Reinforcement Learning Theory

Quentin Huys
MA MBBS PhD MBPsS

Translational Neuromodeling Unit, ETH and University of Zurich
Hospital of Psychiatry, University of Zurich

Friedrich Miescher Institute, Basel, 25/11/2013
Overview

- Reinforcement learning: rough overview
  - mainly following Sutton & Barto 1998

- Learning theory
  - classical & instrumental conditioning

- Dopamine
  - prediction errors and more
Types of learning

- Supervised
- Unsupervised
- Reinforcement learning
Setup

Environment

Agent

\[
\begin{align*}
{a_t} &\leftarrow \text{argmax} \sum_{t=1}^{\infty} r_t \\
\{a_t\} &\leftarrow \text{argmax} \sum_{t=1}^{\infty} r_t
\end{align*}
\]

After Sutton and Barto 1998
Reinforcement learning

State space

Gold +1

Electric shocks -1
A Markov Decision Problem

- \( s_t \in S \)
- \( a_t \in A \)
- \( T_{ss'}^a = p(s_{t+1} | s_t, a_t) \)
- \( r_t \sim R(s_{t+1}, a_t, s_t) \)
- \( \pi(a | s) = p(a | s) \)
\begin{align*}
\mathbf{s}_t & \in \mathcal{S} \\
\mathbf{a}_t & \in \mathcal{A} \\
T^{a}_{ss'} & = p(s_{t+1} | s_t, a_t) \\
\mathbf{r}_t & \sim \mathcal{R}(s_{t+1}, a_t, s_t) \\
\pi(a | s) & = p(a | s)
\end{align*}
Reinforcement Learning

MDP

\[ s_t \in S \]
\[ a_t \in A \]
\[ T^a_{ss'} = p(s_{t+1} | s_t, a_t) \]
\[ r_t \sim R(s_{t+1}, a_t, s_t) \]
\[ \pi(a | s) = p(a | s) \]
Actions

Action left

Action right

\[
T_{\text{left}} = \begin{bmatrix}
1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

\[
T_{\text{right}} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1
\end{bmatrix}
\]
Actions

Action left

Action right

$$T^{\text{left}} = \begin{bmatrix}
.8 & .8 & 0 & 0 & 0 & 0 & 0 \\
.2 & .2 & .8 & 0 & 0 & 0 & 0 \\
0 & 0 & .2 & .8 & 0 & 0 & 0 \\
0 & 0 & 0 & .2 & .8 & 0 & 0 \\
0 & 0 & 0 & 0 & .2 & .8 & 0 \\
0 & 0 & 0 & 0 & 0 & .2 & .8 \\
0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}$$

$$T^{\text{right}} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1
\end{bmatrix}$$

Noisy: plants, environments, agent
Reinforcement learning

Friedrich Miescher Institute, Basel, 25/11/2013

Quentin Huys, ETHZ / PUK

Actions

Action left

Action right

Noisy: plants, environments, agent

Absorbing state -> max eigenvalue < 1

\( T^{\text{left}} = \begin{bmatrix}
\cdot8 & \cdot8 & 0 & 0 & 0 & 0 & 0 \\
\cdot2 & \cdot2 & \cdot8 & 0 & 0 & 0 & 0 \\
0 & 0 & \cdot2 & \cdot8 & 0 & 0 & 0 \\
0 & 0 & 0 & \cdot2 & \cdot8 & 0 & 0 \\
0 & 0 & 0 & 0 & \cdot2 & \cdot8 & 0 \\
0 & 0 & 0 & 0 & 0 & \cdot2 & \cdot8 \\
0 & 0 & 0 & 0 & 0 & 0 & \cdot8
\end{bmatrix} \)

\( T^{\text{right}} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1
\end{bmatrix} \)
Markovian dynamics

\[ p(s_{t+1} \mid a_t, s_t, a_{t-1}, s_{t-1}, a_{t-2}, s_{t-2}, \cdots) = p(s_{t+1} \mid a_t, s_t) \]

