DRL Course Домашнее задание 4

Алексей Матушевский

> Это прекрасное явление - Model Free RL, в нем нет моделей, но есть цели! А что еще нужно настоящему пирату? Он не знает какие невзгоды ждут его впереди, какой новый остров появиться на горизонте и сколько золота он добудет, у него есть только ветер и верная команда. Наш герой должен двигаться вперед, потому что конец ждет каждого из путешественников, а мы, наблюдатели, надеемся на хорошую историю.

В этой домашней работе мы изучаем работу Model Free RL (MF). В отличии от Model-base, мы не можем знать как среда отреагирует на наши действия. Так как среда стахостича, существует множество исходов с некоторой вероятностью. MF обладает преимуществом в случаях когда сложно составить достаточно точную модель среды и являются явными примером использования фундаментального метода «проб и ошибок». За счет доступности к программной реализации среды, мы может проводить огромное число экспериментов (проб) и находить оптимальные политики для максимизации средней награды (учесть ошибки).

Задание 1:

Реализовать Q-Learning и сравнить его результаты с реализованными ранее алгоритмами: Cross-Entropy, Monte Carlo, SARSA в задаче Taxi-v3. Для сравнения как минимум нужно использовать графики обучения.

Для решение этого задания, с учетом результатов домашнего задания # 1, я использовал Cross-Entropy с Policy-Smoothing:

> *При сравнении Laplase Smoothing и Policy Smoothing – Policy Smoothing находит стратегии лучше при том-же количестве эпизодов при тех же аргументах Quant и схожих лямбда, хотя параметр лямбда сравнивать не корректно, т.к. они разные, у LS — он не ограничен, а у PS имеет ограничение (0, 1] .*

Lambda	Quantile	Trajectory_n	Episodes	MaxRewad	TopEpisode
0.5	0.2	640	100	4.729687	66

Параметры рассматриваемых алгоритмов:

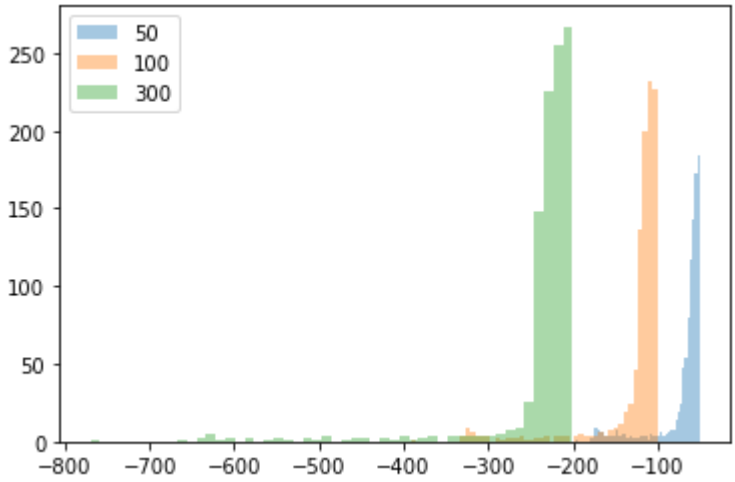
Cross Entropy	Monte-Carlo	SARSA	Q-Learning
Lambda Quintilian Trajectory_len Episodes	Epsilon Gamma K(Trajectory_len)	Epsilon Gamma K(Trajectory_len) Alpha	Epsilon Gamma K(Trajectory_len) Alpha

Epsilon - Вероятность выбрать действие с максимальной Q(s,a)
Gamma - коэффициент дисконтирования, задает значимость будущей награды.
K - количество эпизодов обучения
Alpha - (Learning Rate) определяет на сколько новая информация перекрывает старую инфомацию. В детерминистичной среде alpha=1 оптимальна. В нашем случае оптимальное значение нужно найти.

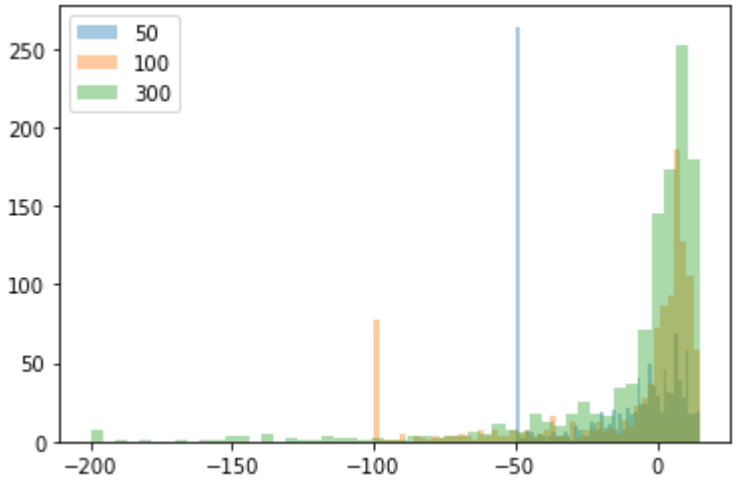
Проведено 6300 экспериментов:

```
Espilon        0.001, 0.11189, 0.22278, 0.33367, 0.44456, 0.55544, 0.66633,
0.77722, 0.88811, 0.999
gamma          0.001, 0.11189, 0.22278, 0.33367, 0.44456, 0.55544, 0.66633,
0.77722, 0.88811, 0.999
Epsodes_n      1000
Trajectory len 50, 100, 300
```

Средняя оценка в зависимости от максимальной длинны траектории. Чем длиннее траектория тем больше минусов может набрать таксист.



