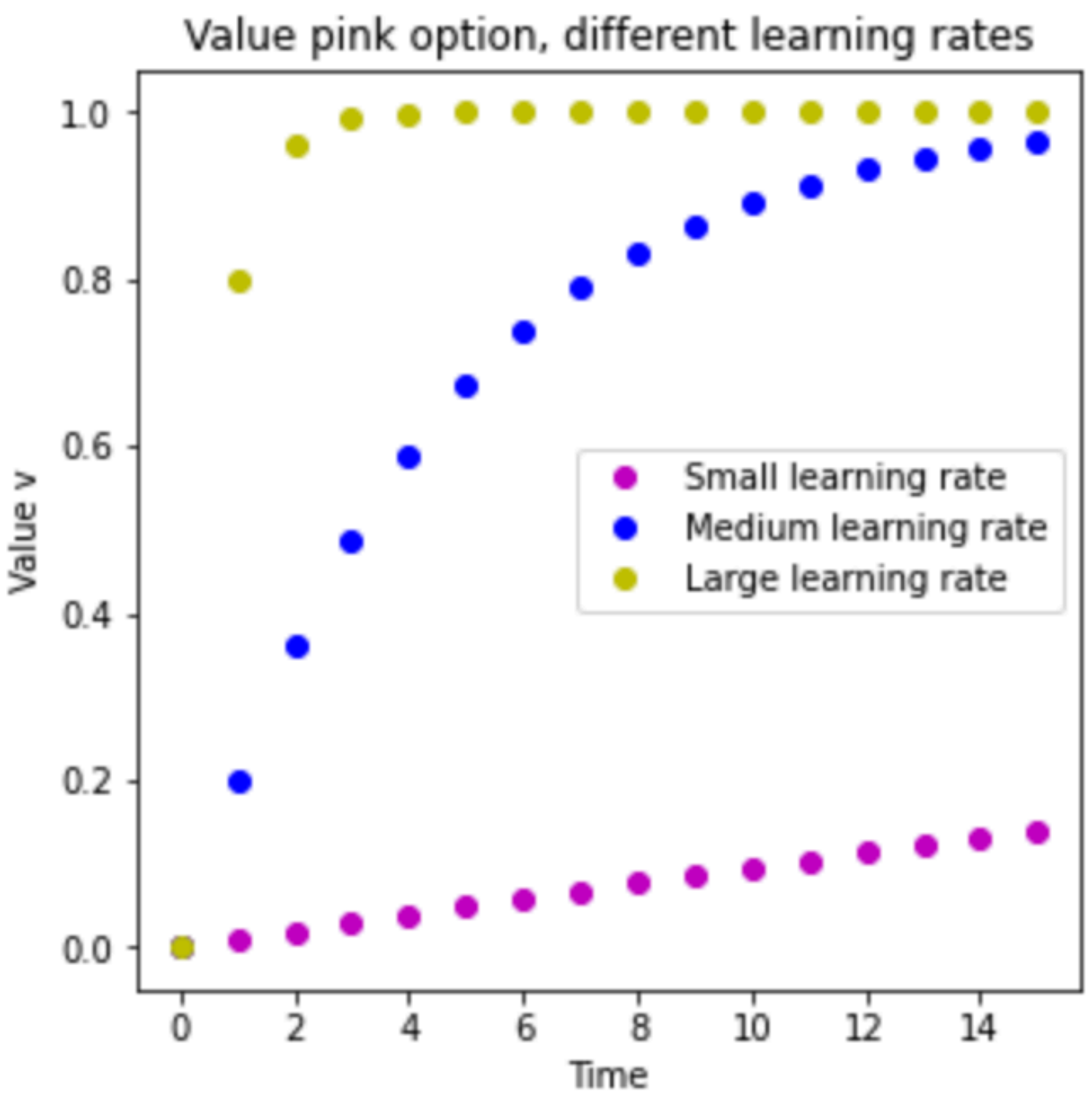
An introduction to Reinforcement Learning

