Intelligent swimmer

Anthony Ge

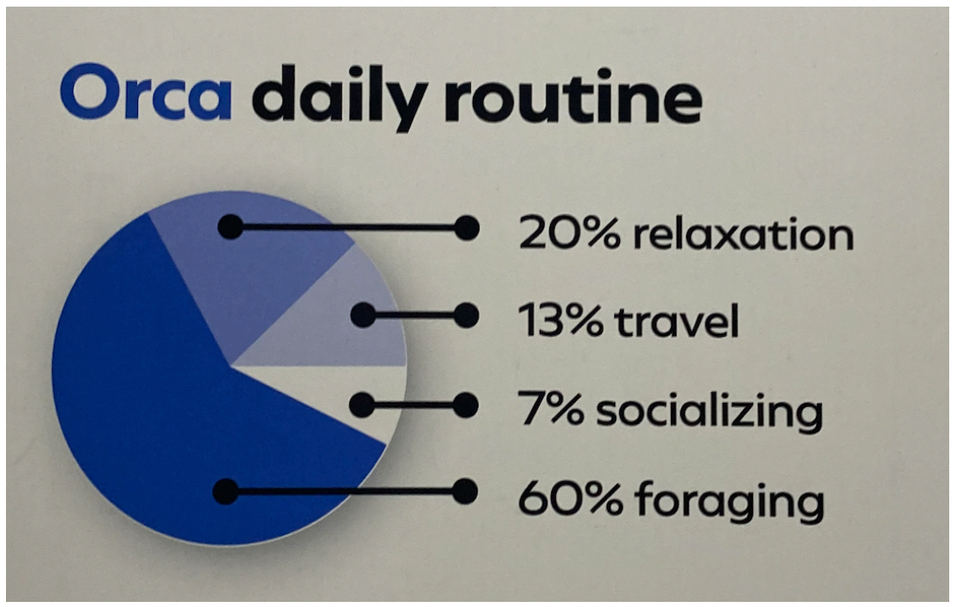
PHYS 350, UBC

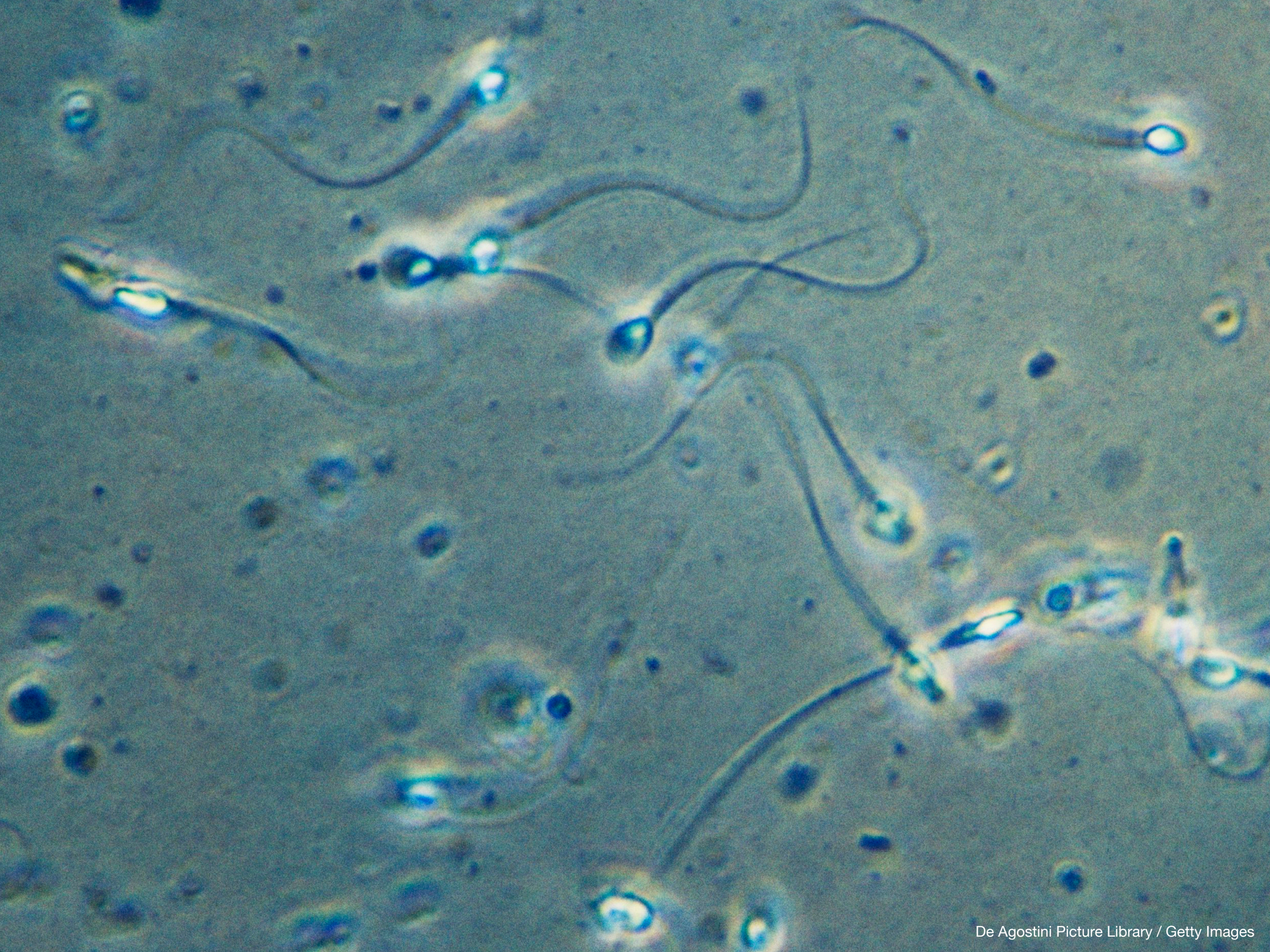
April 4, 2022

THE HUNDRED-YEAR-OLD MATRIARCH

Granny, the longtime leader of J pod, was the oldest known orca in the world. Researchers estimated that she was born in 1911. When she was named, scientists knew Granny was the matriarch of J pod and the mother, grandmother and great-grandmother of many orcas. She was also the knowledge keeper, teaching generations where, what and how to hunt. After Granny died in 2017, J pod quickly lost several members, possibly because the matriarch was no longer there to guide them. As of 2019, J pod's new matriarch is the 47-year-old Slick (J16).

