DRL Course Домашнее задание 1

Алексей Матушевский

В этой практической работы мы изучаем раобту алгоритма Cross Entropy Method (CEM) и его улучшения. Изучение работы алгоритма происходит в среде Taxi. В ней мы управляем машинкой для подвоза клиента с одно из четырех пунктов назначения в другие из оставшихся трех. Всего у нас 4 места где может появиться клиент и где может находит пункт назначение.

С использованием Алгоритма CEM мы пытаемся создать стратегии (policy) поведения таксиста, чтобы тот мог принимать оптимальные решения об перемещении в дискретной среде на основании стратегии, для того, чтобы как можно меньше потратить топлива и при этом довести клиента до пункта назначения, без того чтобы “заглохнуть” по пути. После 200 перемещений по дискретной среде – выполнение останавливается. За каждый шаг водитель тратить топливо или получает штрафы. В случае когда водитель не делает лишних шагов и довозит клиента до путка назначения – нагарада равна 1. или меньше во всех других случаях.

Алгоритм CEM включает в себя два шага:

- Policy Evaluation – применения стратегии водителем/водителями

- Policy Improvement – обновления стратегий

В шаге Policy Evaluation – могут возникать такие ситуации когда улучшения политики не приводит к улучшение, или наоборот ухудшает результаты множества попыток. К примеру – вероятности каких либо решений становятся 0 и последующие шаги просто не возможны.

Чтобы избежать “застревания” в подобных ситуациях – вводятся улучшения в процесс обновления стратегий:

- Laplace smoothing – исключает возможность вероятности какого либо решения стать 0

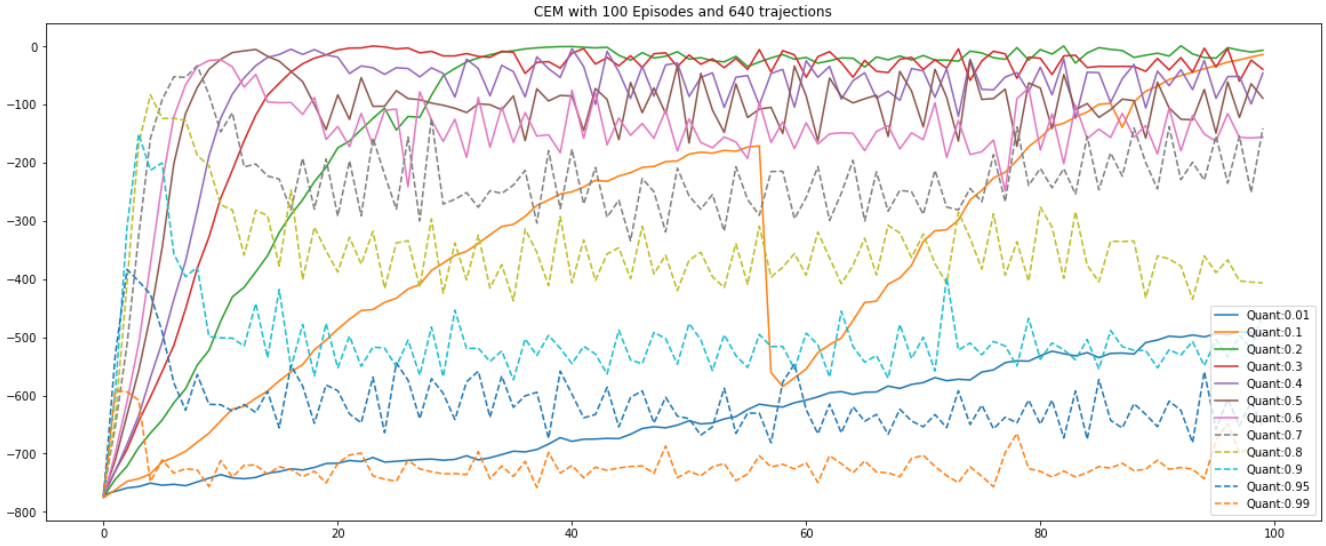
- Policy smoothing – смешивает старую и новую стратегии в заданной пропорции

Оба эти улучшения обучение для получения максимальной средней раграды.