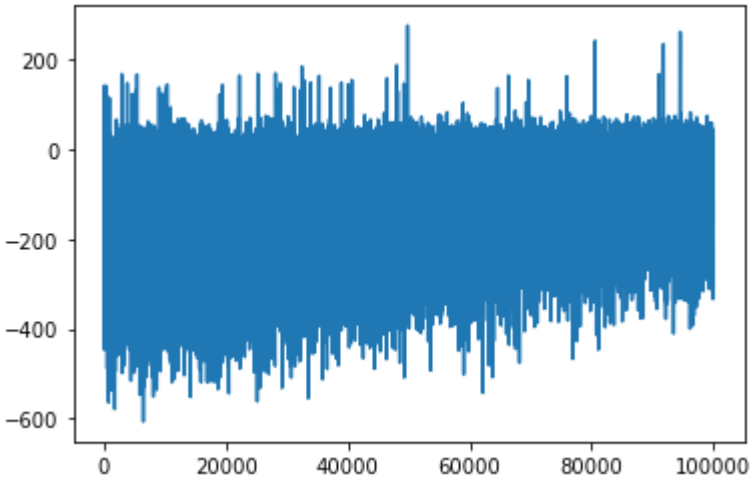
Распределение максимальных значений в зависимости от длинны траекторий.

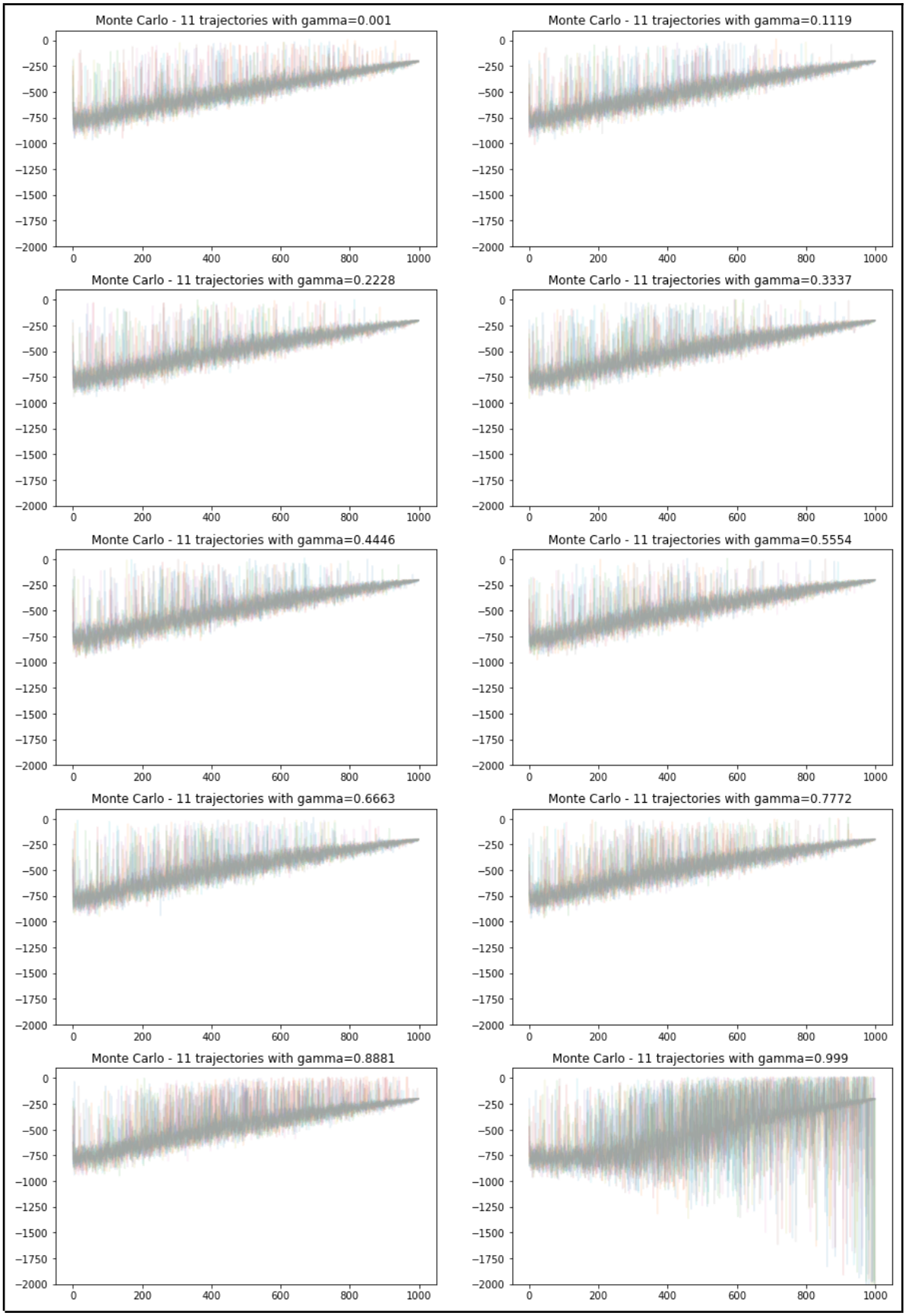


Видно что в более длинных траекториях больше максимальных значений

Во всех алгоритмах уменьшение Epsilon идет равномерно в интервале [1 0) , на каждом шаге мы устанавливаем epsilon=1/k, где k это номер шага. При таком подходе, вне зависимости он длинны траектории, алгоритм м междутакой подход изменения Epsilon приводит к тому что вне за, вне зависимости от длинны траектории