Velocity
Markovian dynamics

\[ p(s_{t+1} | a_t, s_t, a_{t-1}, s_{t-1}, a_{t-2}, s_{t-2}, \cdots) = p(s_{t+1} | a_t, s_t) \]
Markovian dynamics

\[ p(s_{t+1}|a_t, s_t, a_{t-1}, s_{t-1}, a_{t-2}, s_{t-2}, \cdots) = p(s_{t+1}|a_t, s_t) \]

Velocity

\[ s' = [\text{position}] \rightarrow s' = \begin{bmatrix} \text{position} \\ \text{velocity} \end{bmatrix} \]
MDP

\[ s_t \in S \]
\[ a_t \in A \]
\[ T_{ss'}^a = p(s_{t+1} | s_t, a_t) \]
\[ r_t \sim R(s_{t+1}, a_t, s_t) \]
\[ \pi(a | s) = p(a | s) \]
MDP

\[ s_t \in S \]
\[ a_t \in A \]
\[ T_{ss'}^a = p(s_{t+1} | s_t, a_t) \]
\[ r_t \sim R(s_{t+1}, a_t, s_t) \]
\[ \pi(a | s) = p(a | s) \]
MDP

\[ s_t \in \mathcal{S} \]
\[ a_t \in \mathcal{A} \]
\[ T^{a}_{ss'} = p(s_{t+1} | s_t, a_t) \]
\[ r_t \sim \mathcal{R}(s_{t+1}, a_t, s_t) \]
\[ \pi(a | s) = p(a | s) \]
Tall orders

- Aim: maximise total future reward

\[ \sum_{t=1}^{\infty} r_t \]

- i.e. we have to sum over paths through the future and weigh each by its probability

- Best policy achieves best long-term reward
Exhaustive tree search
Exhaustive tree search
Decision tree

\[ \sum_{t=1}^{\infty} r_t \]
Decision tree

\[ \sum_{t=1}^{\infty} r_t \]
\[ \sum_{t=1}^{\infty} r_t \]
Decision tree

\[
\sum_{t=1}^{\infty} r_t
\]
Policy for this talk

- Pose the problem mathematically
- Policy evaluation
- Policy iteration
- Monte Carlo techniques: experience samples
- TD learning

Policy

Evaluate $\leftrightarrow$ Update
Evaluating a policy

- **Aim:** maximise total future reward
  \[ \sum_{t=1}^{\infty} r_t \]

- To know which is best, evaluate it first

- The policy determines the expected reward from each state
  \[ V^\pi(s_1) = \mathbb{E} \left[ \sum_{t=1}^{\infty} r_t | s_1 = 1, a_t \sim \pi \right] \]
Discounting

- Given a policy, each state has an expected value

\[ V^\pi(s_1) = \mathbb{E}\left[ \sum_{t=1}^{\infty} r_t \mid s_1 = 1, a_t \sim \pi \right] \]

- But:

\[ \sum_{t=0}^{\infty} r_t = \infty \]

- Episodic

\[ \sum_{t=0}^{T} r_t < \infty \]

- Discounted
  - infinite horizons

\[ \sum_{t=0}^{\infty} \gamma^t r_t < \infty \]
  - finite, exponentially distributed horizons

\[ \sum_{t=0}^{T} \gamma^t r_t \quad T \sim \frac{1}{\tau} e^{t/\tau} \]
Discounting

- Given a policy, each state has an expected value

\[
V^\pi(s_1) = \mathbb{E}\left[ \sum_{t=1}^{\infty} r_t | s_1 = 1, a_t \sim \pi \right]
\]

- But:

\[
\sum_{t=0}^{\infty} r_t = \infty
\]

- Episodic

\[
\sum_{t=0}^{T} r_t < \infty
\]

- Discounted
  - infinite horizons

\[
\sum_{t=0}^{\infty} \gamma^t r_t < \infty
\]
  - finite, exponentially distributed horizons

\[
\sum_{t=0}^{T} \gamma^t r_t \quad T \sim \frac{1}{\tau} e^{t/\tau}
\]
Discounting

- Given a policy, each state has an expected value

\[ V^\pi(s_1) = \mathbb{E} \left[ \sum_{t=1}^{\infty} r_t | s_1 = 1, a_t \sim \pi \right] \]