Cross Entropy Method

Параметр Quantile. Этот параметр влияет на количество “элитных” стратегий будет участвовать в улучшении. Как мы видим из графика ниже – повышение параметра q с 0.1 до диапазона 0.2-0.4 – даёт более стабильное улучшение в конечных наградах с каждым новым эпизодом обучения. Но в диапазоне начиная с 0.5 улучшение не достигает таких же значений и быстро высот при значении 0.1-0.4 и негативно сказывается на лучшениях конечно метрики — средней

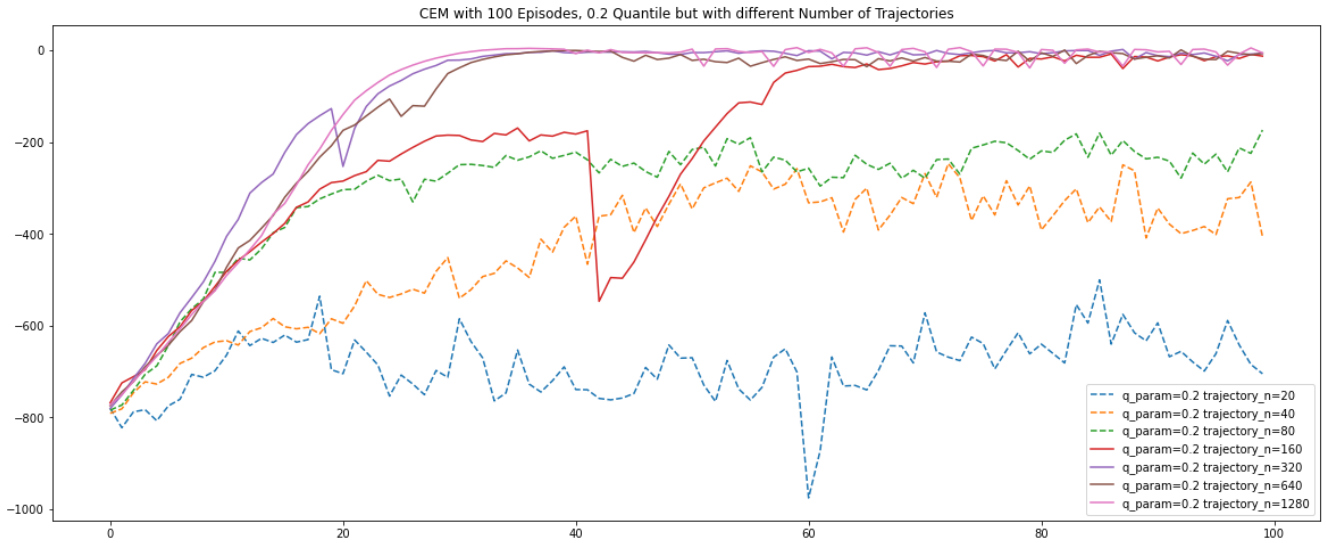
награды. Крайнее значение 0.9 и выше – не сильно отличимо от изначальной точки награды в первом эпизоде обучения. (чем объяснить падение метрики)?



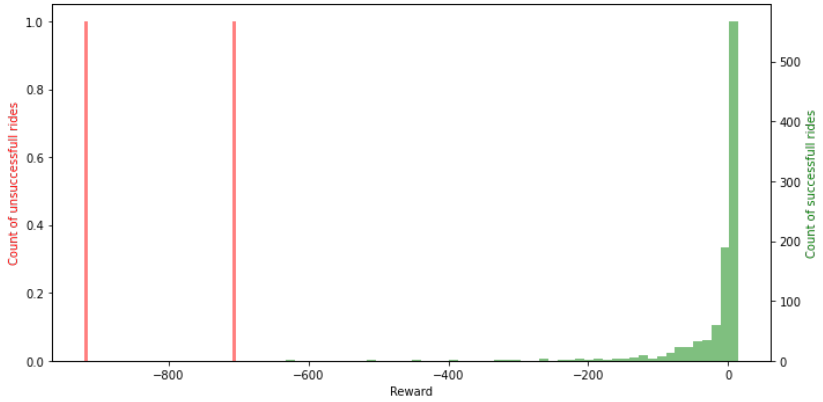
Параметр Number Of Trajectories влияет на количество траекторий за один эпизод. При одном и том же Quantile, увеличение параметра Trajectories_N приводит к улучшению конечной метрикик - средней награды. Но при этом значительно увеличивается время работы алгоритма. Но возможны случаи когда «сильно не повезет» (trajectory_n = 160) и получится такая политики что резко ухудшат серднюю награду.