17th of May 2022

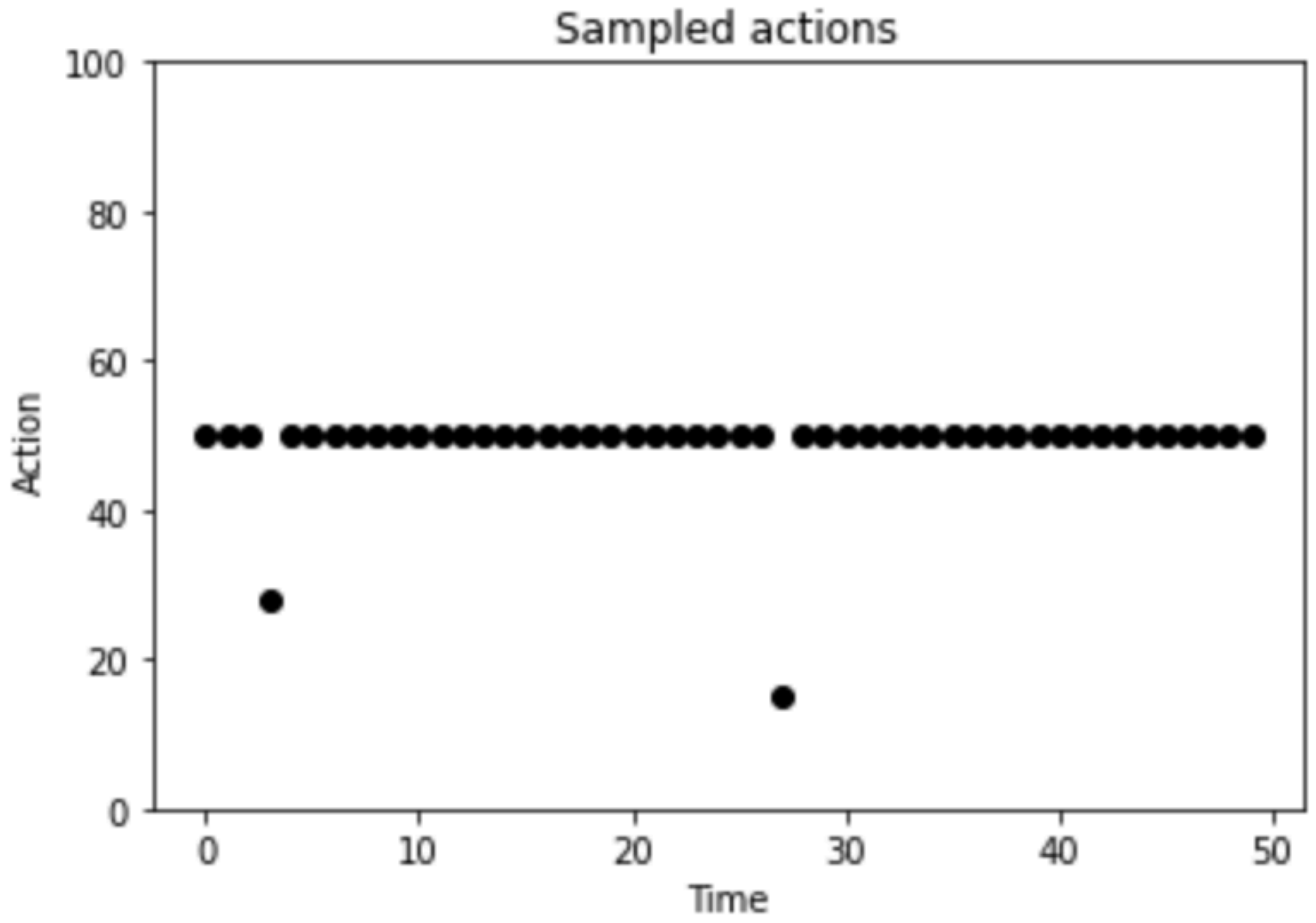
Recap: simple models of value learning



$$V(s) \leftarrow V(s) + \alpha \cdot (r - V(s))$$

```
# chicken learns that picking the pink option is valuable

# initiate the learning rate: this should neither be too high nor too low
# often, it should decrease over time
alpha = 0.2

# let's assume that 1 food item yields a reward of 1
r = 1

n_steps = 16 # the chicken makes 15 choices (and obtains 15 rewards)

# (initial values are important in more complicated examples)
v     = np.zeros(n_steps)

for iter in np.arange(n_steps-1):
  v[iter+1] = v[iter] + alpha * (r - v[iter])

plt.rcParams['figure.figsize'] = [5, 5]
plot_vals(np.arange(n_steps),v,"Value pink option","Time","Value v",'mo')
```