Even in the last years of her life, Granny (J2) would have led younger relatives in search of food. Photograph courtesy of the Center for Whale Research.

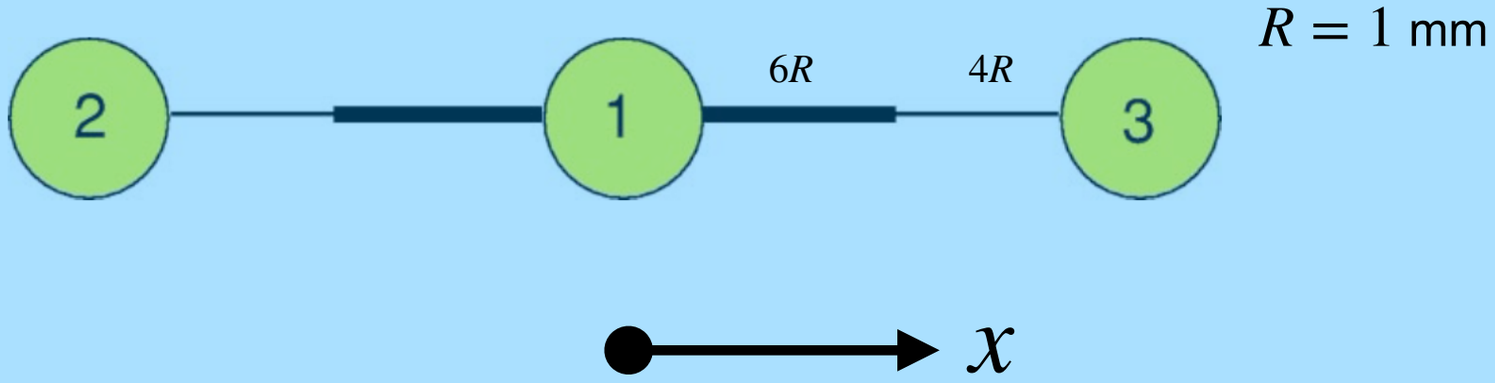




De Agostini Picture Library / Getty Images

Two-link swimmer *

* Najafi & Golestanian, Phys. Rev. E (2004)



Objective: Identify the *optimal* moves to swim forward.

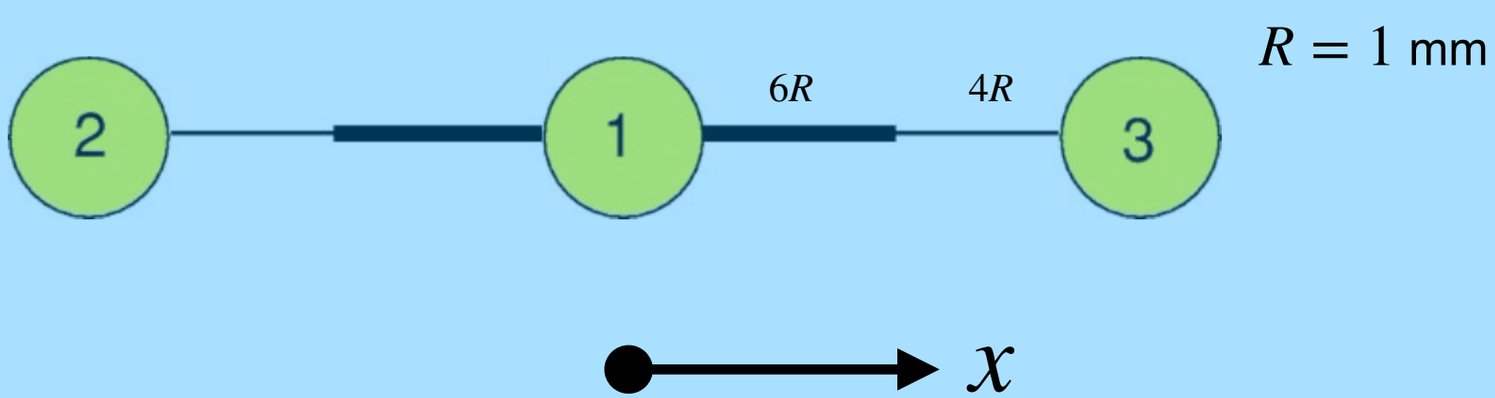
Life at the *mm* scale

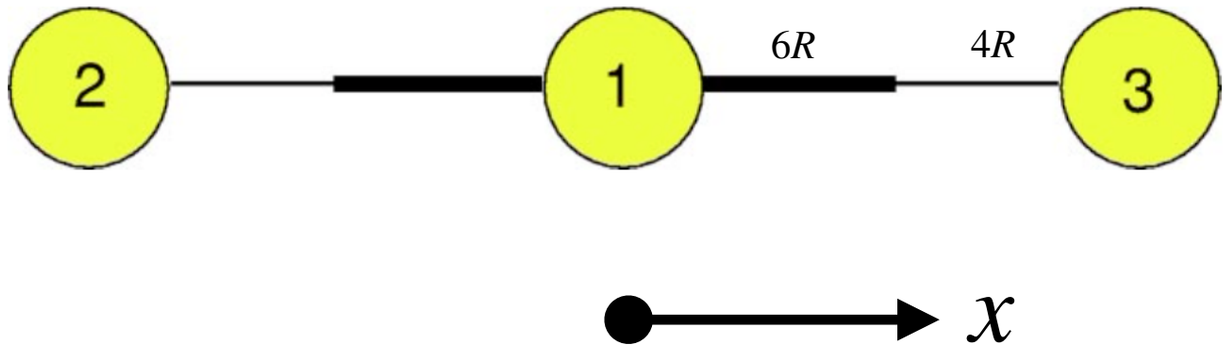
- No inertia
  - i.e. stop moving, stop swimming