Оптимальные параметры можно подобрать исходя и графике выше.

```
episode_n      = 50
trajectory_len = 200
trajectory_n   = 320
q_param        = 0.2
```



Примеры % закончившихся успешно поездок для 1000 симуляцией



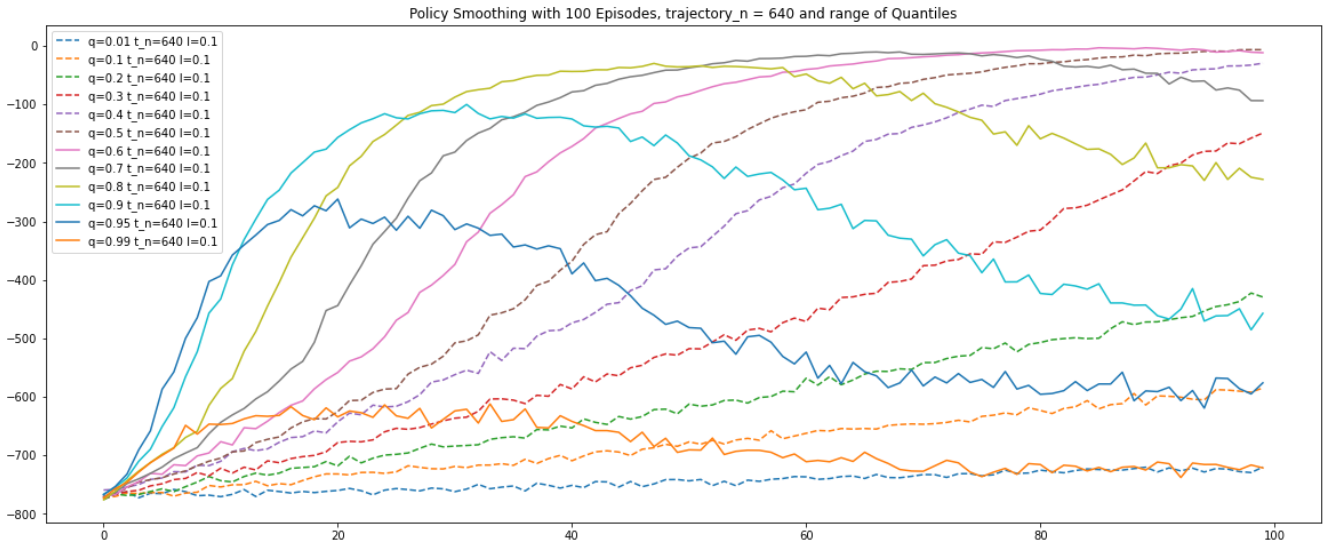
Policy Smoothing

Идея алгоритма состоит в том чтобы “смешивать” включать в новую % от новой и % от старой стратегий. % регулируется параметром лямбда.

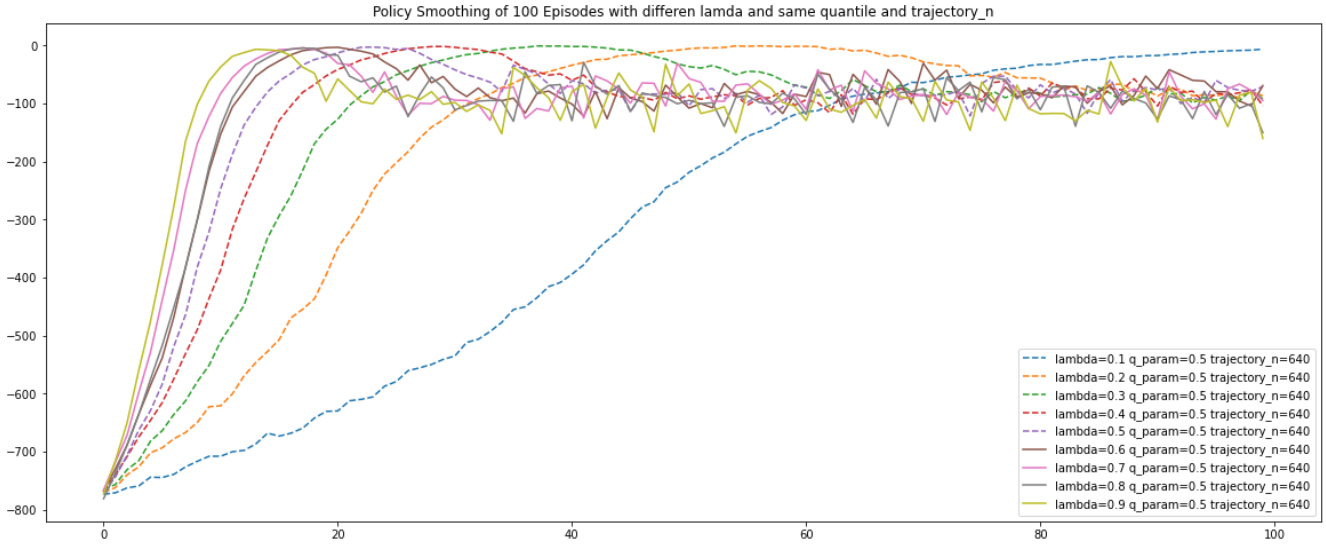
- Policy smoothing

$$\pi_{n+1}(a|s) \leftarrow \lambda\pi_{n+1}(a|s) + (1-\lambda)\pi_n(a|s), \quad \lambda \in (0,1]$$