Recap: simple models of action selection



$$\pi(a, s) = P(a_t = a \,|\, s_t = s)$$

Maximum:

```
    action[iter] = np.argmax(simulate vals)
```

ϵ**-greedy**:

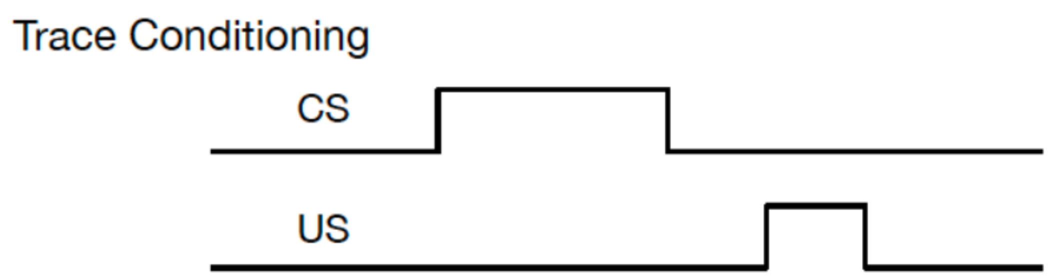
```
epsilon = 0.05

for iter in time_steps:

  rand_num = np.random.rand(1)

  if rand_num <= 1-epsilon:
    action[iter] = np.argmax(simulate_vals)
  else:
    action[iter] = np.random.choice(n_actions,1)
```

“Three” historical branches of RL

- Association learning, prediction (early 1900s)
- Optimal control (1950 onward)
- Learning *and* control (1980 onward)

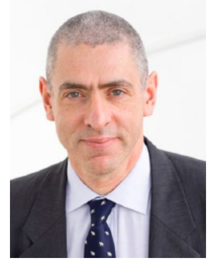
History: Learning *and* Control

- Key idea: use **experience** and **own value estimates**!
  - Allows to back-propagate info in time



- Approaches


  - **TD learning** (+ extensions to Actor-Critic, e.g. Sutton & Barto, 1981, 1982; Sutton 1988)
  - **Q learning** (Watkins 1989; Watkins & Dayan 1992)

    - Integrate *dynamic programming* (optimal control) with online learning
  - **SARSA**

Temporal Difference Learning

- "If one had to identify one idea as central and novel to reinforcement learning, it would undoubtedly be temporal-difference (TD) learning."

- Basic setup:
  - Learn directly from **raw experience** without a model of the environment's dynamics
  - Update estimates based in part on other learned estimates, without waiting for a final outcome (they **bootstrap**)
    - Learn "a guess from a guess"