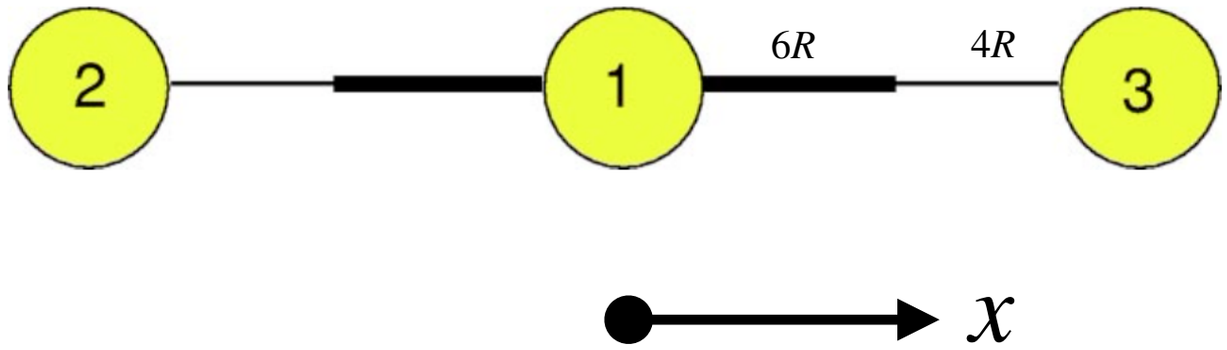
- Linearity
  - $U \propto F$

- Time reversal symmetry
  - Reversing a move will undo its effect

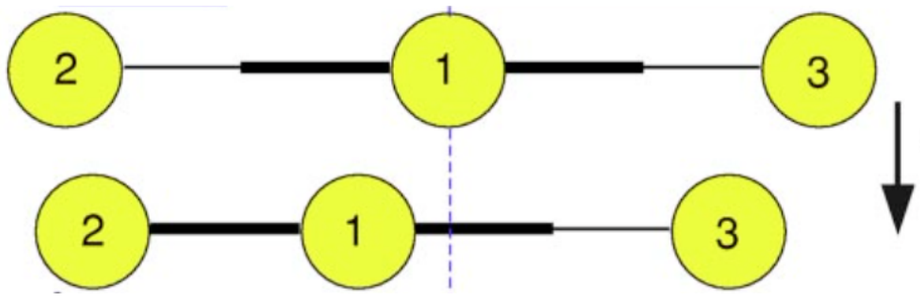
- Mirror symmetry (in our case)




2
1
3
6R
4R
x

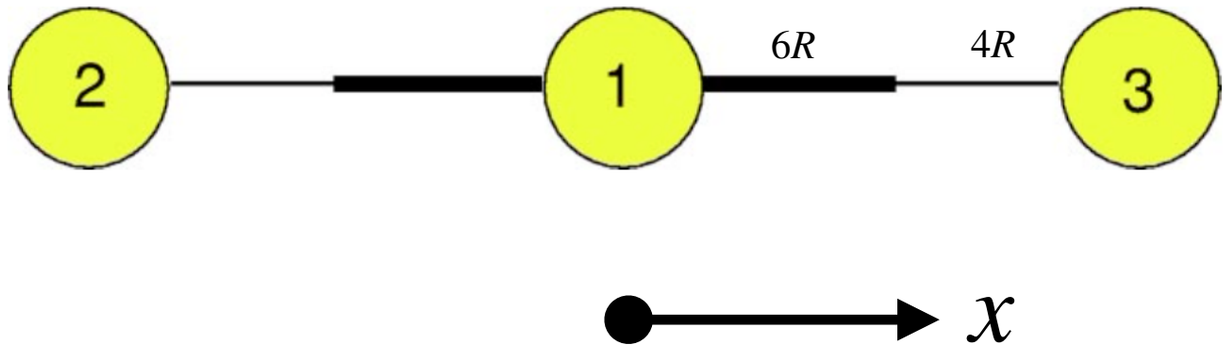


Move (a)

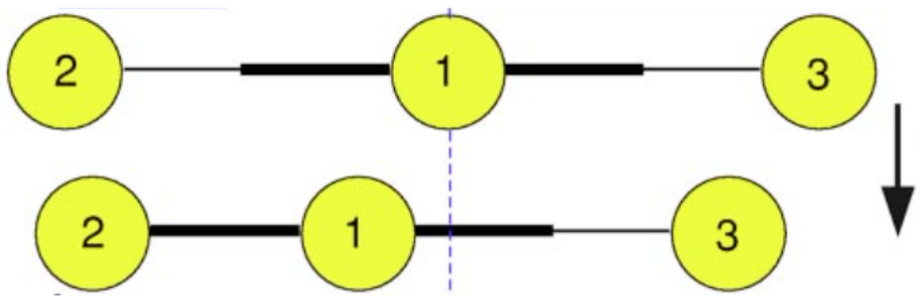


$$\Delta x_1 = -1.35R$$

$$\Delta x_c = \Delta x_1 + 4/3R = -0.02R$$



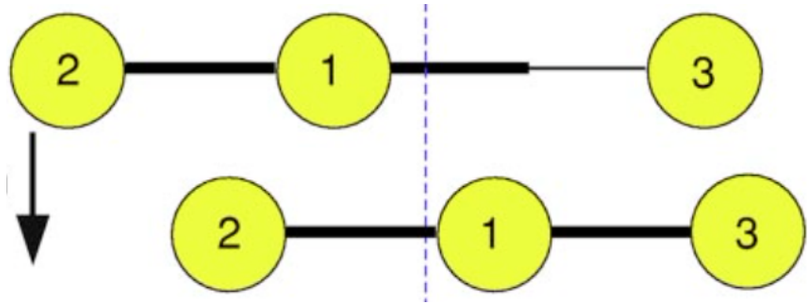
Move (a)