- But:

\[ \sum_{t=0}^{\infty} r_t = \infty \]

- Episodic

\[ \sum_{t=0}^{T} r_t < \infty \]

- Discounted
  - infinite horizons
    \[ \sum_{t=0}^{\infty} \gamma^t r_t < \infty \]
  - finite, exponentially distributed horizons
    \[ \sum_{t=0}^{T} \gamma^t r_t \sim \frac{1}{\tau} e^{t/\tau} \]
Markov Decision Problems

\[ V^\pi(s_t) = \mathbb{E} \left[ \sum_{t'=1}^{\infty} r_{t'} \mid s_t = s, \pi \right] \]

\[ = \mathbb{E} [r_1 \mid s_t = s, \pi] + \mathbb{E} \left[ \sum_{t=2}^{\infty} r_t \mid s_t = s, \pi \right] \]

\[ = \mathbb{E} [r_1 \mid s_t = s, \pi] + \mathbb{E} [V^\pi(s_{t+1}) \mid s_t = s, \pi] \]

This dynamic consistency is key to many solution approaches. It states that the value of a state \( s \) is related to the values of its successor states \( s' \).
Markov Decision Problems

\[ V^\pi(s_t) = \mathbb{E}[r_1 | s_t = s, \pi] + \mathbb{E}[V(s_{t+1}), \pi] \]

\[ r_1 \sim \mathcal{R}(s_2, a_1, s_1) \]

\[ \mathbb{E}[r_1 | s_t = s, \pi] = \mathbb{E} \left[ \sum_{s_{t+1}} p(s_{t+1} | s_t, a_t) \mathcal{R}(s_{t+1}, a_t, s_t) \right] \]

\[ = \sum_{a_t} p(a_t | s_t) \left[ \sum_{s_{t+1}} p(s_{t+1} | s_t, a_t) \mathcal{R}(s_{t+1}, a_t, s_t) \right] \]

\[ = \sum_{a_t} \pi(a_t, s_t) \left[ \sum_{s_{t+1}} \mathcal{T}_{s_t s_{t+1}}^a \mathcal{R}(s_{t+1}, a_t, s_t) \right] \]
Bellman equation

\[ V^\pi(s_t) = \mathbb{E}[r_1|s_t = s, \pi] + \mathbb{E}[V(s_{t+1}), \pi] \]

\[ \mathbb{E}[r_1|s_t, \pi] = \sum_a \pi(a, s_t) \left[ \sum_{s_{t+1}} T^a_{s_t s_{t+1}} R(s_{t+1}, a, s_t) \right] \]

\[ \mathbb{E}[V^\pi(s_{t+1}), \pi, s_t] = \sum_a \pi(a, s_t) \left[ \sum_{s_{t+1}} T^a_{s_t s_{t+1}} V^\pi(s_{t+1}) \right] \]

\[ V^\pi(s) = \sum_a \pi(a|s) \left[ \sum_{s'} T^a_{ss'} [R(s', a, s) + V^\pi(s')] \right] \]
Bellman Equation

\[ V^\pi(s) = \sum_a \pi(a|s) \left[ \sum_{s'} T_{ss'}^a \left[ R(s', a, s) + V^\pi(s') \right] \right] \]
Bellman Equation

\[ V^\pi(s) = \sum_a \pi(a|s) \left[ \sum_{s'} \mathcal{T}_{ss'}^a [R(s', a, s) + V^\pi(s')] \right] \]
Bellman Equation

\[ V^\pi(s) = \sum_a \pi(a|s) \left[ \sum_{s'} T_{ss'}^a [R(s', a, s) + V^\pi(s')] \right] \]

All future reward from state \( s \)

= \( \mathbb{E} \) Immediate reward + All future reward from next state \( s' \)
\[ V^\pi(s) = \sum_a \pi(a|s) \left[ \sum_{s'} T^{a}_{ss'} [R(s', a, s) + V^\pi(s')] \right] \]