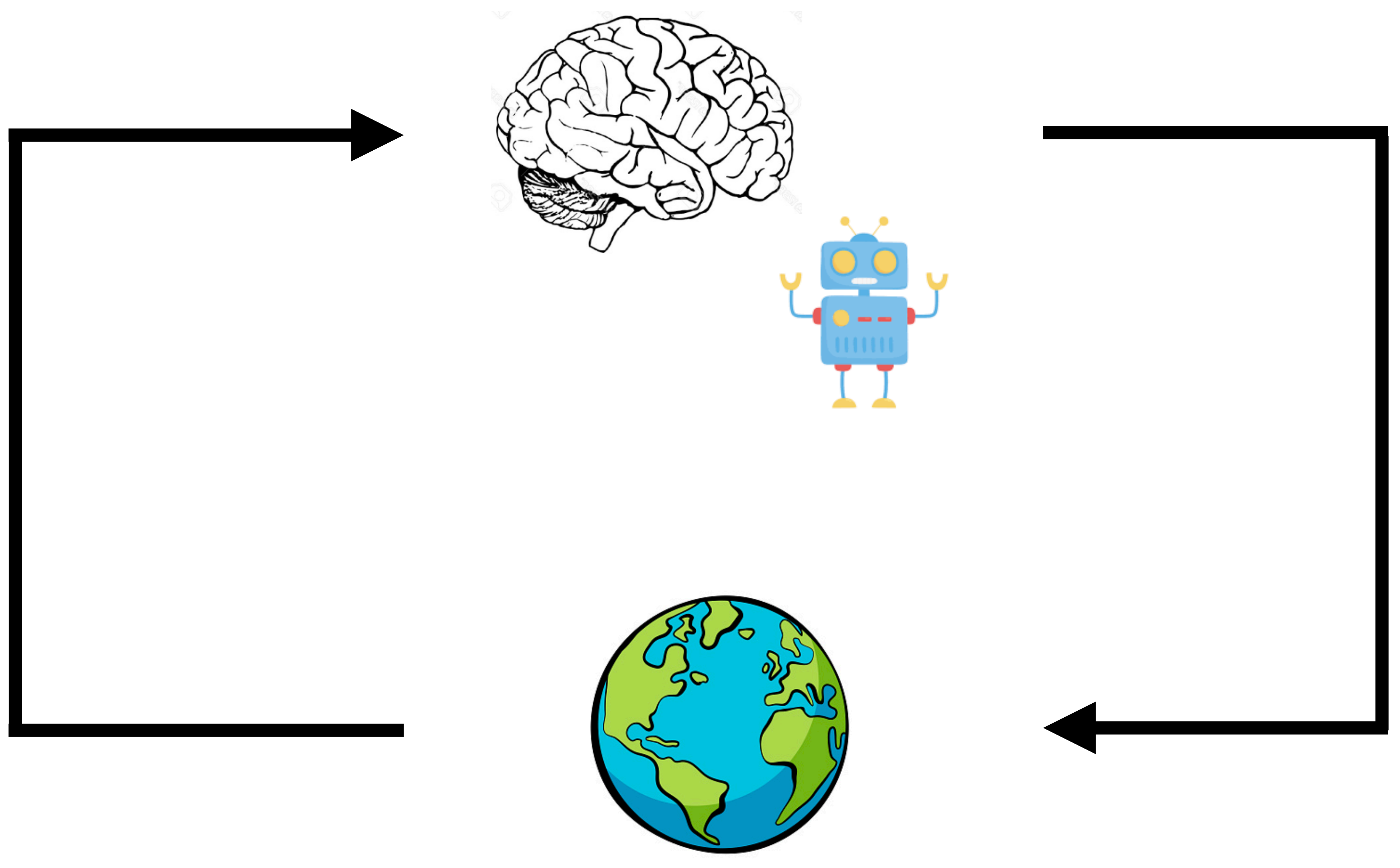
Temporal Difference Learning

- Advantages
  - Do not need a model of the environment
  - Implemented **online**, fully **incremental**
  - Shown to **converge** if learning rate small enough

Temporal Difference Learning

Based on a reward signal, agents learn **values of actions/states:**

$$V_\pi(s) = \mathbb{E}_\pi[R \mid s_0 = s]$$



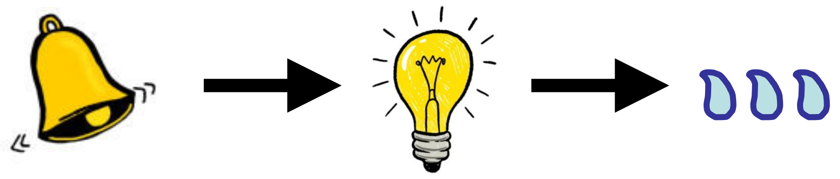
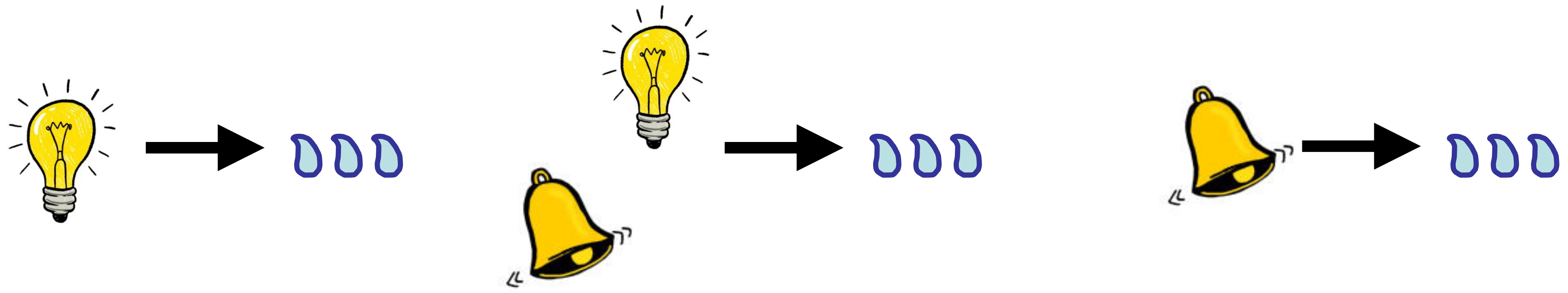
TD Learning:

$$V(s_t) \leftarrow V(s_t) + \underbrace{\alpha}_{\text{Learning rate}} \cdot \overbrace{(r + \underbrace{\gamma}_{\text{Discount rate}} \cdot V(s_{t+1}) - V(s_t))}^{\text{Prediction error}}$$

Rescorla Wagner Learning:

$$V(s_t) \leftarrow V(s_t) + \underbrace{\alpha}_{\text{Learning rate}} \cdot \overbrace{(r - V(s_t))}^{\text{Prediction error}}$$

Temporal Difference Learning

- Extends Rescorla–Wagner model
  - Learn within-trial *and* between-trial relationships
- Operates in 'real-time'
  - t labels time steps within *or* between trials
  - Think of time between t and $t+1$ as a small time interval (e.g. 1ms)
- Solves:
  - Higher order conditioning 
  - NO blocking if CS_2 is moved before previously learnt CS_1 
  - **Dopamine**…

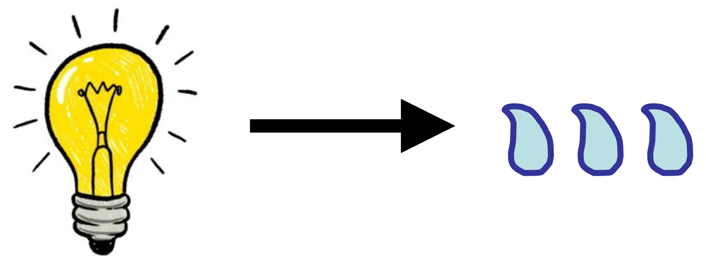
Prediction error

$$V(s_t) \leftarrow V(s_t) + \alpha \cdot (r + \gamma \cdot V(s_{t+1}) - V(s_t))$$