$$\Delta x_1 = -1.35R$$

$$\Delta x_c = \Delta x_1 + 4/3R = -0.02R$$

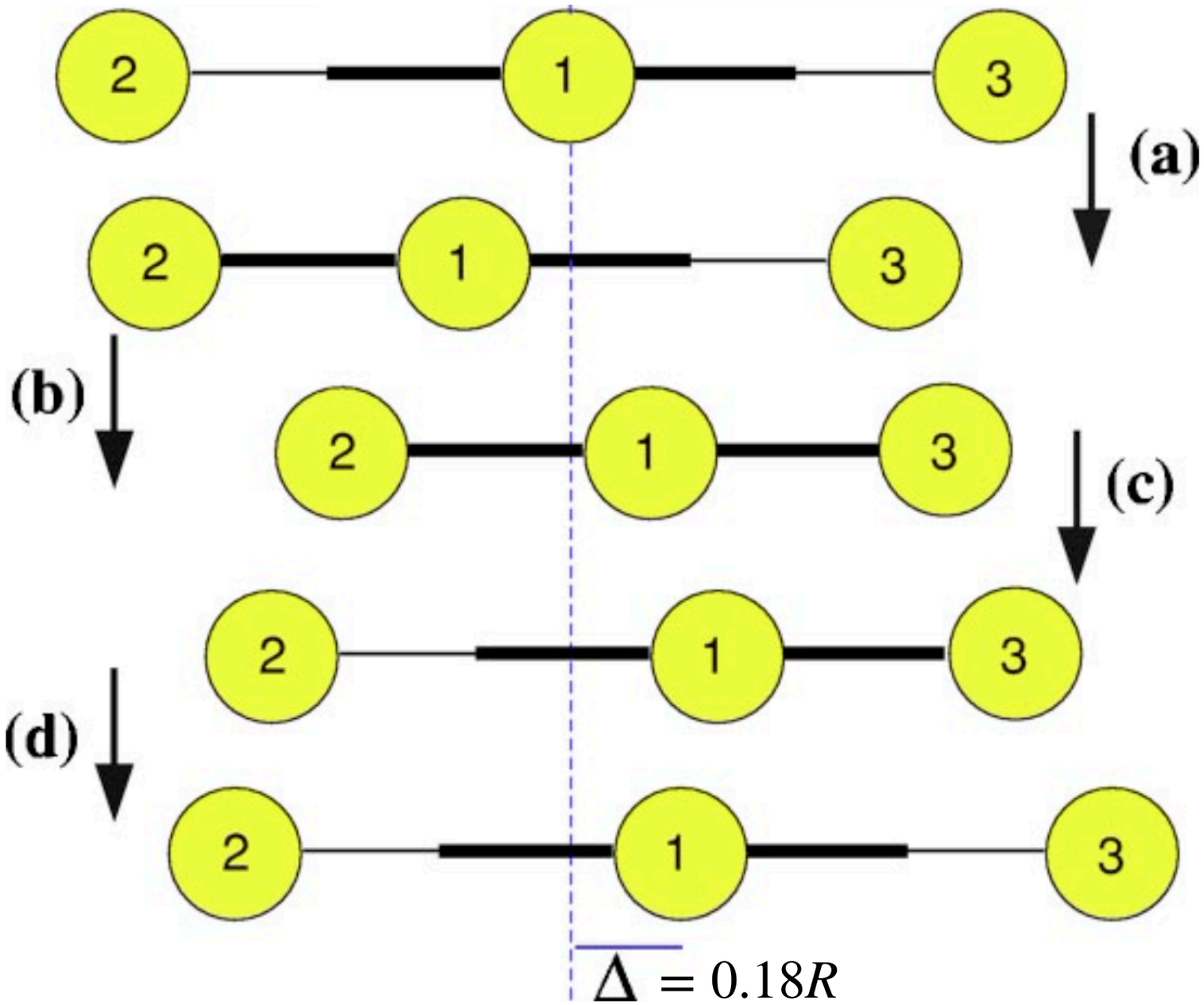
Move (b)



$$\Delta x_1 = 1.44R$$

$$\Delta x_c = \Delta x_1 - 4/3R = 0.11R$$

Optimal sequence (a travelling wave)



