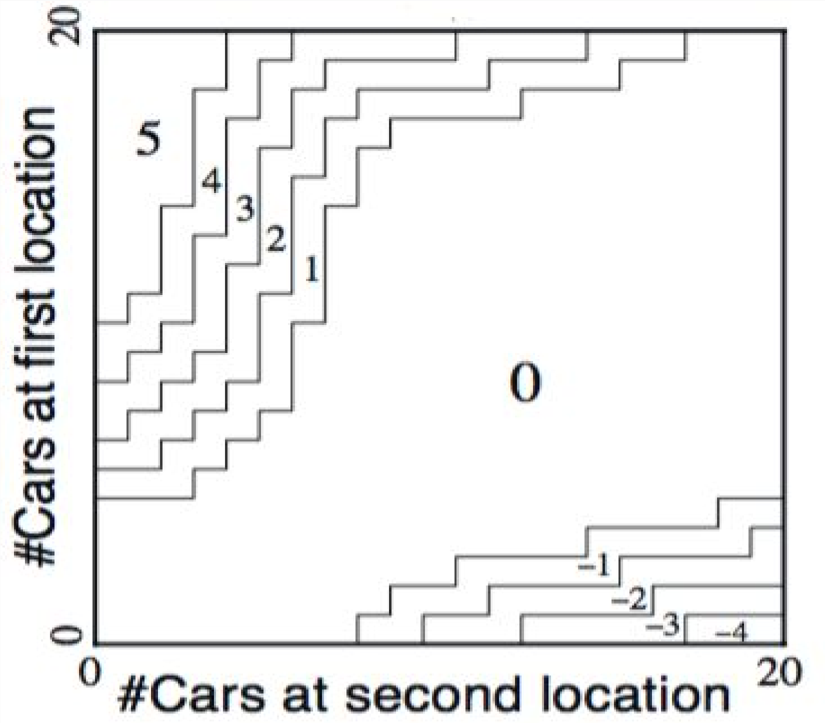
Uniform Distribution

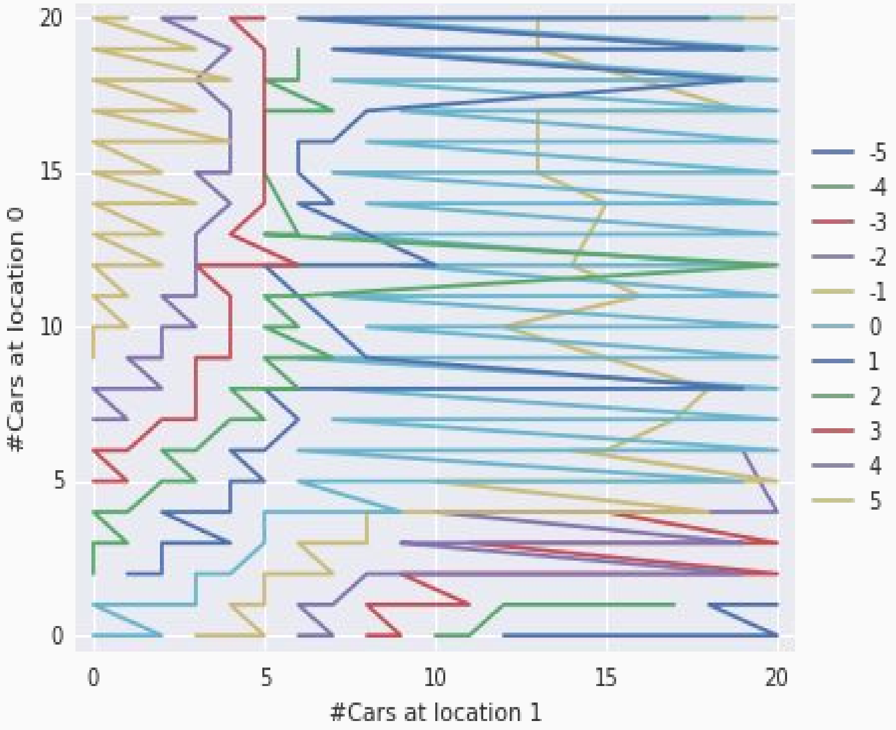
Learnt State Values



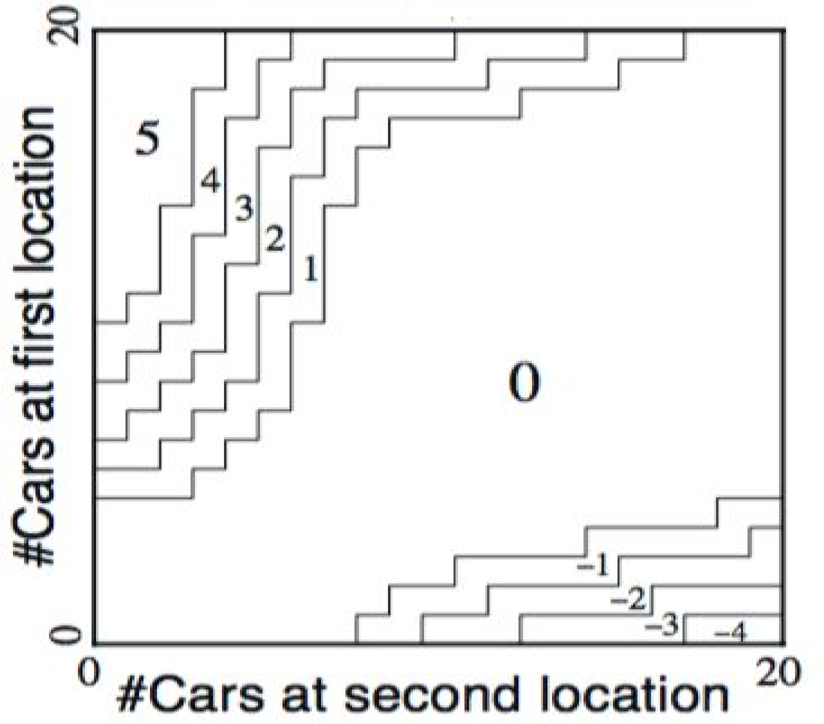