- so we can define state-action values as:
  \[ Q(s, a) = \sum_{s'} T^{a}_{ss'} [R(s', a, s) + V(s')] \]
  \[ = \mathbb{E} \left[ \sum_{t=1}^{\infty} r_t | s, a \right] \]

- and state values are average state-action values:
  \[ V(s) = \sum_a \pi(a|s) Q(s, a) \]
Bellman Equation

\[ V^\pi(s) = \sum_a \pi(a|s) \left[ \sum_{s'} \mathcal{T}_{ss'}^a [R(s', a, s) + V^\pi(s')] \right] \]

- to evaluate a policy, we need to solve the above equation, i.e. find the self-consistent state values

- options for policy evaluation
  - exhaustive tree search - outwards, inwards, depth-first
  - linear solution in 1 step
  - value iteration: iterative updates
  - experience sampling
Solving the Bellman Equation

Option 1: turn it into update equation

Option 2: linear solution  (w/ absorbing states)

\[ V(s) = \sum_a \pi(a, s_t) \left[ \sum_{s'} T_{ss'}^a \left[ R(s', a, s) + V(s') \right] \right] \]

\[ \Rightarrow \mathbf{v} = \mathbf{R}^\pi + \mathbf{T}^\pi \mathbf{v} \]

\[ \Rightarrow \mathbf{v}^\pi = (\mathbf{I} - \mathbf{T}^\pi)^{-1} \mathbf{R}^\pi \quad \mathcal{O}(|S|^3) \]
Solving the Bellman Equation

Option 1: turn it into update equation

\[ V^{k+1}(s) = \sum_{a} \pi(a, s_t) \left[ \sum_{s'} T_{ss'}^a \left[ R(s', a, s) + V^k(s') \right] \right] \]

Option 2: linear solution

\[ V(s) = \sum_{a} \pi(a, s_t) \left[ \sum_{s'} T_{ss'}^a \left[ R(s', a, s) + V(s') \right] \right] \]

\[ \Rightarrow \mathbf{v} = \mathbf{R}^\pi + \mathbf{T}^\pi \mathbf{v} \]

\[ \Rightarrow \mathbf{v}^\pi = (\mathbf{I} - \mathbf{T}^\pi)^{-1} \mathbf{R}^\pi \quad \mathcal{O}(|S|^3) \]
Given the value function for a policy, say via linear solution

\[
V^\pi(s) = \sum_a \pi(a|s) \left[ \sum_{s'} T_{ss'}^a \left[ R(s', a, s) + V^\pi(s') \right] \right] \\
Q^\pi(s, a)
\]

Given the values V for the policy, we can improve the policy by always choosing the best action:

\[
\pi'(a|s) = \begin{cases} 
1 & \text{if } a = \arg\max_a Q^\pi(s, a) \\
0 & \text{else}
\end{cases}
\]

It is guaranteed to improve:

\[
Q^\pi(s, \pi'(s)) = \max_a Q^\pi(s, a) \geq Q^\pi(s, \pi(s)) = V^\pi(s)
\]

for deterministic policy
Policy iteration

Policy evaluation

\[ \pi^\pi = (I - T^\pi)^{-1} R^\pi \]

\[
\pi(a|s) = \begin{cases} 
1 & \text{if } a = \arg\max_a \sum_{s'} T^{a}_{ss'} \left[ R^{a}_{ss'} + V^{pi}(s') \right] \\
0 & \text{else}
\end{cases}
\]
Policy iteration

Policy evaluation

\[ v^\pi = (I - T^\pi)^{-1} R^\pi \]

greedy policy improvement

\[ \pi(a|s) = \begin{cases} 1 & \text{if } a = \text{argmax}_a \sum_{s'} T_{ss'}^a [R_{ss}^a + V^{p_i}(s')] \\ 0 & \text{else} \end{cases} \]
Policy iteration

Policy evaluation

\[ v^\pi = (I - T^\pi)^{-1} R^\pi \]

Value iteration

\[ V^*(s) = \max_a \sum_{s'} T^a_{ss'} [R^a_{ss} + V^*(s')] \]

greedy policy improvement

\[ \pi(a|s) = \begin{cases} 
  1 & \text{if } a = \arg\max_a \sum_{s'} T^a_{ss'} [R^a_{ss} + V^{\pi}(s')] \\
  0 & \text{else} 
\end{cases} \]
Model-free solutions