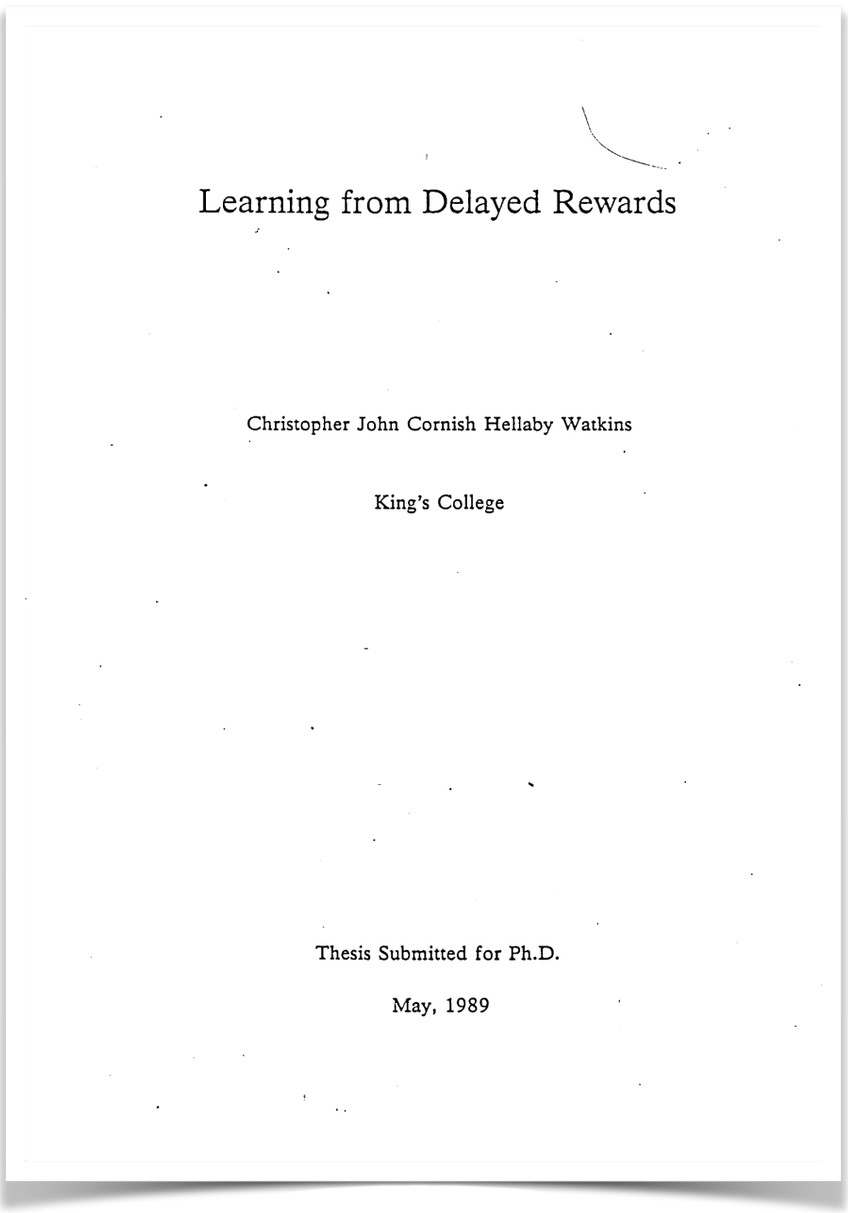
Is it possible to learn swimming *without* prior knowledge of physics?

Is it possible to learn swimming *without* prior knowledge of physics?

What underlies *intelligence*?

“Learning to act in ways that are rewarded is a sign of intelligence.”

Christopher Watkins



Learning from Delayed Rewards

Christopher John Cornish Hellaby Watkins

King's College

Thesis Submitted for Ph.D.

May, 1989

- Reinforcement learning
  - Q-learning
- Multistage decision processes
  - Dynamic programming

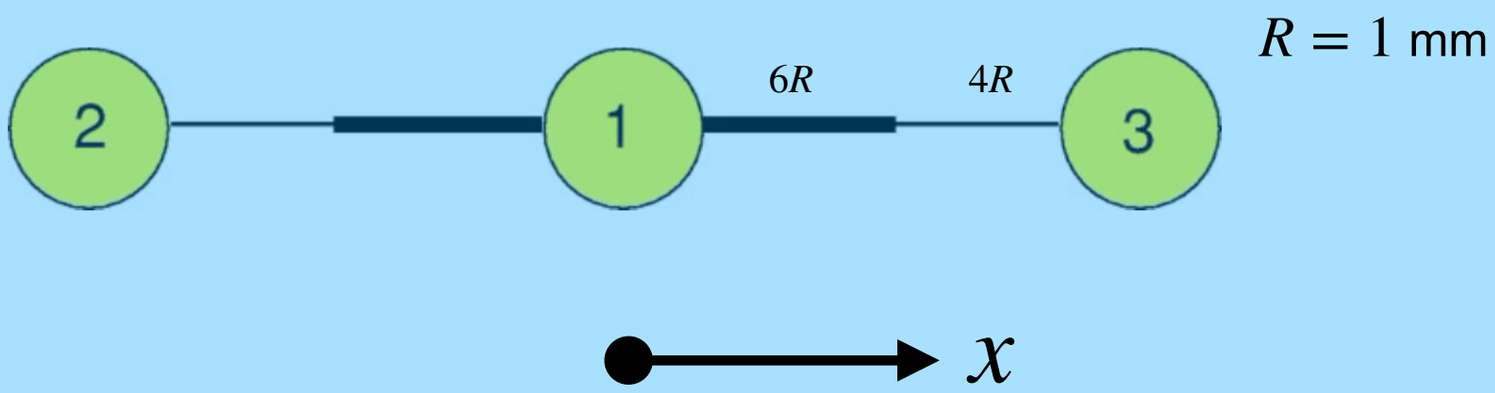
Optimization

Learning to swim

- Register the environment (requires sensing)
  - i.e. the current state

$$x_n \in S$$

Time step $n = 0,1,2,3,\ldots$



Learning to swim

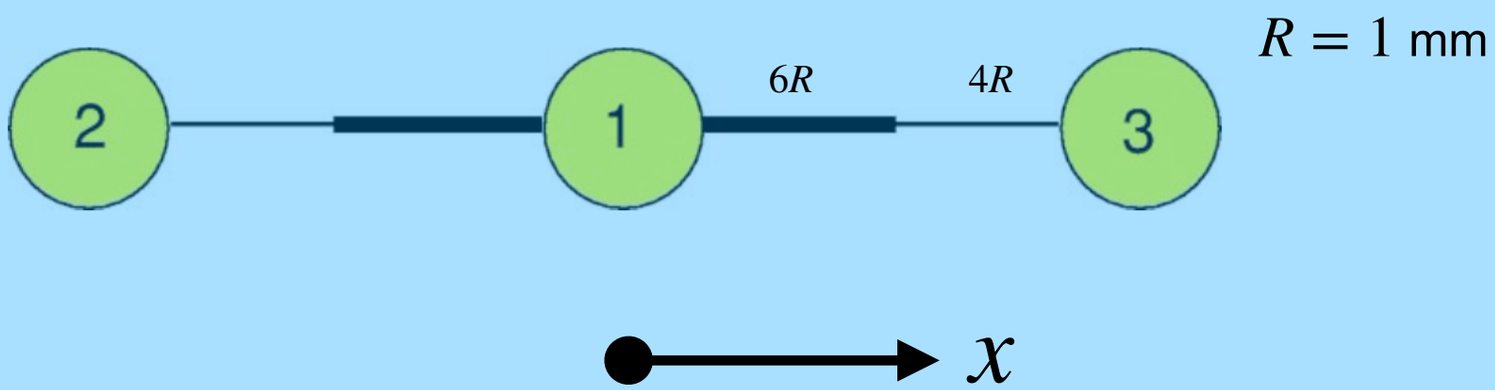
- Register the environment (requires sensing)
  - i.e. the current state