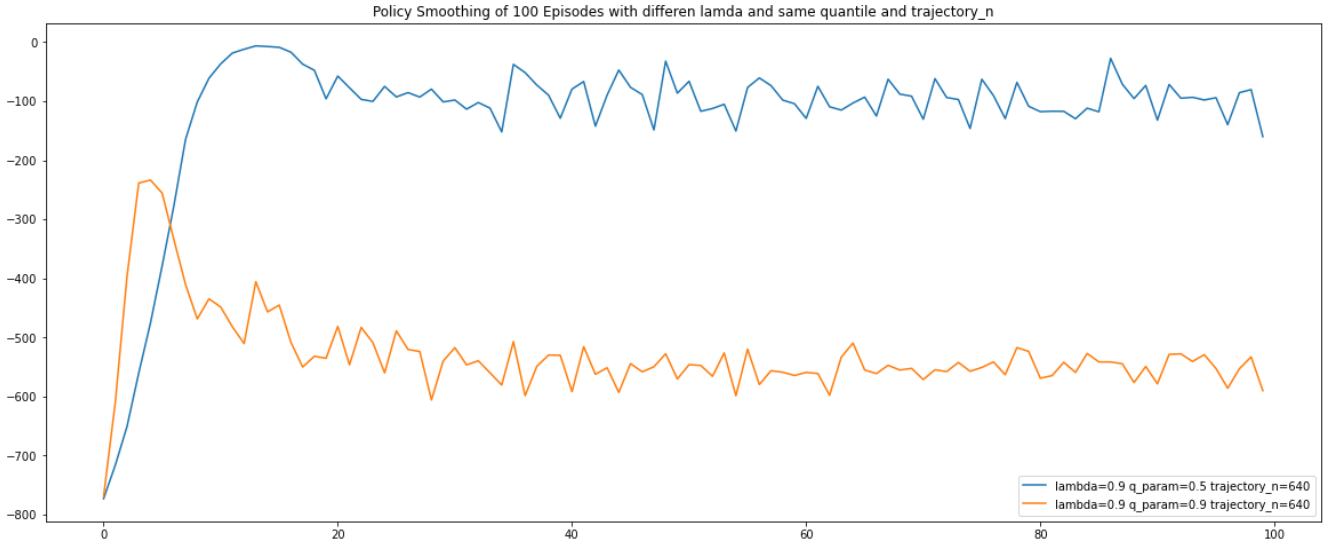
Также как и CEM, В Policy Smoothing ведет себя по разному в зависимости от параметра Quant. Чем меньше quant тем более стабильней рост в средних наградах с каждым новым эпизодом обучения. При значениях выше 0.7 алгоритм быстро уходит в увеличение ошибки.



Новый параметр лямбда, при тонкой настройке, позволяет быстрее найти стратегию с минимальной наградой но напрямую зависит от параметра q. К примеру при таблице ниже может создаться впечатление что параметры lambda=0.9 ялвяется оптимальным.