Which expectation approximation technique do you expect to be better ?

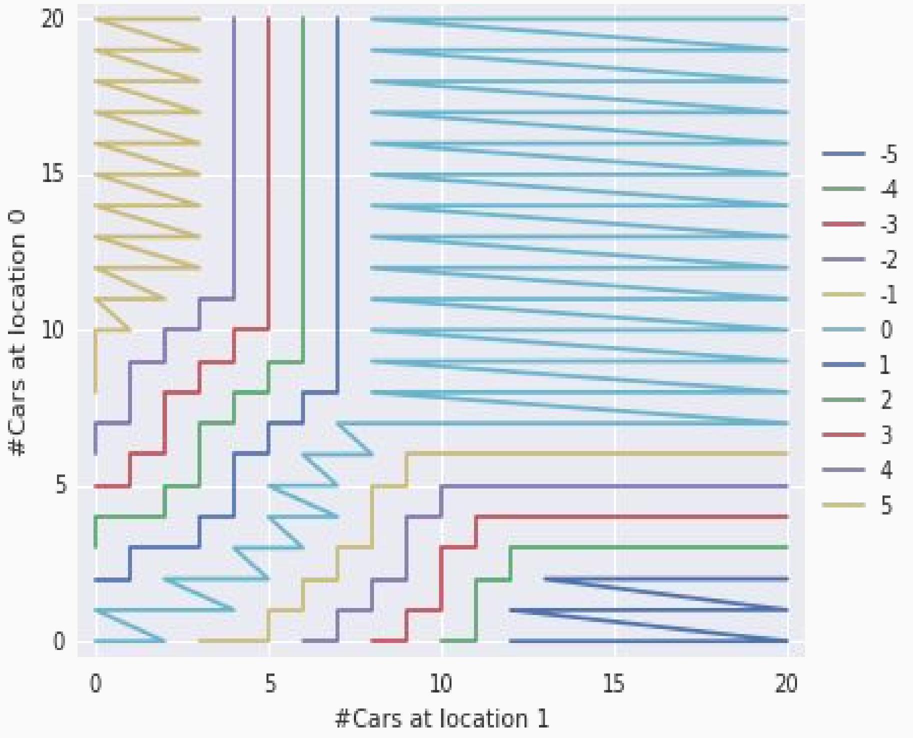
Actual vs Estimate (Poisson sampling)