$$x_n \in S$$

Time step $n = 0,1,2,3,\ldots$

- Follow a policy (requires memory)
  - i.e. state → action

$$a_n(x_n) \in A$$



Learning to swim

- Register the environment (requires sensing)
  - i.e. the current state

$x_n \in S$

Time step $n = 0,1,2,3,\ldots$

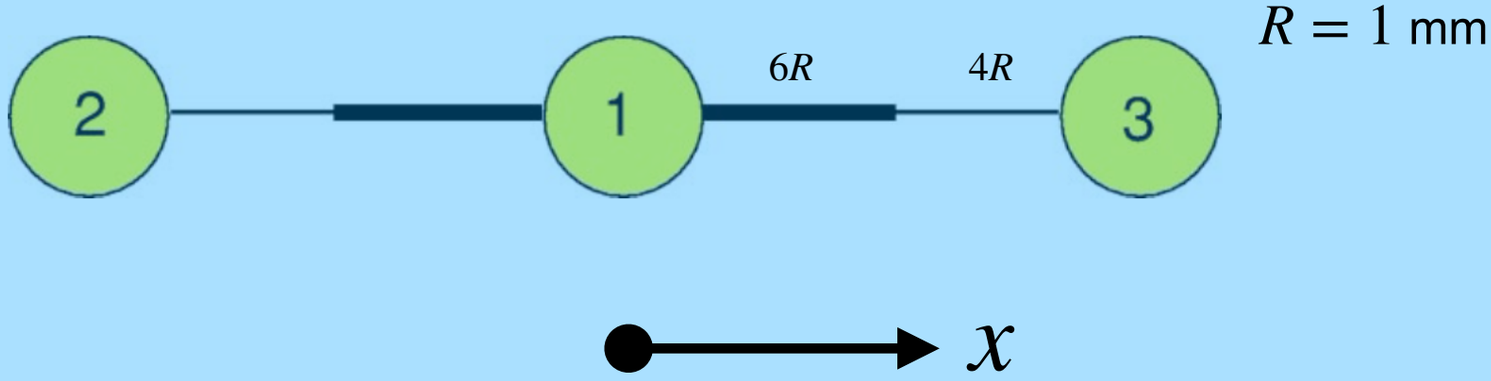
- Follow a policy (requires memory)
  - i.e. state → action

$a_n(x_n) \in A$

- Improve the policy (requires intelligence)
  - by maximizing the reward
  - both immediate and long-term

$R_n + \gamma R_{n+1} + \gamma^2 R_{n+2} + \ldots$

Discount factor $0 < \gamma < 1$



Policy improvement theorem

Define an *evaluation function* of the state under policy f

$$V_f(x_n) = R_n + \gamma R_{n+1} + \gamma^2 R_{n+2} + \ldots,$$

or

$$V_f(x_n) = R_n + \gamma V_f(x_{n+1})\,.$$

If there is another policy g such that $V_g(x) \geqslant V_f(x), \forall x \in S$, then policy g is uniformly better than policy f.

Policy improvement theorem

Define an *evaluation function* of the state under policy f

$$V_f(x_n) = R_n + \gamma R_{n+1} + \gamma^2 R_{n+2} + \ldots,$$

or

$$V_f(x_n) = R_n + \gamma V_f(x_{n+1})\,.$$

If there is another policy g such that $V_g(x) \geqslant V_f(x), \forall x \in S$, then policy g is uniformly better than policy f.

Now, if we follow policy g for only one step at the current state, and switch back to policy f subsequently, we may define an *action-value function*

$$Q_f(x_n, g) = R_n\big|_g + \gamma V_f(x_{n+1})\,.$$

If $Q_f(x, g) \geqslant V_f(x), \forall x \in S$, then $V_g(x) \geqslant V_f(x), \forall x \in S$.

Policy improvement theorem

↓

Bellman, *Dynamic programming* (1957).

Policy improvement algorithm

- Start from any policy, improve the policy by taking the best action, guaranteed to reach the optimal policy
- Evaluation function (V) calculated many times (costly)

Policy improvement theorem

↓

Bellman, *Dynamic programming* (1957).

Policy improvement algorithm

- Start from any policy, improve the policy by taking the best action, guaranteed to reach the optimal policy
- Evaluation function (V) calculated many times (costly)

↓

...

↓

Q-learning

Watkins, PhD thesis (1989).

- Start from scratch, iteratively learn the policy via local improvements, guaranteed to converge to the optimum policy (under certain conditions)
- Work with Q instead of V (more efficient computationally)

Learning to swim via Q-learning

In the nth step, the swimmer

1. Observes its current state x_n

Watkins & Dayan, *Machine learning* (1992).
Sutton & Barto, Reinforcement learning: An introduction (2018).

Learning to swim via Q-learning

In the nth step, the swimmer

1. Observes its current state x_n
2. Performs an action a_n under the current policy

But can explore a random move with probability ϵ

Watkins & Dayan, *Machine learning* (1992).
Sutton & Barto, Reinforcement learning: An introduction (2018).

Learning to swim via Q-learning

In the nth step, the swimmer

1. Observes its current state x_n
2. Performs an action a_n under the current policy
3. Observes the subsequent state x_{n+1}

But can explore a random move with probability ϵ

Watkins & Dayan, *Machine learning* (1992).
Sutton & Barto, Reinforcement learning: An introduction (2018).

