RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

FUNCTION APPROXIMATION [TUTORIAL]

Cartpole: from discretized (tabular) to continuous state

Debabrota Basu (Inria Lille)
Guillaume Pourcel (Inria Lille, University of Groningen)

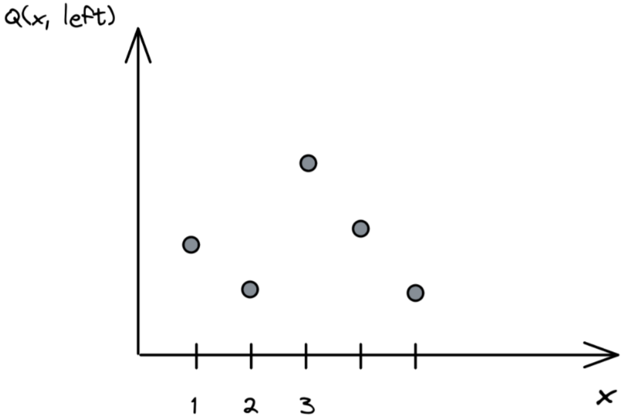
Summary

1. **Introduction to GYM**
   1. Continuous vs Discrete
2. **Exercises: implementing FQI in tabular mode (notebook 1a)**
   1. Recap FQI
   2. Different exercises
   3. Improving the discretization
3. **Continuous states function approximation (notebook 1b)**
   1. Linear Regression
   2. Improvement on Linear Regression
4. **Policy Gradient (notebook 2)**
   1. Just run the system (nothing to fill)
   2. Improvements (cf. slides: Natural Policy Gradient (NPG) covariance correction)

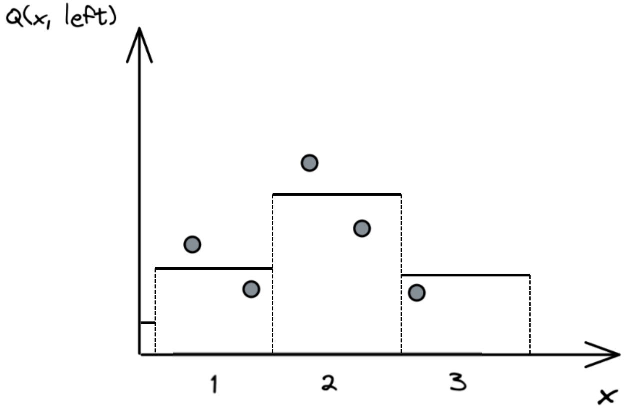
1. Introduction to GYM

Discrete vs Continuous

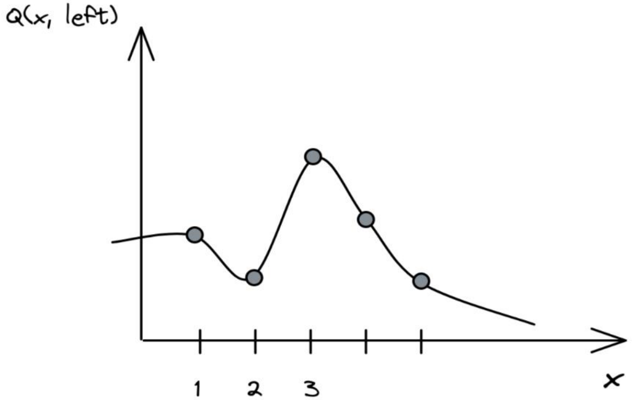
Tabular setting
(discrete state)



Discretized then tabular
(continuous state)

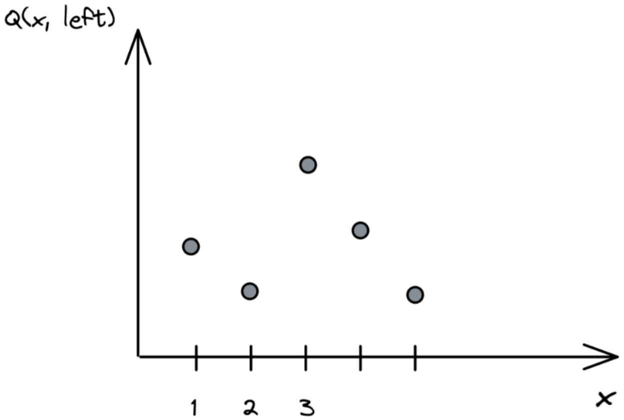


Function approximation
(continuous state)