- So far we have assumed knowledge of R and T
  - R and T are the ‘model’ of the world, so we assume full knowledge of the dynamics and rewards in the environment

- What if we don’t know them?
- We can still learn from state-action-reward samples
  - we can learn R and T from them, and use our estimates to solve as above
  - alternatively, we can directly estimate V or Q
Solving the Bellman Equation

Option 3: sampling

\[ V(s) = \sum_a \pi(a, s_t) \left[ \sum_{s'} T^{a}_{ss'} [R(s', a, s) + V(s')] \right] \]

So we can just draw some samples from the policy and the transitions and average over them:

\[ a = \sum_{k} f(x_k)p(x_k) \]

\[ x^{(i)} \sim p(x) \rightarrow \hat{a} = \frac{1}{N} \sum_i f(x^{(i)}) \]
Solving the Bellman Equation

Option 3: sampling

So we can just draw some samples from the policy and the transitions and average over them:

\[ a = \sum_{k} f(x_k) p(x_k) \]

\[ x(i) \sim p(x) \rightarrow \hat{a} = \frac{1}{N} \sum_{i} f(x(i)) \]
Solving the Bellman Equation

Option 3: sampling

this is an expectation over policy and transition samples.

So we can just draw some samples from the policy and the transitions and average over them:

\[
\hat{a} = \sum_k f(x_k)p(x_k)
\]

\[
x^{(i)} \sim p(x) \rightarrow \hat{a} = \frac{1}{N} \sum_i f(x^{(i)})
\]
Solving the Bellman Equation

Option 3: sampling

this is an expectation over policy and transition samples.

So we can just draw some samples from the policy and the transitions and average over them:

\[ a = \sum_{k} f(x_k)p(x_k) \]

\[ x^{(i)} \sim p(x) \rightarrow \hat{a} = \frac{1}{N} \sum_{i} f(x^{(i)}) \]

more about this later...
Learning from samples

A new problem: exploration versus exploitation
Monte Carlo

- **First visit MC**
  - randomly start in all states, generate paths, average for starting state only
  \[ \mathcal{V}(s) = \frac{1}{N} \sum_i \sum_{t' = 1}^{T} r^i_{t'} | s_0 = s \]  

- **More efficient use of samples**
  - Every visit MC
  - Bootstrap: TD
  - Dyna

- **Better samples**
  - on policy versus off policy
  - Stochastic search, UCT...
Update equation: towards TD

Bellman equation

\[ V(s) = \sum_a \pi(a, s) \left[ \sum_{s'} T_{ss'}^a \left[ \mathcal{R}(s', a, s) + V(s') \right] \right] \]

Not yet converged, so it doesn’t hold:

\[ dV(s) = -V(s) + \sum_a \pi(a, s) \left[ \sum_{s'} T_{ss'}^a \left[ \mathcal{R}(s', a, s) + V(s') \right] \right] \]

And then use this to update

\[ V^{i+1}(s) = V^i(s) + dV(s) \]
TD learning

\[ dV(s) = -V(s) + \sum_a \pi(a, s) \left[ \sum_{s'} T_{ss'}^a \left[ R(s', a, s) + V(s') \right] \right] \]
TD learning

\[ dV(s) = -V(s) + \sum_a \pi(a, s) \left[ \sum_{s'} T_{ss'}^a \left[ R(s', a, s) + V(s') \right] \right] \]

\[ a_t \sim \pi(a|s_t) \]

\[ s_{t+1} \sim T_{s_t,s_{t+1}}^{a_t} \]

\[ r_t = R(s_{t+1}, a_t, s_t) \]
TD learning

\[ dV(s) = -V(s) + \sum_a \pi(a, s) \left[ \sum_{s'} T^{a}_{ss'} [\mathcal{R}(s', a, s) + V(s')] \right] \]

\[ a_t \sim \pi(a|s_t) \]
\[ s_{t+1} \sim T^{a_t}_{s_t, s_{t+1}} \]
\[ r_t = \mathcal{R}(s_{t+1}, a_t, s_t) \]

\[ \delta_t = -V_{t-1}(s_t) + r_t + V_{t-1}(s_{t+1}) \]
### TD learning

\[
dV(s) = -V(s) + \sum_a \pi(a, s) \left[ \sum_{s'} T_{ss'}^a [R(s', a, s) + V(s')] \right]
\]