С текущей стратегией для епсилон — найти рабочий вариант не получается

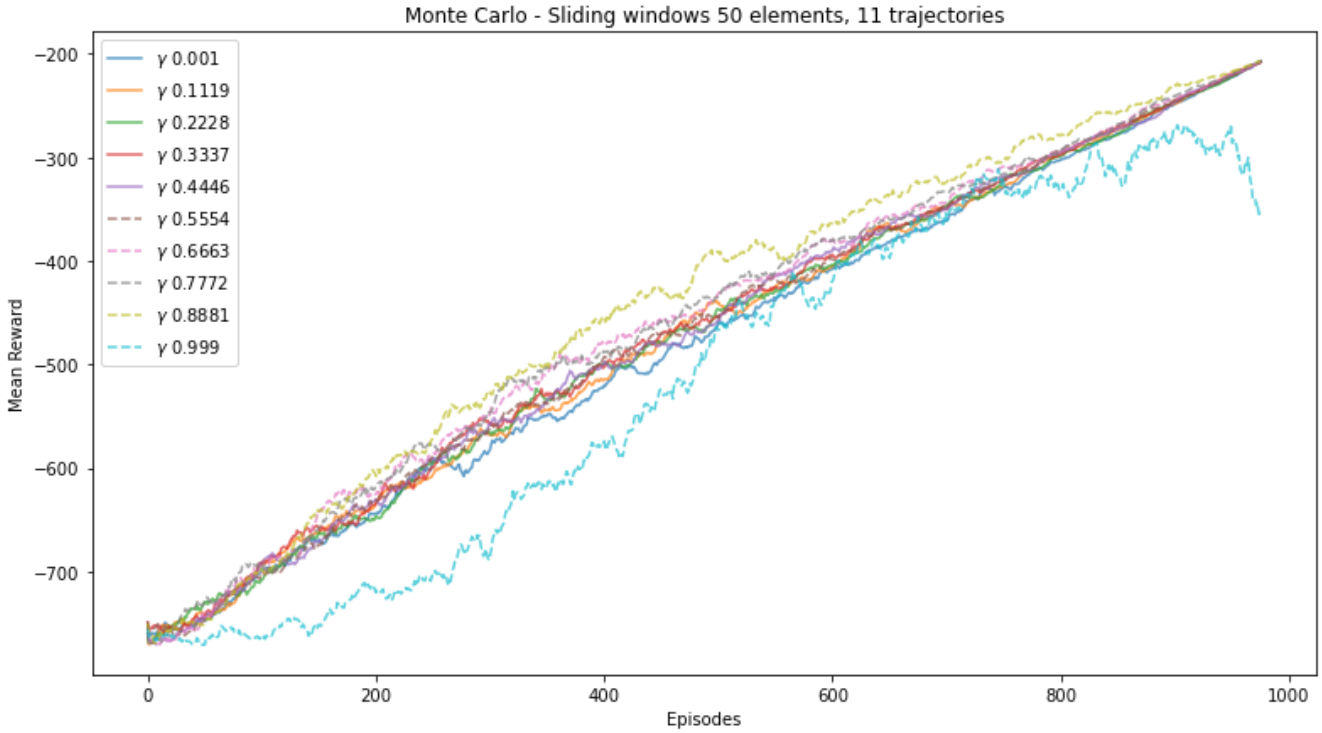




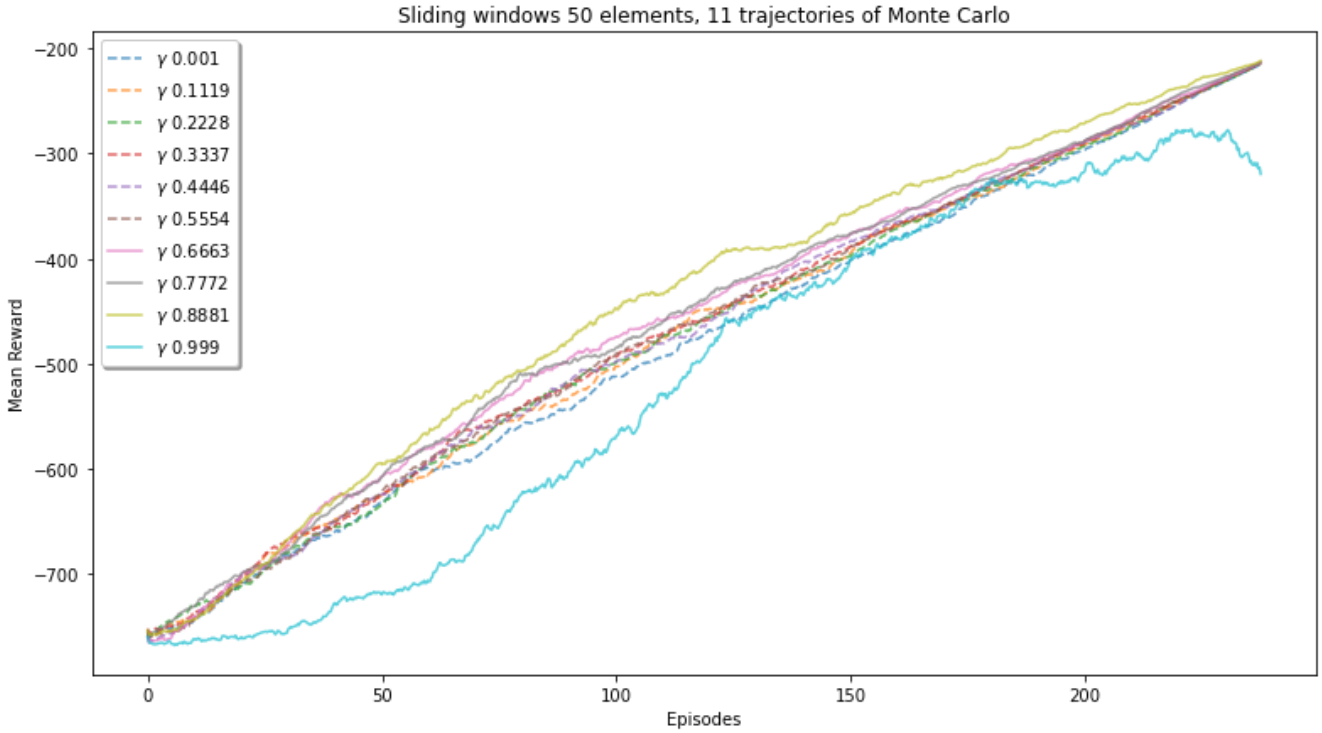
МС при разных гаммах

Монте-Карлго при разных гамма в принципе сходится одинаково. Только при высоких гамма дисперсия растет

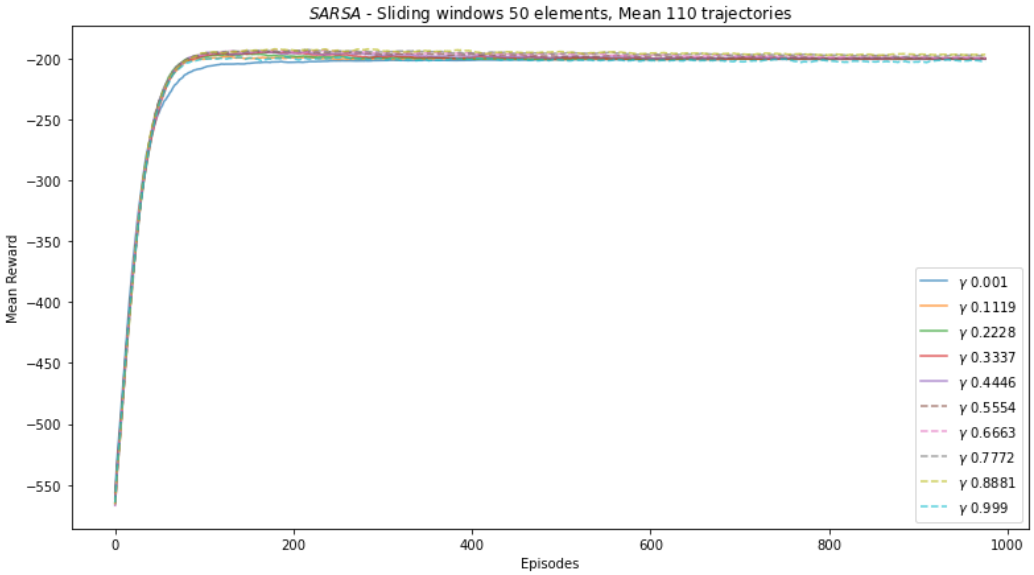