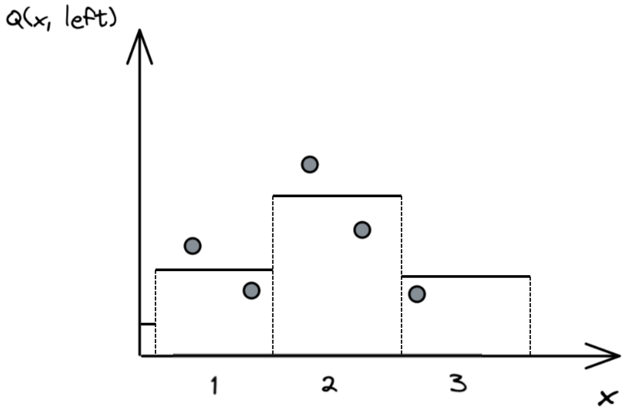


1. Introduction to GYM
Discrete vs Continuous

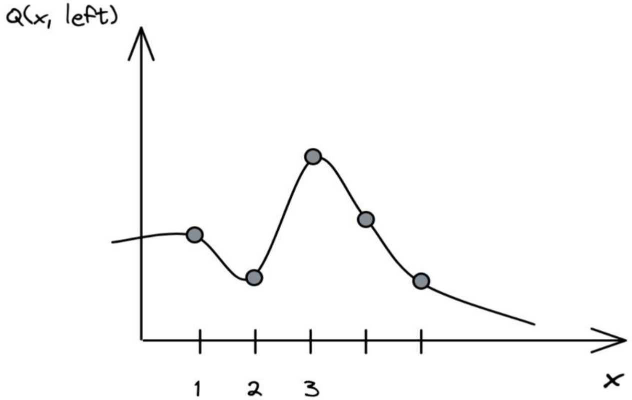
Tabular setting
(discrete state)

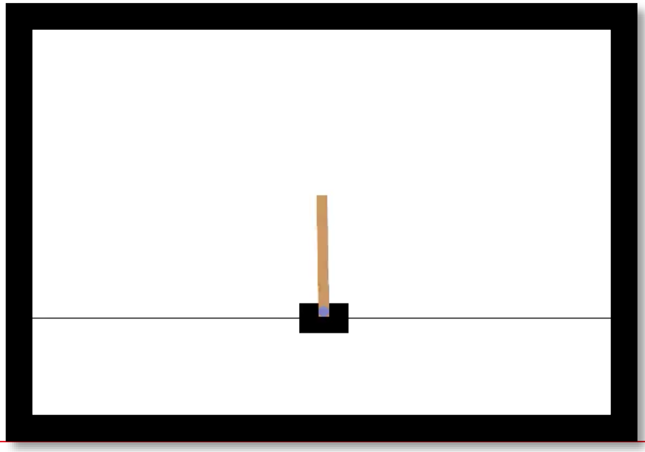


Discretized then tabular
(continuous state)



Function approximation
(continuous state)





2. Exercises: implementing FQI in tabular mode

Recap FQI: Fitted Q-iteration (Q-learning as regression)

$$Q_\theta(s_t, a_t) = r_t + \gamma \cdot \max_{a' \in A}(Q_\theta(s_{t+1}, a'))$$

$$f_\theta(x) = y$$

θ: parameters of the estimator

1. Create the training set based on the previous iteration $Q_\theta^{n-1}(s, a)$ and the transitions:
   - input: $x = (s_t, a_t)$
   - if s_{t+1} is **non** terminal: $y = r_t + \gamma \cdot \max_{a' \in A}(Q_\theta^{n-1}(s_{t+1}, a'))$
   - if s_{t+1} is terminal: $y = r_t$
2. Fit a model using a regression algorithm to obtain $Q_\theta^n(s, a)$