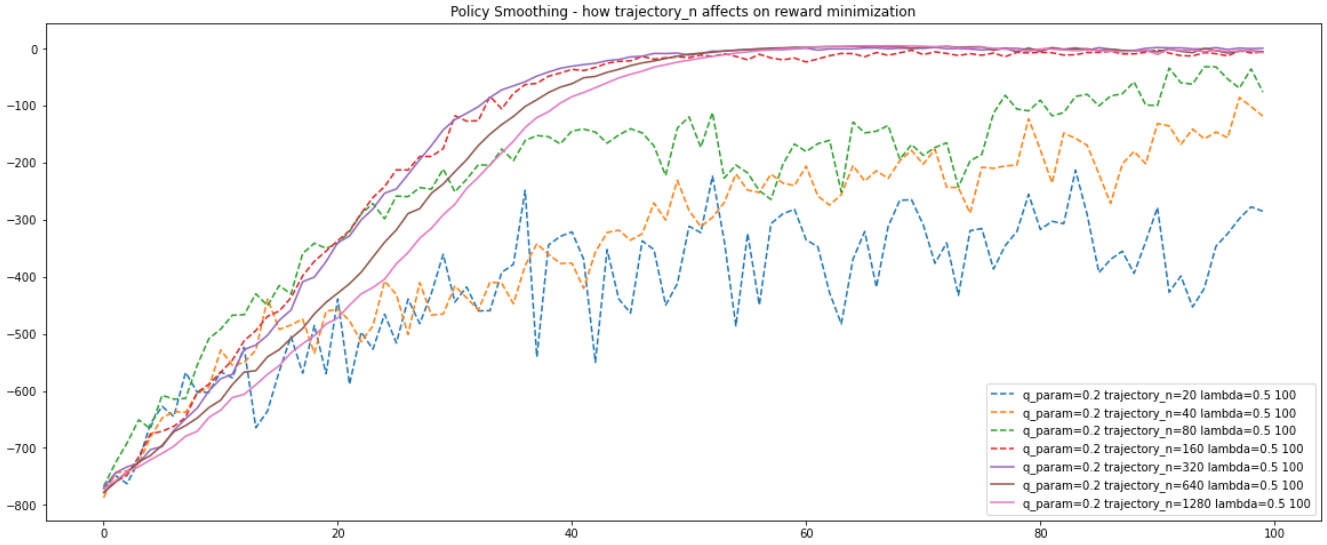


Но при изменения параметра q_param результат сильно меняется



Влияние trajectory_n - б*о*льшие значение позволяют получить более высокую среднюю награду, при прочих равных q_param и lamda. Но того же результата можно добиться и правильным подбором параметра q_param (оранжевая линия). На Графике ниже, данные отсортированы по максимальному значению их всех эпизодов. На первом места алгоритм с trajectory_n=640 опережаятrajectory_n=5120 и trajectory_n=2560, что значительно экономит ресурсы, даже с учетом того что было пройдено больше эпизодов.

Количество траекторий по разному влияет на алгоритм. При низких Высокие значения trajectory_n не приводят к более быстрой минимизации награды, как и очень низкие значения.

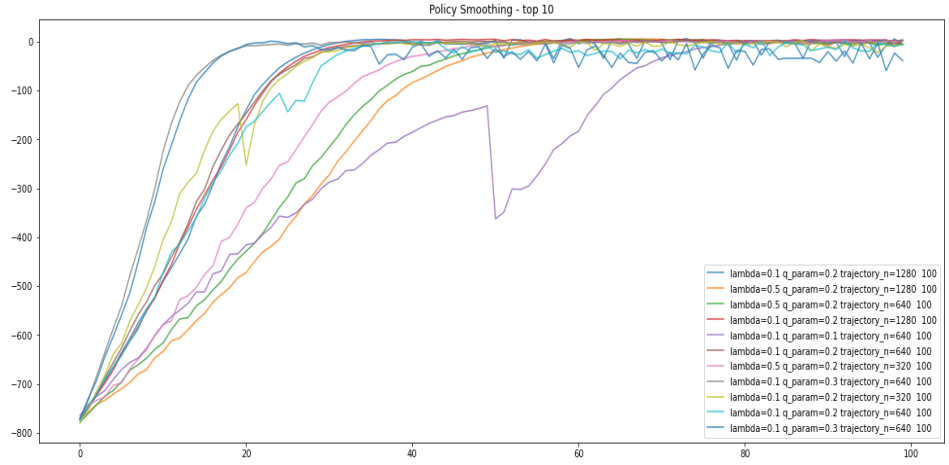


В таблице ниже представленны топ 10 настроек гипер параметров Plolicy Smooting. Начиная с 10 места, можно создать водителя с почти 100% результативностью довоза пассажиров до пункта назначения. Очевидно что большее количество траекторий дает более гарантированный конечный результат. Из всех настроек, место 3 кажется более приемлемым. Так как дает заведамо стабильный результата и не требует стольких многих вычислений как первые 2 места в рэйтинге.

	Lambda	Quantile	Trajectory_n	Episodes	MaxRewad	TopEpisode
1	0.1	0.2	1280	100	6.211719	73

	Lambda	Quantile	Trajectory_n	Episodes	MaxRewad	TopEpisode
2	0.5	0.2	1280	100	5.332031	67
3	0.5	0.2	640	100	4.729687	66
4	0.1	0.2	1280	100	4.636719	72
5	0.1	0.1	640	100	3.732813	99
6	0.1	0.2	640	100	2.429688	99
7	0.5	0.2	320	100	2.409375	71
8	0.1	0.3	640	100	2.325000	99
9	0.1	0.2	320	100	2.178125	87
10	0.1	0.2	640	100	1.314062	92