Sample

\[
\begin{align*}
    a_t & \sim \pi(a|s_t) \\
    s_{t+1} & \sim T_{s_t, s_{t+1}}^{a_t} \\
    r_t & = R(s_{t+1}, a_t, s_t)
\end{align*}
\]

\[
\delta_t = -V_{t-1}(s_t) + r_t + V_{t-1}(s_{t+1})
\]

\[
V^{i+1}(s) = V^i(s) + dV(s) \\
V_t(s_t) = V_{t-1}(s_t) + \alpha \delta_t
\]
TD learning

\[ a_t \sim \pi(a|s_t) \]

\[ s_{t+1} \sim T_{s_t,a_t}^{s_{t+1}} \]

\[ r_t = R(s_{t+1}, a_t, s_t) \]

\[ \delta_t = -V_t(s_t) + r_t + V_t(s_{t+1}) \]

\[ V_{t+1}(s_t) = V_t(s_t) + \alpha \delta_t \]
Aside: what makes a TD error?

- unpredicted reward expectation change
- disappears with learning
- stays with probabilistic reinforcement
- sequentiality
  - TD error vs prediction error

Schultz et al.
Pavlovian conditioning

WATCH WHAT I CAN MAKE PAVLOV DO. AS SOON AS I DROOL, HE'LL SMILE AND WRITE IN HIS LITTLE BOOK.
Pavlovian conditioning

prior to training

CS: bell -> no response
US: food -> salivation & consumption
Pavlovian conditioning

prior to training
  CS: bell -> no response
  US: food -> salivation & consumption

training
  CS: bell  US: food
Pavlovian conditioning

prior to training

CS: bell -> no response
US: food -> salivation & consumption

training

CS: bell → US: food
Pavlovian conditioning

prior to training
CS: bell -> no response
US: food -> salivation & consumption

training
CS: bell -> US: food

after training:
CS: bell -> salivation
US: food -> salivation & consumption
Pavlovian conditioning

prior to training

CS: bell -> no response
US: food -> salivation & consumption

training

CS: bell → US: food

after training:

CS: bell -> salivation
US: food -> salivation & consumption

omission training:

CS: bell

salivation: US: omitted

no salivation: US: food
Pavlovian and instrumental conditioning

- Pavlovian model-free learning:

\[ V_t(s) = V_{t-1}(s) + \epsilon (r_t - V_{t-1}(s)) \]

e.g. Rescorla & Wagner 1972
Pavlovian values in the brain?

- Where the values $V(s)$ are is not so clear
- Midbrain dopamine neurons report a TD error
Pavlovian values in the brain?

- Where the values $V(s)$ are is not so clear
- Midbrain dopamine neurons report a TD error

Montague et al. 1996
Pavlovian values in the brain?

- Where the values $V(s)$ are is not so clear
- Midbrain dopamine neurons report a TD error
Phasic signals in humans

D'Ardenne et al. 2008 Science

Zaghloul et al. 2009 Science
Blocking

- Are predictions and prediction errors really causally important in learning?

1. A $\rightarrow$ Reward
2. A+B $\rightarrow$ Reward
3. A $\rightarrow$ ? approach
4. B $\rightarrow$ ? approach

Kamin 1968
Causal role of phasic DA in learning

<table>
<thead>
<tr>
<th>Single cue</th>
<th>Compound cue</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>14–15 d</td>
<td>4 d</td>
<td>1 d</td>
</tr>
<tr>
<td>A → US</td>
<td>AX → US</td>
<td>X?</td>
</tr>
</tbody>
</table>

With paired or unpaired optical stimulation

**Figure 1**

**Figure 2**

**Graph**

*Steinberg et al. 2013 Nat. Neurosci.*
### TABLE 8.4
Blocking of aversive conditioning