Возьмем траектории с одинаковой гамма, усредним каждый для каждого эпизода посчитаем среднее значение награды.

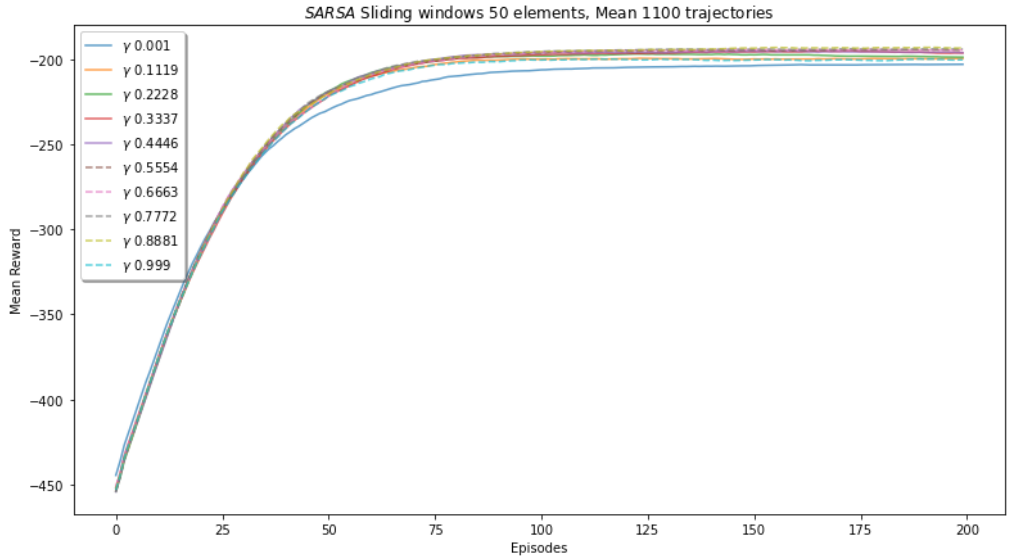


Мы видим видим что в последних эпизодах все сводится к одному числу ~-200 и только при высокой гамме 0.999, значение значительно меньше других