График победителей Policy Smoothing



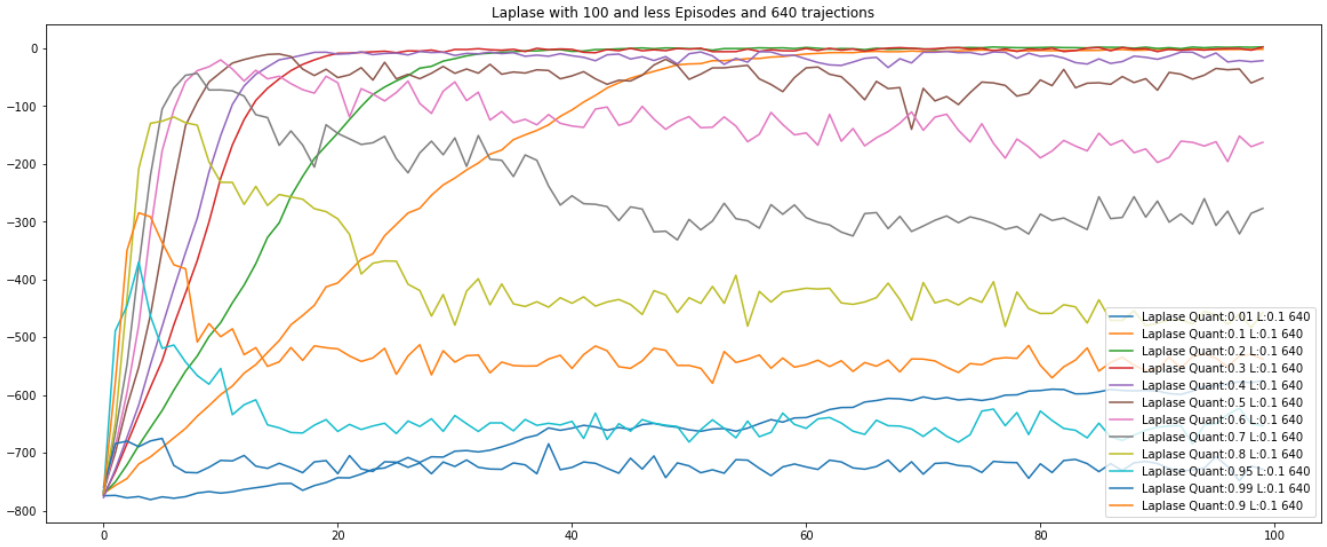
Laplase smoothing

Идея улучшения Laplase Smoothing состоит в том чтобы не допускать появления нулевой вероятности. Параметр Lamda используется в процессе создания новой стратегии.

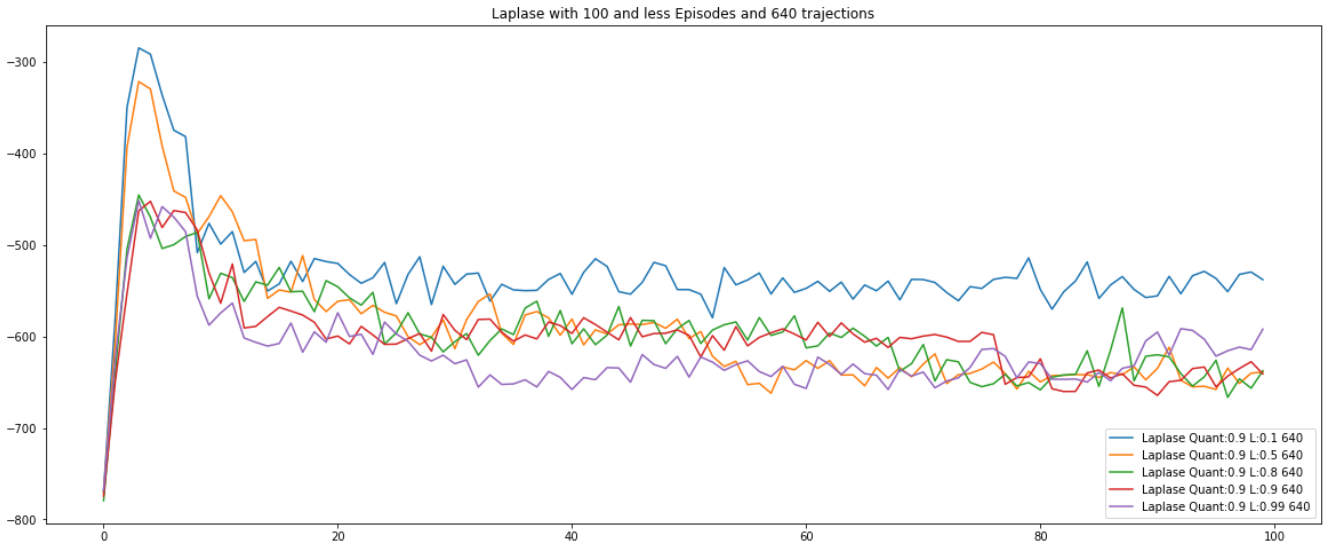
- Laplace smoothing

$$\pi_{n+1}(a|s) = \frac{|(a|s) \in \mathcal{T}_n| + \lambda}{|s \in \mathcal{T}_n| + \lambda|\mathcal{A}|}, \quad \lambda > 0$$