Learning to swim via Q-learning

In the nth step, the swimmer

1. Observes its current state x_n
2. Performs an action a_n under the current policy
3. Observes the subsequent state x_{n+1}
4. Receives an immediate payoff r_n

But can explore a random move with probability ϵ

Watkins & Dayan, *Machine learning* (1992).
Sutton & Barto, Reinforcement learning: An introduction (2018).

Learning to swim via Q-learning

In the nth step, the swimmer

1. Observes its current state x_n
2. Performs an action a_n under the current policy
3. Observes the subsequent state x_{n+1}
4. Receives an immediate payoff r_n
5. Updates the policy using a learning factor $0 < \alpha \leqslant 1$, according to

$$Q_{n+1}(x_n, a_n) = Q_n(x_n, a_n) + \alpha\left(r_n + \gamma \max_{a_{n+1}} Q_n(x_{n+1}, a_{n+1}) - Q_n(x_n, a_n) \right).$$

But can explore a random move with probability ϵ

Watkins & Dayan, *Machine learning* (1992).
Sutton & Barto, Reinforcement learning: An introduction (2018).

Learning to swim via Q-learning

In the nth step, the swimmer

1. Observes its current state x_n
2. Performs an action a_n under the current policy
3. Observes the subsequent state x_{n+1}
4. Receives an immediate payoff r_n
5. Updates the policy using a learning factor $0 < \alpha \leqslant 1$, according to

But can explore a random move with probability ϵ

$$Q_{n+1}(x_n, a_n) = Q_n(x_n, a_n) + \alpha\left(r_n + \gamma \max_{a_{n+1}} Q_n(x_{n+1}, a_{n+1}) - Q_n(x_n, a_n) \right).$$

Bellman optimality equation $Q_* = r_n + \gamma \max Q_* \Leftrightarrow$ **Steady state** $Q_n \to Q_{n+1} \to Q_*$

Watkins & Dayan, *Machine learning* (1992).
Sutton & Barto, Reinforcement learning: An introduction (2018).

Learning to swim via Q-learning

In the nth step, the swimmer

1. Observes its current state x_n
2. Performs an action a_n under the current policy
3. Observes the subsequent state x_{n+1}
4. Receives an immediate payoff r_n
5. Updates the policy using a learning factor $0 < \alpha \leqslant 1$, according to

But can explore a random move with probability ϵ

$$Q_{n+1}(x_n, a_n) = Q_n(x_n, a_n) + \alpha\left(r_n + \gamma \max_{a_{n+1}} Q_n(x_{n+1}, a_{n+1}) - Q_n(x_n, a_n) \right).$$

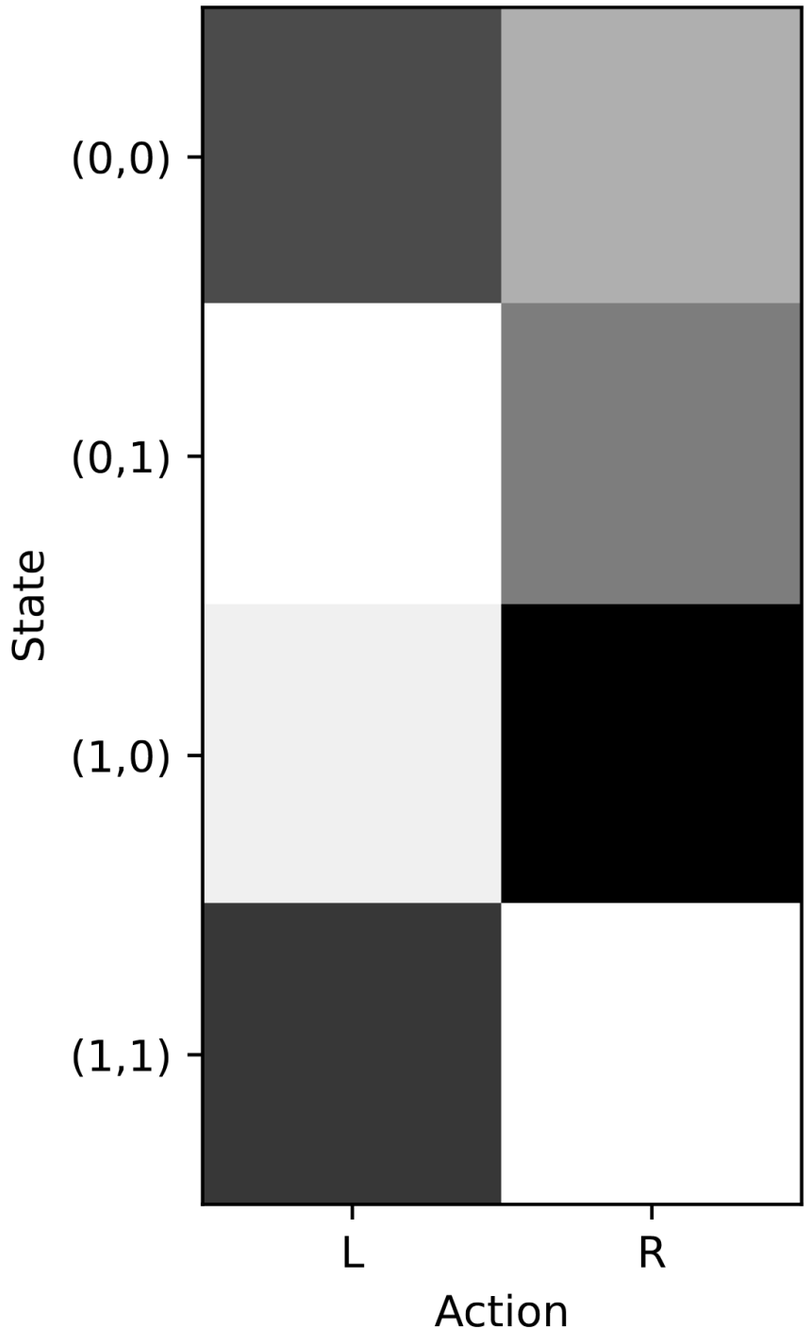
Bellman optimality equation $Q_* = r_n + \gamma \max Q_*$ $\Leftrightarrow$ **Steady state** $Q_n \to Q_{n+1} \to Q_*$