<table>
<thead>
<tr>
<th>Conditions</th>
<th>Stage 1</th>
<th>Stage 2</th>
<th>test</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A→ shock</td>
<td>AX→ shock</td>
<td>X</td>
</tr>
<tr>
<td>2</td>
<td>control treatments</td>
<td>AX→ shock</td>
<td>X</td>
</tr>
<tr>
<td>3</td>
<td>A→ food omission</td>
<td>AX→ shock</td>
<td>X</td>
</tr>
</tbody>
</table>

**Panel A - aversive excitor** 
(Rescorla, 1971)

**Panel B - attractive inhibitor** 
(Dickinson, 1976)

Dickinson & Dearing 1979
Pavlovian conditioning

- By being predictive of an affective outcome, a neutral stimulus can come to elicit the innate preparatory response usually evoked by the affective outcome.

![Graph showing Pavlovian conditioning](image)
Pavlovian and instrumental conditioning

- Pavlovian model-free learning:
  \[ V_t(s) = V_{t-1}(s) + \epsilon (r_t - V_{t-1}(s)) \]
  \[ p(a|s, V) \propto f(a, V(s)) p(a|s) \]

- Instrumental model-free learning:
  \[ Q_t(a, s) = Q_{t-1}(a, s) + \epsilon (r_t - Q_{t-1}(a, s)) \]

  e.g. Rescorla & Wagner 1972
Prediction errors for habits

- **SARSA**

- **Q learning**

Morris et al. 2006

Roesch et al. 2007
Cached learning ~ prediction errors
prediction errors ~ phasic dopamine

phasic dopamine ~ cached learning
Pavlovian influences on instrumental learning

Go

Nogo

Rewarded

Avoids loss

Pavlovian influences on instrumental learning

Pavlovian influences on instrumental learning

Model

Reinforcement learning

Quentin Huys, ETHZ / PUK
Friedrich Miescher Institute, Basel, 25/11/2013

Go rewarded
Go to win
Probability(Go)
20 40 60
0
0.5
1

Nogo punished
Go to avoid
20 40 60
0
0.5
1

Nogo rewarded
Nogo to win
20 40 60
0
0.5
1

Go punished
Nogo to avoid
20 40 60
0
0.5
1

Model

\[ p(\text{go} | s_t) \propto Q_t(\text{go} | s_t) + \text{bias(\text{go})} \]

Model

\[ p(go|s_t) \propto Q_t(go|s_t) + \text{bias}(go) + V_t(s_t) \]
\[ V_t(s_t) = V_{t-1}(s_t) + \epsilon(r_t - V_{t-1}(s_t)) \]

Model

\[ P(\text{go}) \propto \text{value of stimulus} \]

\[ p(\text{go}|s_t) \propto Q_t(\text{go}|s_t) + \text{bias(\text{go})} + V_t(s_t) \]

\[ V_t(s_t) = V_{t-1}(s_t) + \epsilon(r_t - V_{t-1}(s_t)) \]


Reinforcement learning
Addictive Pavlovian values

Addictive Pavlovian values

Addictive Pavlovian values

Pavlovian state value acquisition

No Pavlovian state learning

Addictive Pavlovian values

Pavlovian state value acquisition

Impulsive
Amphetamine
DA antagonism
D2R<->phasic DA

No Pavlovian state learning

Pavlovian state values: sign tracking

Flagel et al. 2011, Huys et al. 2011a

Sign trackers

Goal trackers
Pavlovian state values: sign tracking

Sign trackers

Goal trackers

at risk for addiction

Flagel et al. 2011, Huys et al. 2011a
Sign-tracking vs goal-tracking

The idea that the incentive motivational properties of reward-reinforcer pairs are computed and used to classify animals. In both Panels red filled symbols indicate GTs, white symbols INs, and blue filled symbols STs, classified as two subpopulations by Days 4 and 5 of training. (A) Activity, as the number of lever deflections (A) or food cup entries (B) during the CS period, over 5 days of Pavlovian training. A similar association between PCA and conditioned reinforcement was found in data reanalyzed from two independent studies.[9,25].