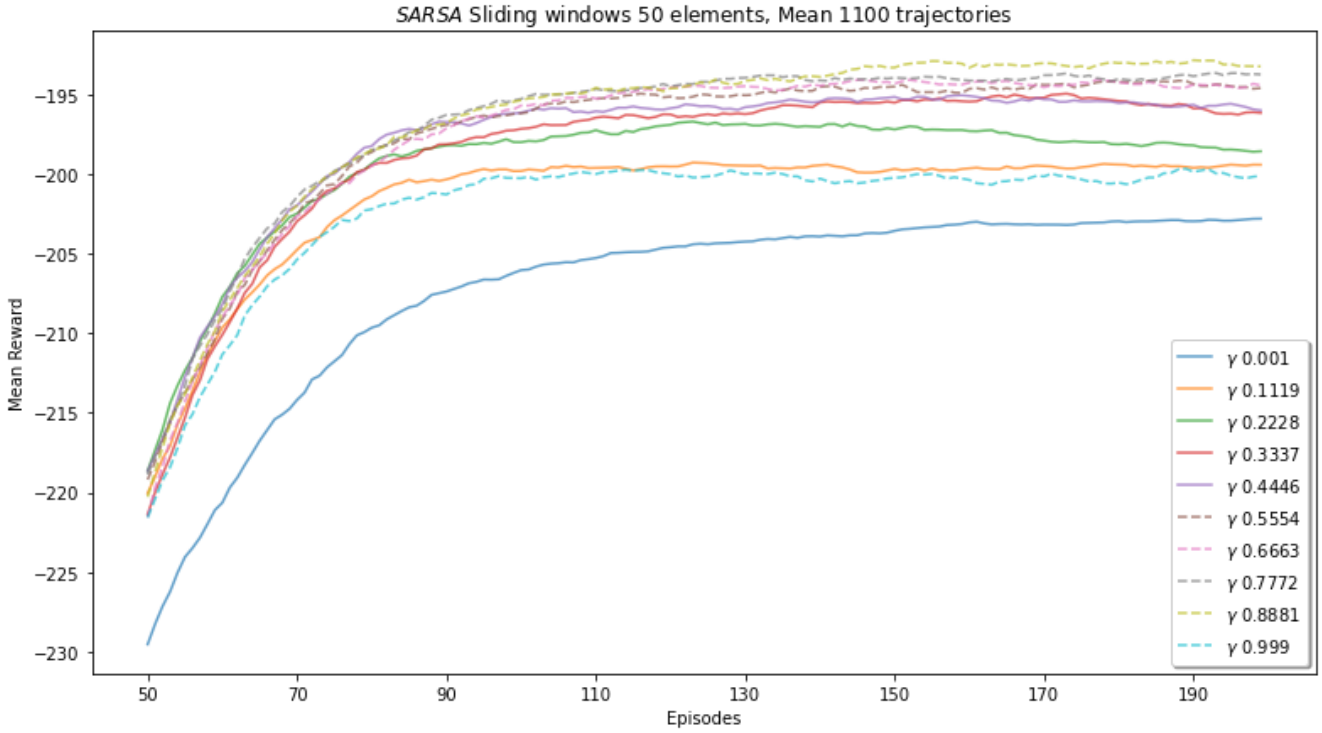


Алгоритм SARSA показывает намного более быструю сходимость, но все равно низкую среднюю оценку