$$f_\theta(x) = y$$

3. Repeat, $n = n + 1$

2. Exercises: implementing FQI in tabular mode

Recap FQI: Fitted Q-iteration with tabular state and linear regression

$$Q_\theta(s_t, a_t) = r_t + \gamma \cdot \max_{a' \in A}(Q_\theta(s_{t+1}, a'))$$

$$f_\theta(x) = y$$

θ: parameters of the estimator

Liner Regression model:

$$f_\theta(x) = \theta x$$

Tabular state/action:

$$f_\theta(g(s,a)) = \theta \cdot g(s,a)$$

$$g(s,a) = \mathrm{OneHotEncoder(Discretized(s,a))}$$

2. Exercises: implementing FQI in tabular mode

Recap FQI: Fitted Q-iteration with tabular state and linear regression

$$Q_\theta(s_t, a_t) = r_t + \gamma \cdot \max_{a' \in A}(Q_\theta(s_{t+1}, a'))$$

$$f_\theta(x) = y$$

θ: parameters of the estimator

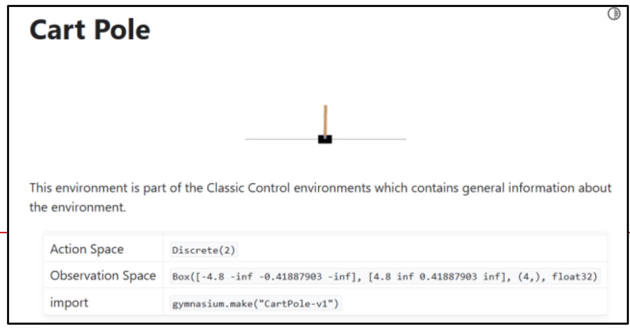
Liner Regression model:

$$f_\theta(x) = \theta x$$

Tabular state/action:

$$f_\theta(g(s,a)) = \theta \cdot g(s,a)$$

$$g(s,a) = \text{OneHotEncoder}(\text{Discretized}(\text{s}, \text{a}))$$



Cart Pole

This environment is part of the Classic Control environments which contains general information about the environment.

Action Space	`Discrete(2)`
Observation Space	`Box([-4.8 -inf -0.41887903 -inf], [4.8 inf 0.41887903 inf], (4,), float32)`
import	`gymnasium.make("CartPole-v1")`

2. Exercises: implementing FQI in tabular mode

Recap FQI: Fitted Q-iteration (Q-learning as regression)

Notebook plan (1.a, 1.b)

- **Introduction to Gym**
- **Exercise 1: Collecting data with random policy**
- **Exercise 2: Generate the targets for the FQI regression**
- **Exercise 3: Evaluate the greedy policy defined by the Q-values**
- **Exercise 4: Run FQI**

1.a: Discrete state (linear regression) (40min)

- Complete the Exercise
- Improve on the initial Discretization

1.b: Continuous state (linear regression) (30min)

- Adapt the code for continuous state
- Improve (better features, other models (KNN…))

2. Policy Gradient (20min)

- Run Reinforce
- Improve on Reinforce (for instance: Natural Gradient correction, slides 28 of the lecture)

Bonus: DQN (3., 4.)

2. Exercises: implementing FQI in tabular mode

Improving the discretization

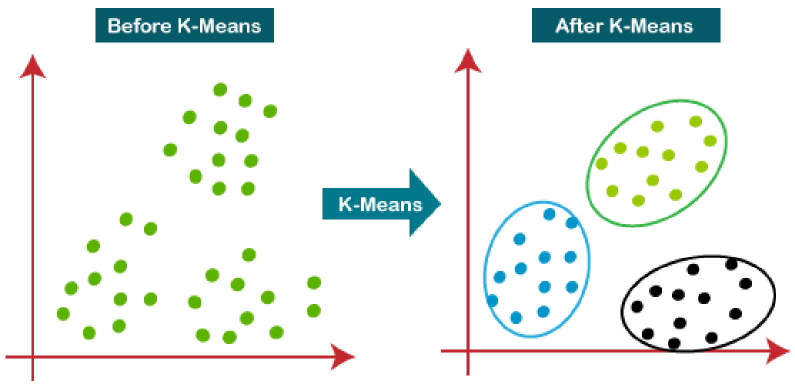
- **Too fine-grained:**
  - Weak generalization (need a lot of data)
  - High computational cost
- **Too coarse-grained:**
  - Weak expressivity: cannot represent fined-grained control policies