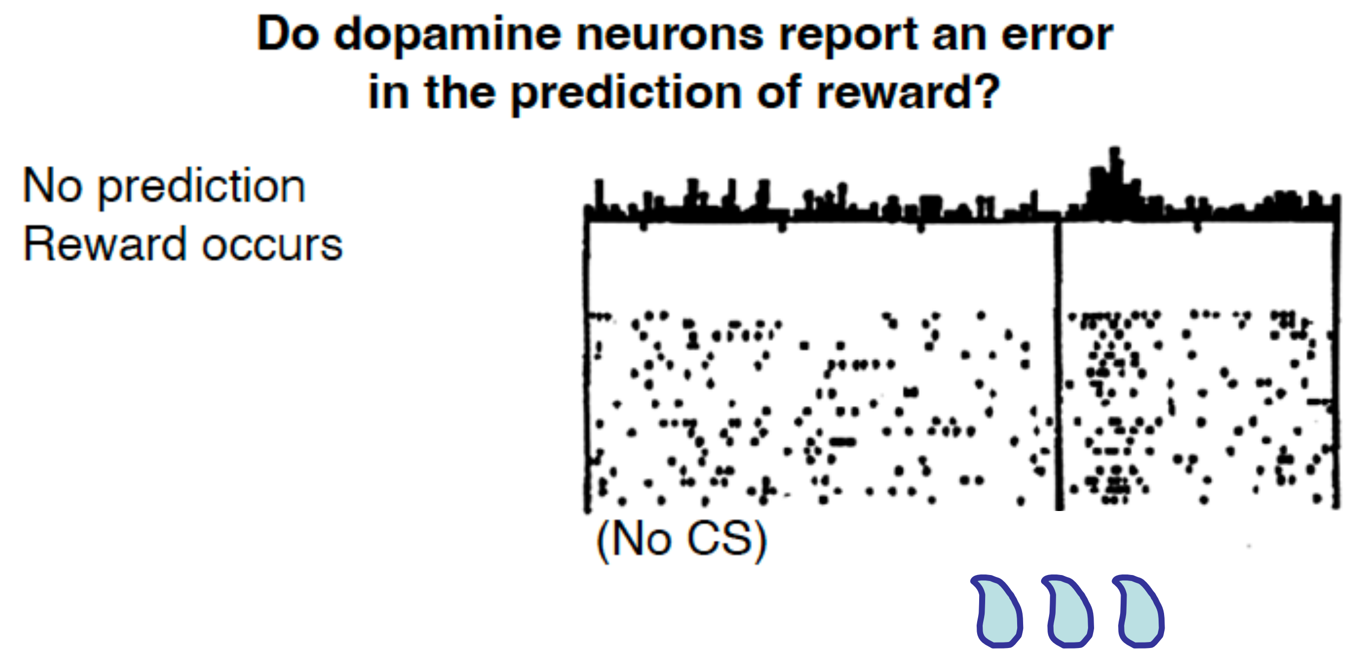
Learning rate Discount rate

Can RL tell us anything about the brain?

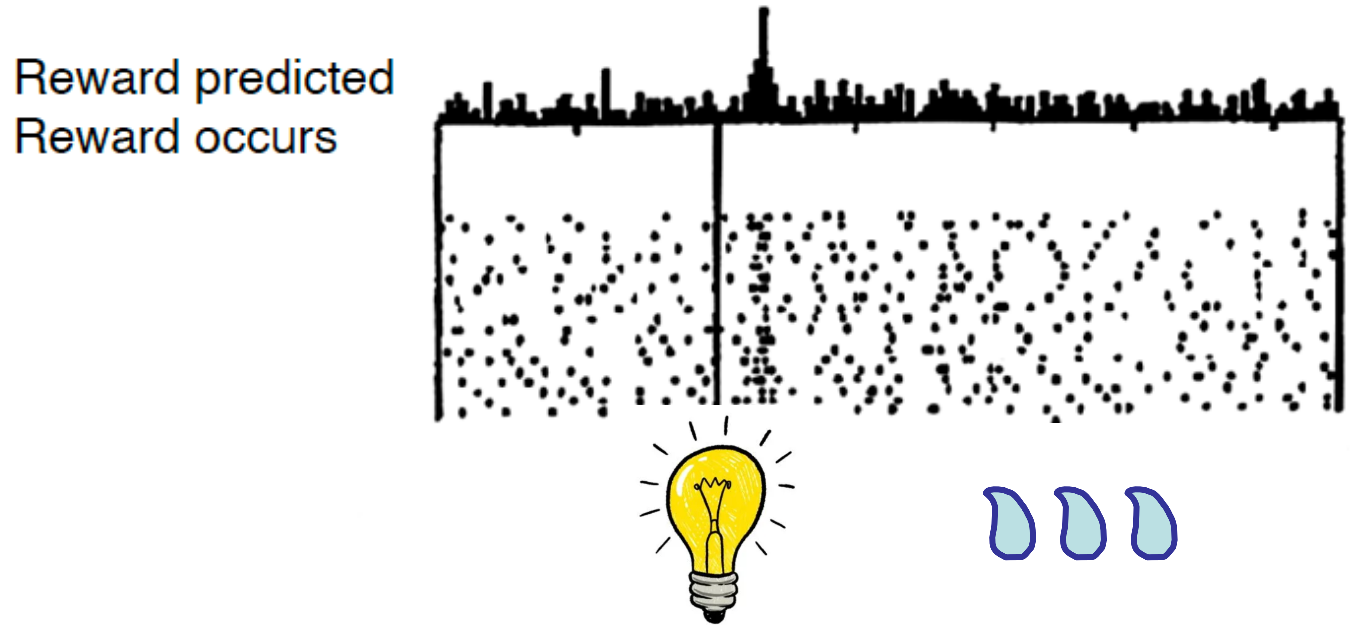
- Yes, quite a lot.
- Particularly, it looks like dopamine (DA) is a key neurotransmitter for (TD) reward learning
  - Schultz, Dayan & Montague (Science, 1997):