Figure 7. Mean ± SEM number of lever deflections (A) or food cup entries (B) during the CS period, over 5 days of Pavlovian training.

Figure 8. The propensity to approach a food cue (CS) correlates with the ability of the same lever-CS to support learning a new instrumental conditioned reinforcement, as a function of total lever contacts. Panel B shows the same data, but when each animal's PCA Index Score was calculated and used to classify animals. In both Panels red filled symbols indicate GTs, white symbols INs, and blue filled symbols STs, classified as two subpopulations by Days 4 and 5 of training.

Figure 9. For both STs and GTs, pairing the CS and US increased the probability of approach during the CS period, to a craving level, indicative of learning a new instrumental conditioned reinforcement or PIT, for three reasons. First, PCA is procedurally dissociated and stable.[9]. Finally, STs and GTs do not differ in the topography of the CR, indicating the extent to which animals fail to learn either a ST or GT CR (Fig. 8; [9,25]; Fig. 9). For both STs and GTs, pairing the CS and US increased the probability of approach during the CS period, to a craving level, indicative of learning a new instrumental conditioned reinforcement or PIT, for three reasons. First, PCA is procedurally dissociated and stable.[9]. Finally, STs and GTs do not differ in the topography of the CR, indicating the extent to which animals fail to learn either a ST or GT CR (Fig. 8; [9,25]; Fig. 9). For both STs and GTs, pairing the CS and US increased the probability of approach during the CS period, to a craving level, indicative of learning a new instrumental conditioned reinforcement or PIT, for three reasons. First, PCA is procedurally dissociated and stable.[9]. Finally, STs and GTs do not differ in the topography of the CR, indicating the extent to which animals fail to learn either a ST or GT CR (Fig. 8; [9,25]; Fig. 9). For both STs and GTs, pairing the CS and US increased the probability of approach during the CS period, to a craving level, indicative of learning a new instrumental conditioned reinforcement or PIT, for three reasons. First, PCA is procedurally dissociated and stable.[9]. Finally, STs and GTs do not differ in the topography of the CR, indicating the extent to which animals fail to learn either a ST or GT CR (Fig. 8; [9,25]; Fig. 9). For both STs and GTs, pairing the CS and US increased the probability of approach during the CS period, to a craving level, indicative of learning a new instrumental conditioned reinforcement or PIT, for three reasons. First, PCA is procedurally dissociated and stable.[9]. Finally, STs and GTs do not differ in the topography of the CR, indicating the extent to which animals fail to learn either a ST or GT CR (Fig. 8; [9,25]; Fig. 9).
Absent model?

Figure 3 of training.

was recorded in the core of the nucleus accumbens using FSCV across six days of training. Parenthetically, bHR rats responded similarly to the bLR saline group (P > 0.05) and for rats blocked the performance of both sign-tracking and goal-tracking CRs. To demonstrate that animals demonstrate that animals and performance of sign-tracking behaviour. This work demonstrates that animals and dopamine-specific lesions and dopamine neurotransmission versus the unique pattern of dopamine release: Dopamine is necessary for learning CS–US associations that lead to sign-tracking, but not goal-tracking. a

\[ \delta = r - Q \]

Flagel et al. 2011, Nature
**Absent model?**

Sign trackers

Goal trackers

\[ \delta = r - Q \]

Flagel et al. 2011, Nature
Shift towards model-free in addiction

A reinforcement

B model-based

C data

B Controls
Freq F(1,8)=0.08; p=0.788
Rew F(1,8)=13.63; p=0.006
FxR F(1,8)=11.37; p=0.01

C Patients
Freq F(1,6)=0.42; p=0.542
Rew F(1,6)=7.56; p=0.033
FxR F(1,6)=0.61; p=0.463
Conclusion I

- Long-term rewards have internal consistency
- This can be exploited for solution
- Exploration and exploitation trade off when sampling
- Clever use of samples can produce fast learning
  - Brain most likely does something like this
- Model-free learning relies on phasic dopamine signals
  - iterative learning from experience
  - individual variability & addiction