2. Exercises: implementing FQI in tabular mode

Improving the discretization

- **Too fine-grained:**
  - Weak generalization (need a lot of data)
  - High computational cost
- **Too coarse-grained:**
  - Weak expressivity: cannot represent fined-grained control policies

Automatic discretization



2. Exercises: implementing FQI in tabular mode

Improving the discretization

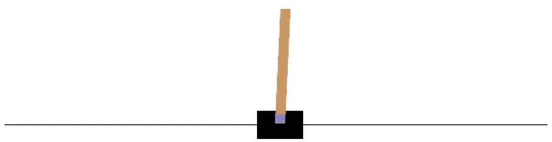
Automatic discretization

1. (optional?) have access to a trained policy (after training with KNN)
2. Collect states with the policy (+ noise, epsilon-greedy =0.1)
3. Use K-means (K=81) to cluster the observation (replace the grid-discretization) after normalization
4. FQI with the custom-linear regression

Thank's to Radji Waris for the good results

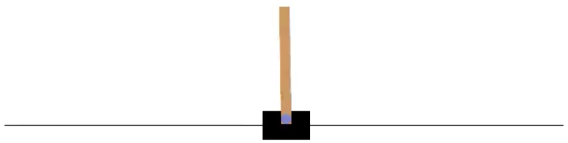
Discretization into 81 bins

```
Iter 2
Score: 0.33
()
Total reward = 11.10 +/- 1.37
Iter 4
Score: 0.47
()
Total reward = 10.90 +/- 1.04
Iter 6
Score: 0.57
()
Total reward = 14.00 +/- 8.37
Iter 8
Score: 0.63
()
Total reward = 10.70 +/- 1.00
Iter 10
Score: 0.67
()
Total reward = 10.50 +/- 1.20
```



K-means discretization (K-81)

```
Iter 2
Score: 0.19
()
Total reward = 404.00 +/- 100.37
Iter 4
Score: 0.34
()
Total reward = 322.90 +/- 164.06
Iter 6
Score: 0.43
()
Total reward = 354.20 +/- 164.05
Iter 8
Score: 0.49
()
Total reward = 489.10 +/- 32.70
Iter 10
Score: 0.52
()
Total reward = 353.80 +/- 168.29
```



2. Exercises: implementing FQI in tabular mode

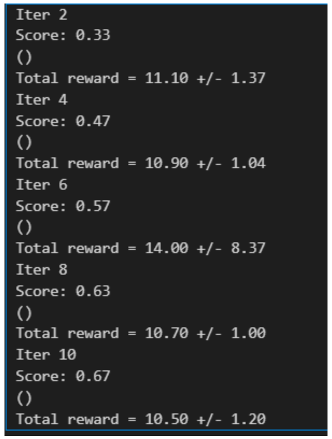
Improving the discretization

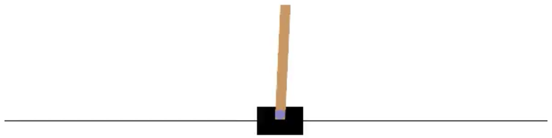
Automatic discretization

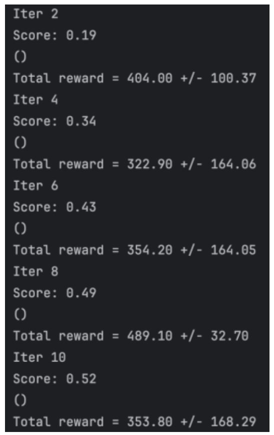
1. (optional?) have access to a trained policy (after training with KNN)
2. Collect states with the policy (+ noise, epsilon-greedy =0.1)
3. Use K-means (K=81) to cluster the observation (replace the grid-discretization) after normalization
4. FQI with the custom-linear regression

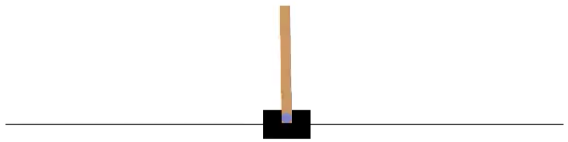
Discretization into 81 bins

K-means discretization (K-81)