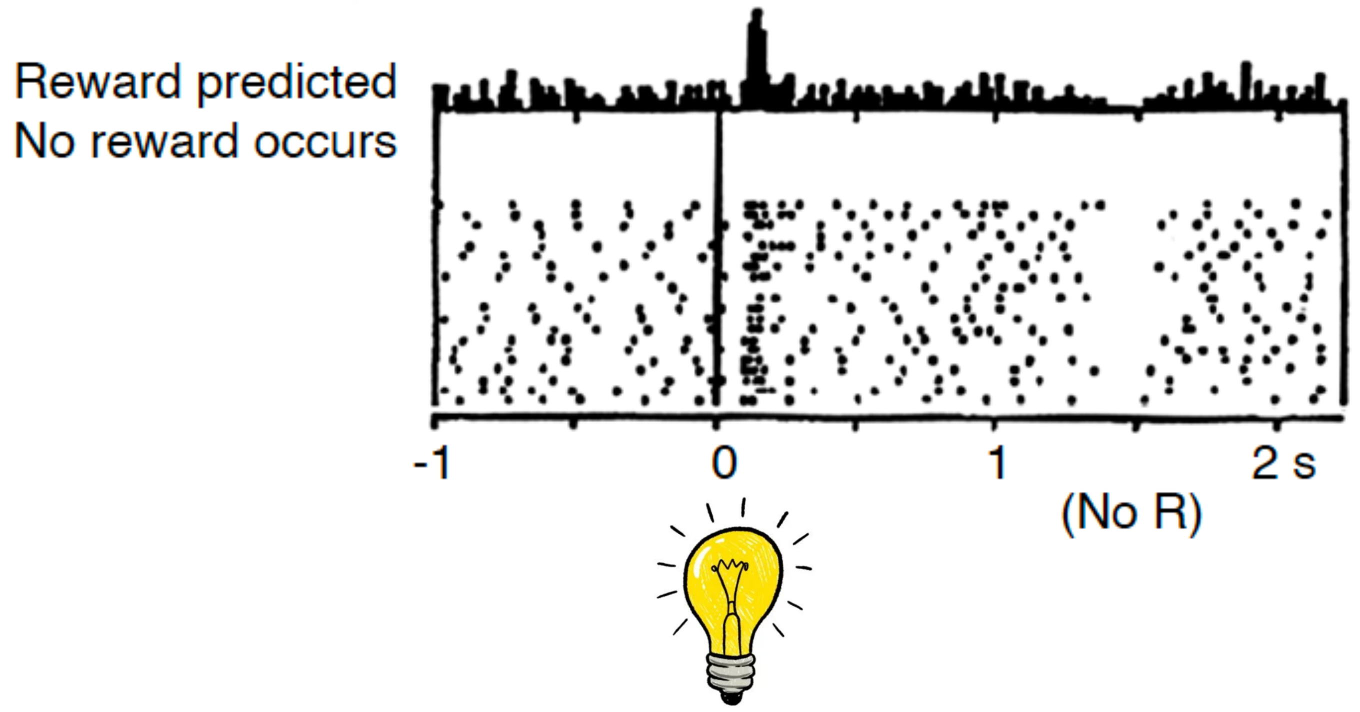
Dopamine neurons signal immediate reward



BUT: after training…
- DA signal reward prediction
- But not correctly predicted reward!



AND: it signals the unexpected omission of a reward!