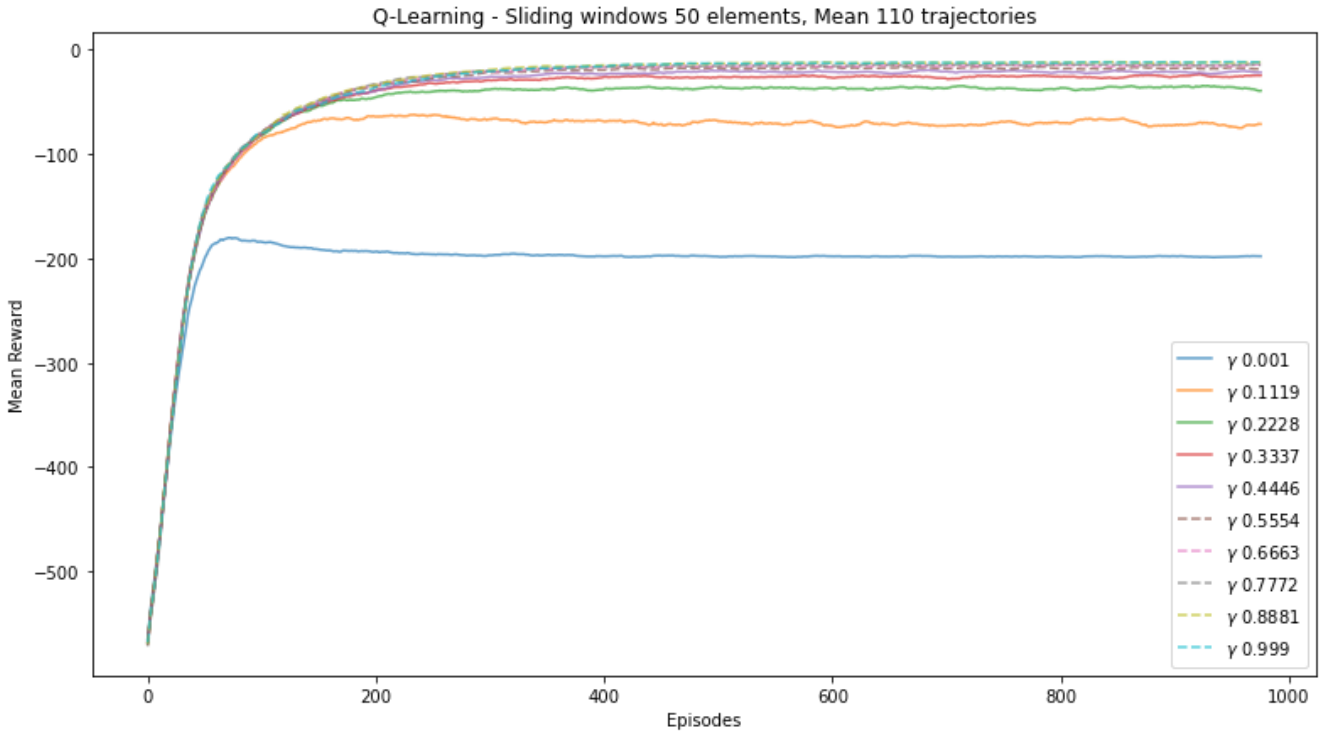




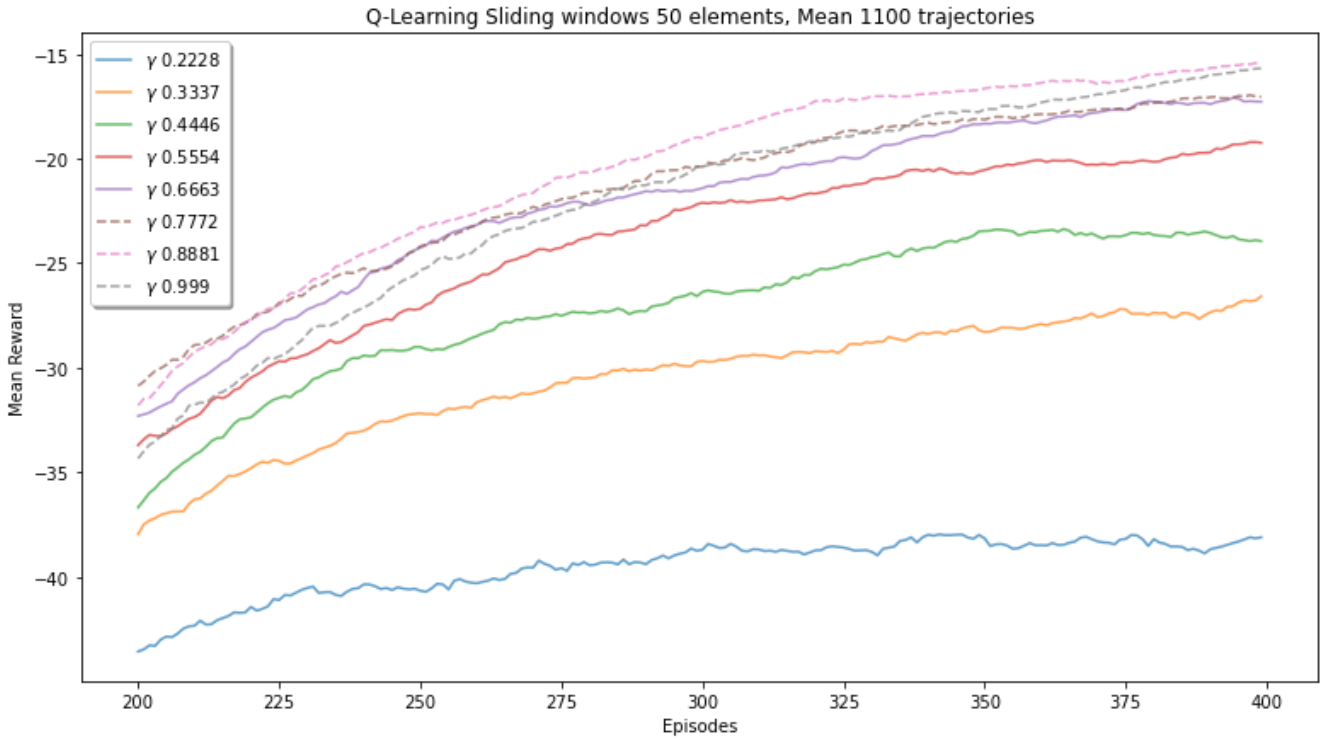
Если взглянуть на средние с 50 эпизода по 200ый. Видно что крайние значения 0.999 и 0.001 значительно отличаются.