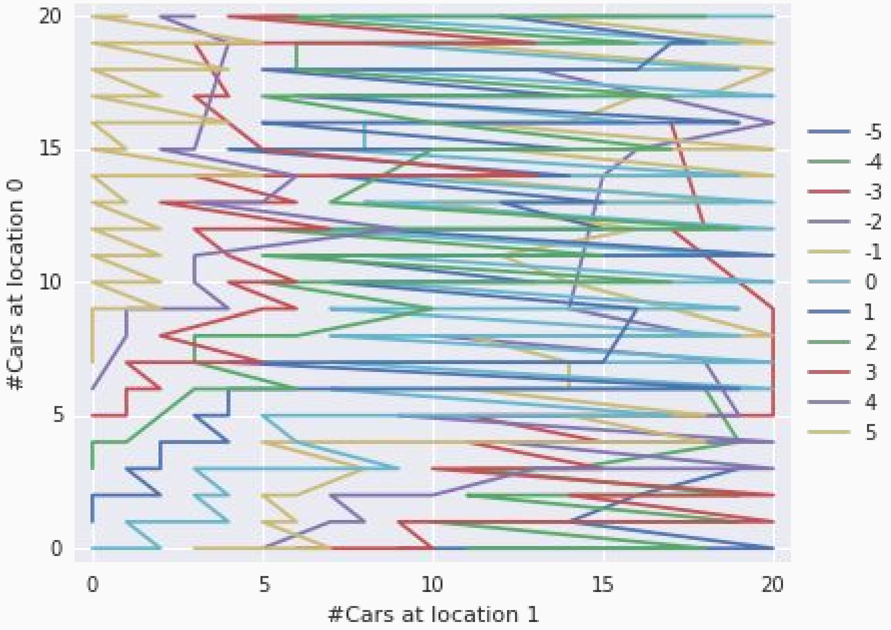


Optimal vs Estimate (Uniform Distribution)

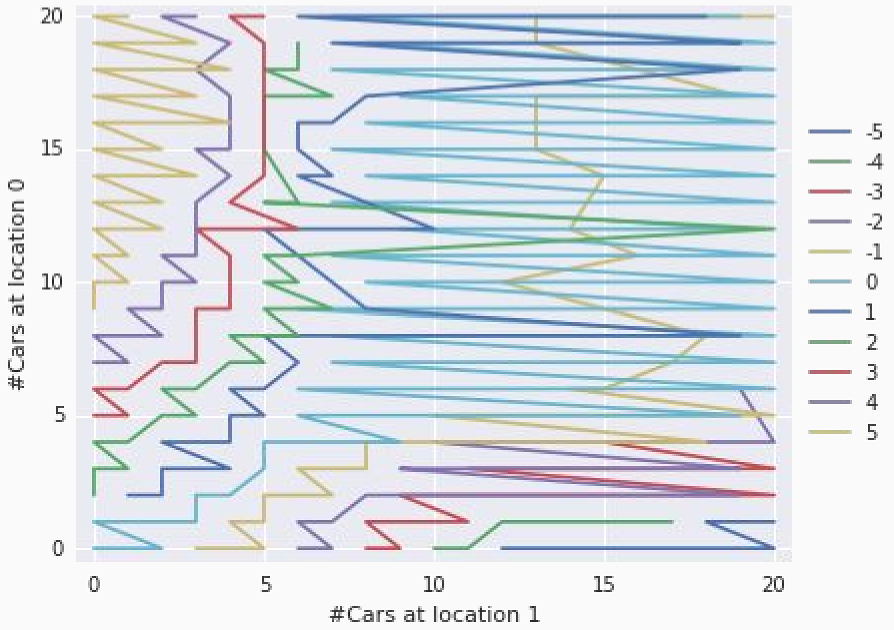




Why was poisson approximation worse ?