This provides strong evidence that DA signals a **reward prediction error**

Coding: TD Learning

https://github.com/schwartenbeckph/RL-Course/tree/main/2022_05_17

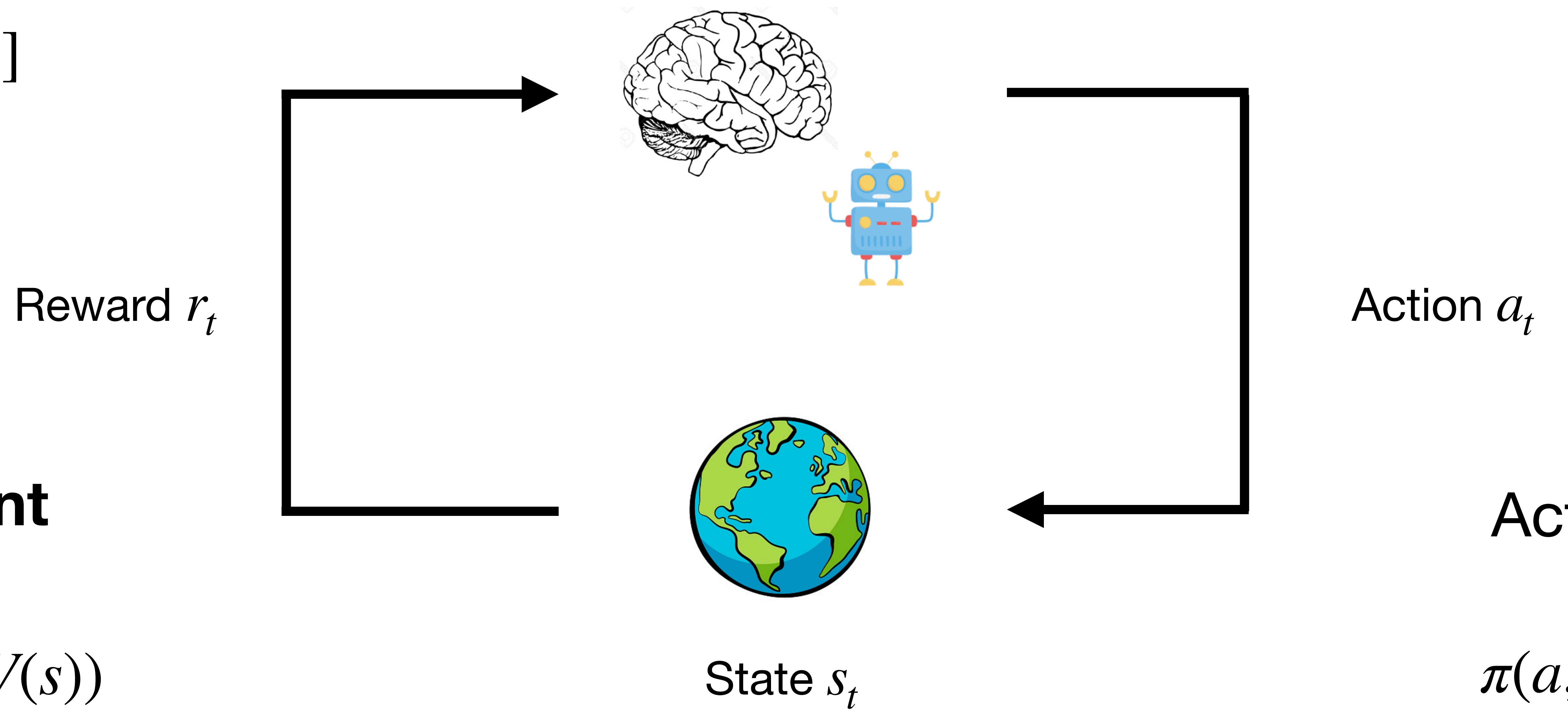
Recap: Basic setup: how to agents learn to act?

Based on a reward signal, agents learn **values of actions/states:**

$$V_\pi(s) = \mathbb{E}_\pi[R \mid s_0 = s]$$

Agents can learn a **model of the environment** to make smarter decisions, e.g.:

$$P(s_{t+1} = s \mid s_t = s, a_t = a)$$



Values can be **learnt** (simplified!!):

$$V(s) \leftarrow V(s) + \alpha \cdot (r - V(s))$$

Learning rate

Prediction error

Action is governed by a **policy**:

$$\pi(a, s) = P(a_t = a \mid s_t = s)$$