Алгоритм Q-Learning дает значительно лучшие результаты в средних оценках, но более чувствителен к значению гамма



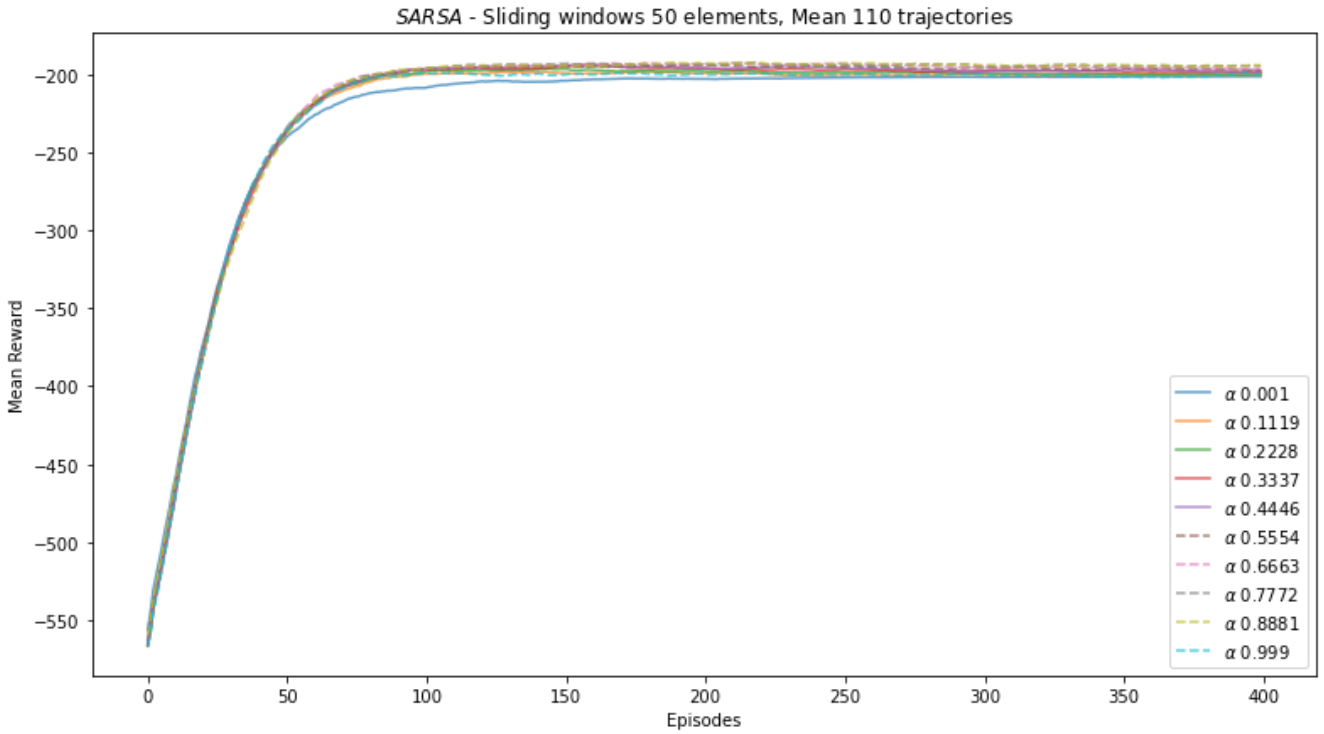
Как и SARSA, Q-Leraning сходится значительно быстрее чем MC.



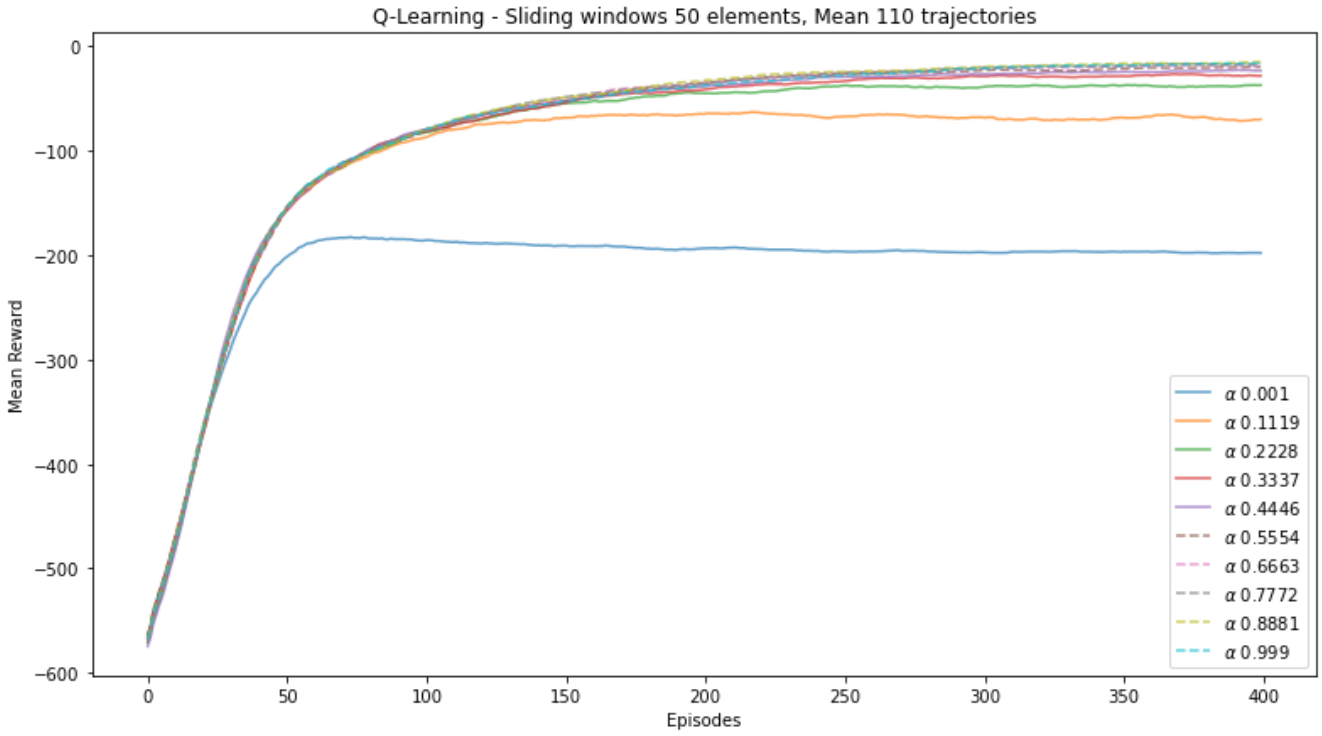
Более высокие значения гамма дают более высокую среднюю оценку