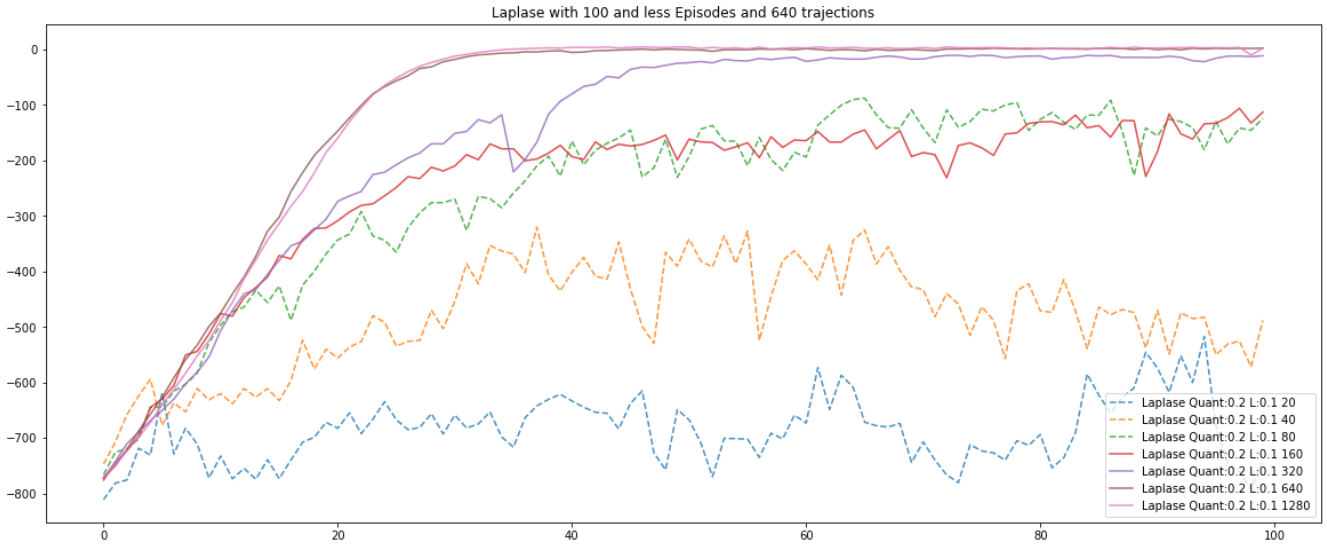
Влиянее параметр Quant и имеет схожий эффект что и в предыдущих примерах. С ростом Quant уменьшается максимальная средняя награда.



Ввиду того что Laplase Smoothing по свой сути не исключает нулевые вероятности – это дает возможно большему количеству не эффективных стратегий учитываться при создании новых стратегий. С ростом lambda — уменьшается максимальная средняя награда с каждым новым эпизодом, после предоления максимума. При прочих равных — низкое значение лябма дает лучший результат.

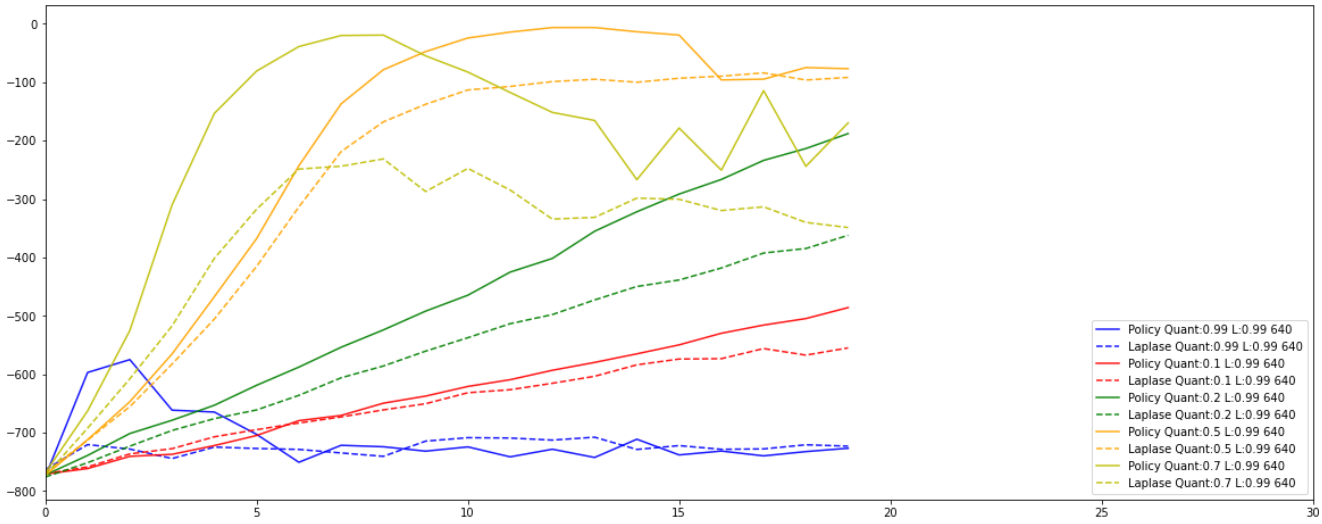


Увеличение количества траекторий в эпизоде позволяет за меньшее количество эпизодов получать наибольший результат



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] .