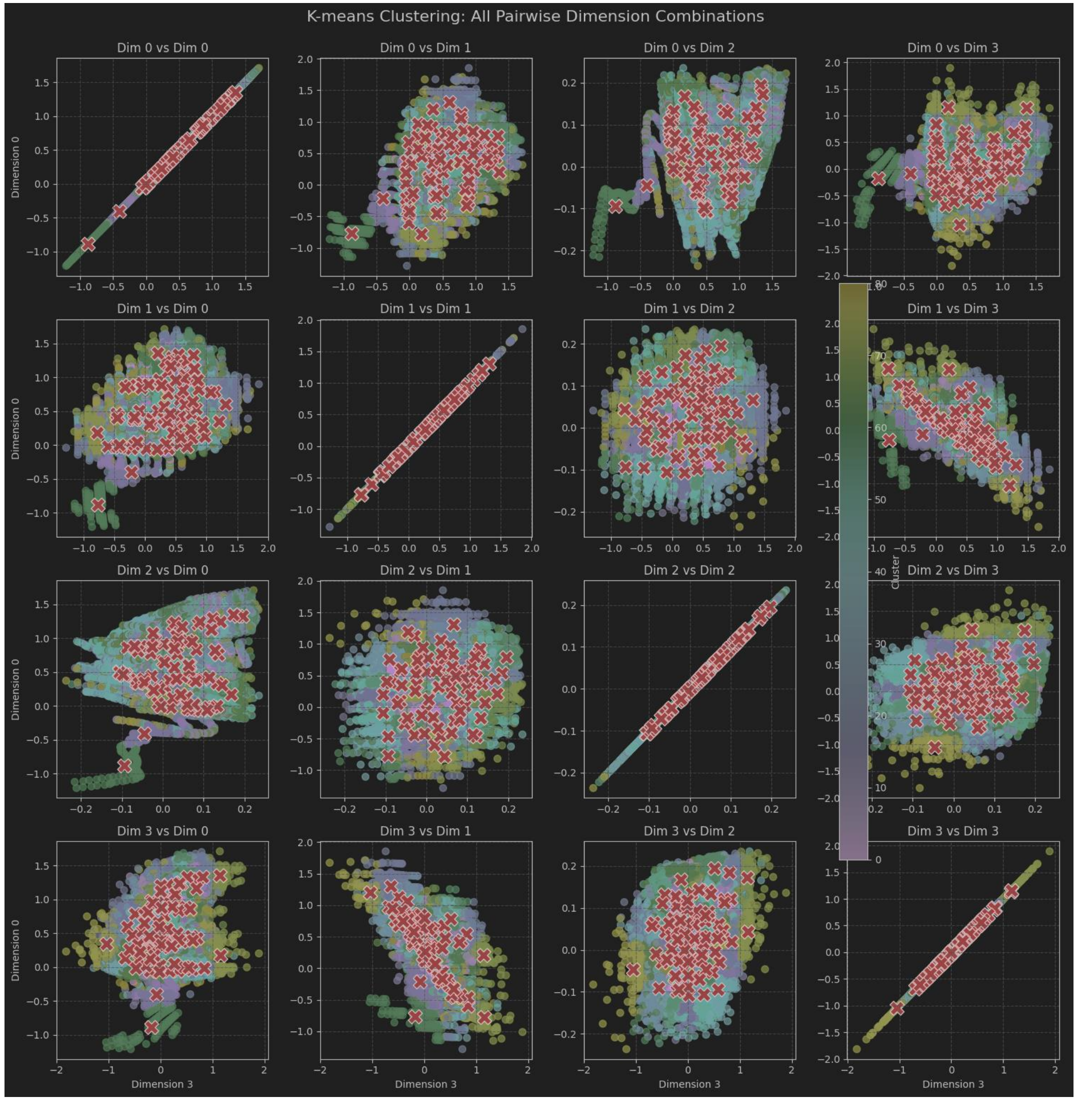






2. Exercises: implementing FQI in tabular mode

Improving the discretization



Thank you.