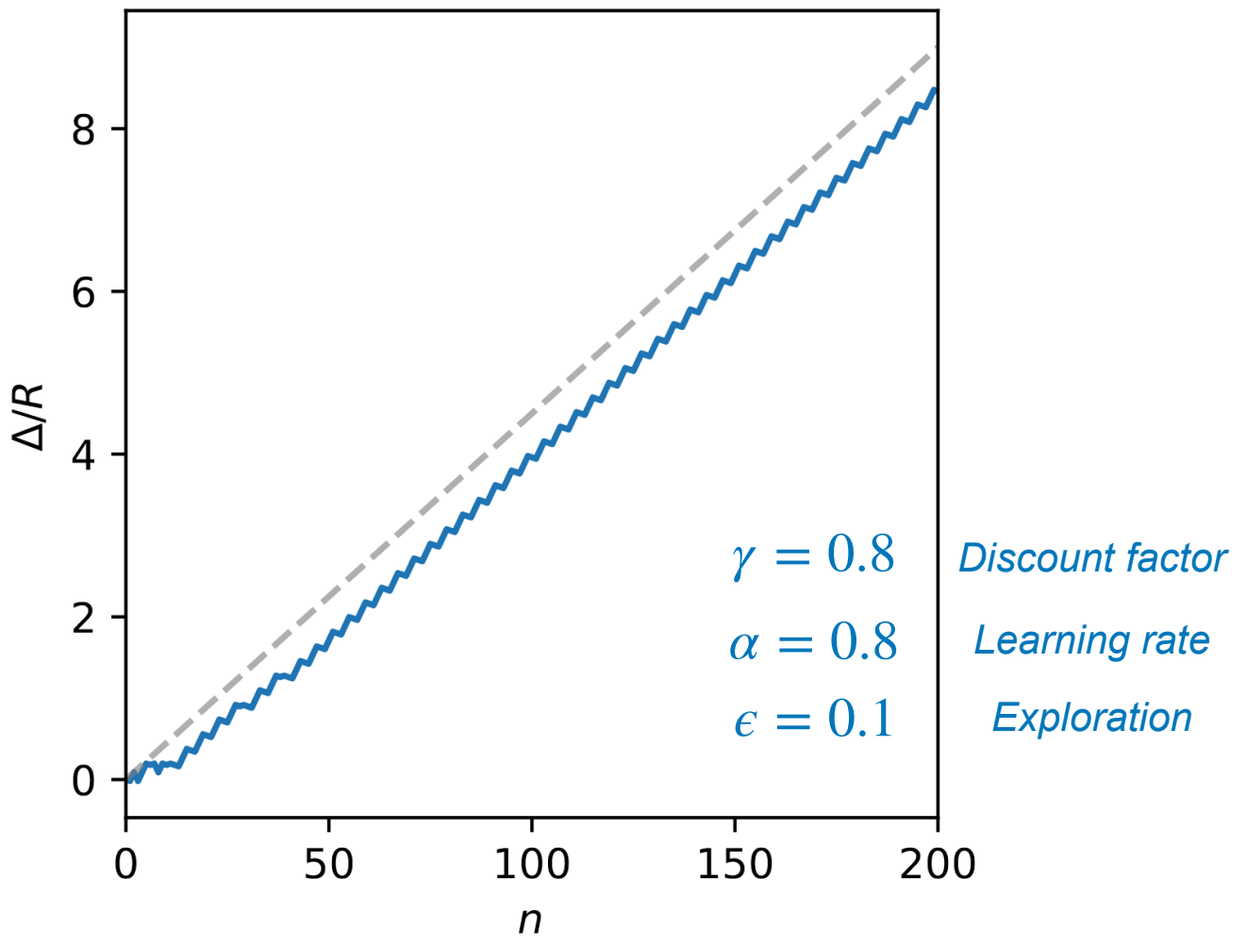
Three model parameters: γ, α, ϵ.

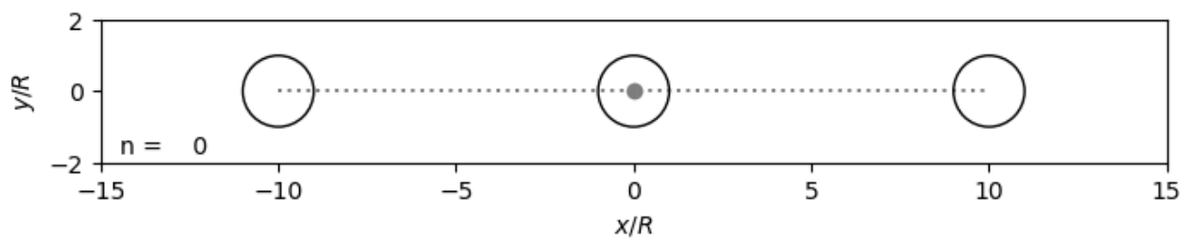
Watkins & Dayan, *Machine learning* (1992).
Sutton & Barto, Reinforcement learning: An introduction (2018).

Does it work?

https://github.com/GeZhouyang/two-link-swimmer







Think…

- How do the parameters affect the performance? Why?
- Can you propose a more difficult task?

Think…

- How do the parameters affect the performance? Why?
- Can you propose a more difficult task?

Take-home messages

- Failure and repeated practice are necessary for learning.
  - Don’t give up!
- Exploration is important when faced with novelty.
  - Be open-minded!
- “Learning from delayed rewards” — foresight is essential.

zhoge@mail.ubc.ca

Think…

- How do the parameters affect the performance? Why?
- Can you propose a more difficult task?

Take-home messages

- Failure and repeated practice are necessary for learning.
  - Don’t give up!
- Exploration is important when faced with novelty.
  - Be open-minded!
- “Learning from delayed rewards” — foresight is essential.
  - What’s the ultimately delayed reward? *Learning* itself.

zhoge@mail.ubc.ca