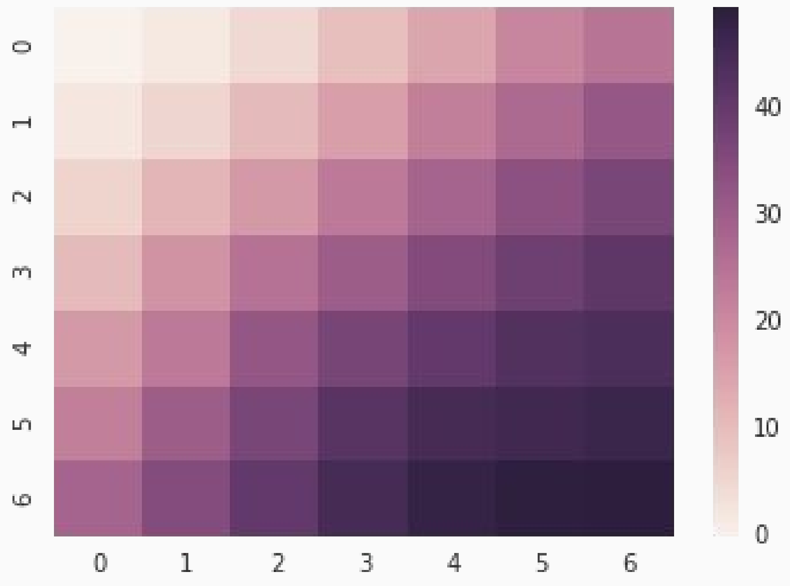


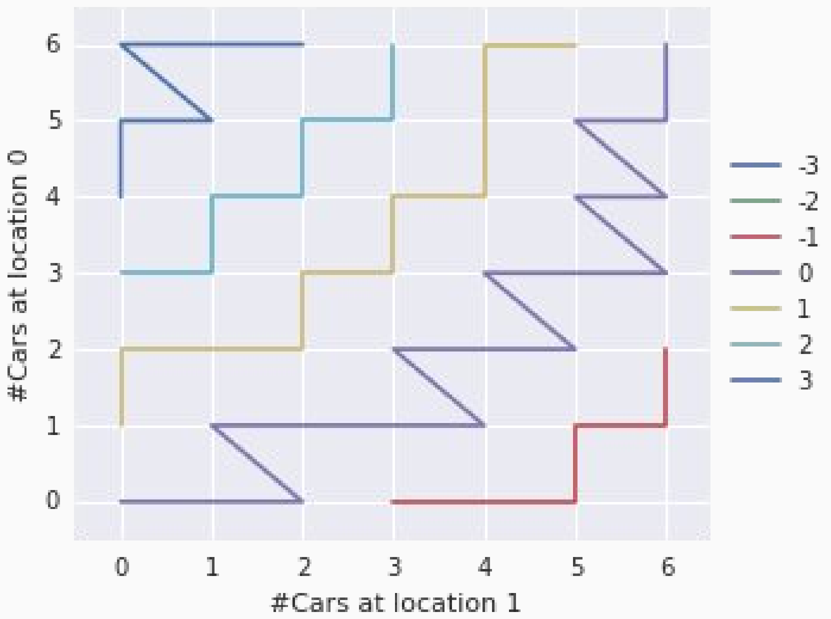
100 samples



1000 Samples

Optimal Policy & State Values (Discount 0)





Why did discount have negligible effect on policy ?

- State values reduce almost in same proportion.
- For larger state space discount might have significant effect

What else could we have done ?

- Predicting policy values directly !!

Key Takeaways

If environment model is know we can exploit structure in state - action space and apply supervised learning techniques to learn state values/policies to:

- Get good initial guesses for State Values and Policy
- Approximate solution for larger state - action space