Параметр Альфа.

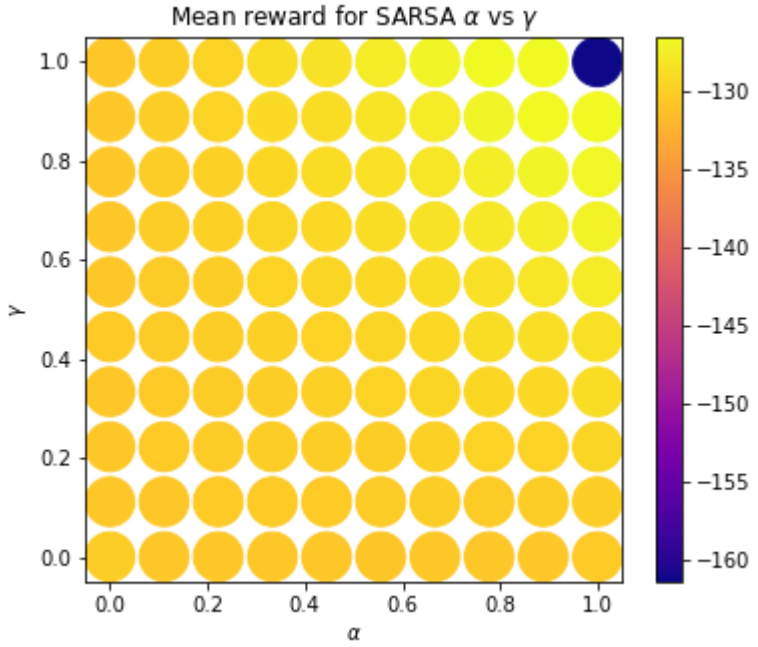
В алгоритма SARSA параметр влияет на скорость сходимости и максимальной средней награды



В алгоритме Q-Learning — влияние альфа также как и гаммы более высокие значения дают более высокое значение средней оценки



2D график средней оценки для SARSA альфа и гамма. Оптимальные значение в области 0.8 для обоих параметров



2D график средней оценки для Q-Leaning альфа с гамма. Оптимальные значение в области 0.8 для обоих параметров

Задание 2

Дискретизировать (можно использовать numpy.round()) пространство состояний и обучить Агента решать CartPole-v1, Acrobot-v1, MountainCar-v0, или LunarLander-v2 (одну на выбор) методами Monte Carlo, SARSA и Q-Learning. Сравнить результаты этих алгоритмов и реализованного ранее алгоритма Deep Cross-Entropy на графиках.

В пространстве LunarLender2 среда состоит их 8ми измерений

- координаты X, Y каждая в пределе [3, 3]
- линейной скорости по X, Y каждая в пределе [3, 3]
- угол наклона корабля [-3.14, 3.14]
- угловая скорость [-5, 5]
- дискретные значения — есть ли касания «ног» земли {0,1}

Чтобы дискретизировать эти параметры — мы можем умножить их на M (множитель) и округлить.

Проблема такого подхода состоит в том что с округлением — мы изначально не знаем все возможные состояния (но мы конечно можем их подсчитать). Наши алгоритмы расчитывают использовать матрицы в своих основах. Чтобы хранить все комбинации состояний среды Lunar Lender - нам не хватит оперативной памяти.
6*M*6*M*6.14*M*10*M*4=8640*M^4

Поэтому можно хранить эти значения в словаре.

- Из округленных значений состояния среды можно «составить ключ» состояния — объединив все значения в одну строку через разделитель (не цифру).
- Полученный ключ будет использоваться в словаре, для определения состояний действий. Ключ будет хранить одно состояние и любое изменения в параметра среды, будет создавать новый ключ и таким образом отличать их друг от друга.

Все наши алгоритмы «рассчитывают» на то, что мы будет посещать одни и тоже состояния хотя-бы один раз, чтобы накопить статистику по ним, и определить Эпсилон-Жадным способом — оптимальное действие.

Так как количество возможных состояний хоть и меньше бесконечности, но все же невозможно много. Поиск оптимального Множителя — займет время.

Задание 3

Придумать стратегию для выбора epsilon позволяющую агенту наилучшим образом решать Taxi-v3 алгоритмом Monte Carlo.

Литература:
1. Лекция №4 Атнона Плаксина
1. Sutton, Richard S.; Barto, Andrew G. (November 13, 2018).
http://incompleteideas.net/book/bookdraft2018mar21.pdf
2. https://en.wikipedia.org/wiki/Q-learning#Discount_factor
3. https://www.baeldung.com/cs/epsilon-greedy-q-learning