(пунктиром Laplase, сплошная Policy Smoothing).



Решение проблемы стохастичности.

Алгоритм CEM зависит от конечности состояний среды. В случае когда среда сотхастична, алгоритму CEM может не хватить ресурсов из-за больших размеров среды. Чтобы решить эту проблему мы можем обратиться к детерминированным стратегиям.

Посредства «сэмплирования» мы можем создавать множество дет. стратегий и оценивать их средние награды.

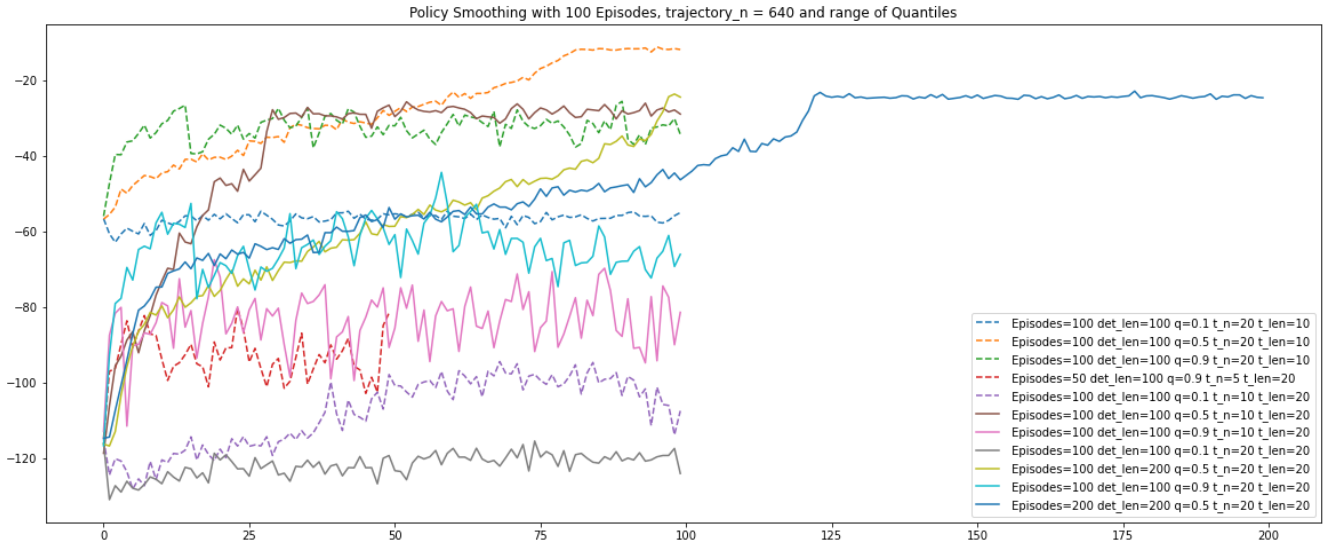
Из наблюдения за работой алгоритма можно сказать что

- количество таксистов определяет размер начальной ошибки, чем их большое - тем больше ошибка.

- параметр — t_len — имеет пороговое значение, после которого таксисту не хватает ходов для достижение цели среды.

- параметр лямбда показывает лучший результат в средних значениях как и в предыдущих алгоритмах.

При высоких значениях t_n и t_len (схожих со значениями предыдущих алгоритмов) — на порядки увеличивает время работы алгоритма.



det_len — количество детерминированных стратегий

t_n — количество таксистов за эпизод.

t_len — максимальная длинна поездки одного таксиста.

Q — quantile

При этом ни одна из комбинаций не дала каких либо результатов.

В сравнении с предыдущими алгоритмами мне не удалось найти параметры которые бы привели к доставке пассажиров.

- Возможно проблемы в реализации

- возможно проблема в том что нужно еще исследовать простраснтво числе гиперпараметров

Выводы

Изучив работу алгоритма EMC и улучшениями, мы приходим к выводу что ECM и LS и PS может создать стратегии для водителя, следуя которым он сможет почти со 100% вероятностью завершить задачу успешно.

Не законченное исследование по решению стахатичности среды — не позволяет сделать каких либо выводов.

Спасибо за увлекательную домашнюю работу!