Cogmaster
Computational Neuroscience Methods

February 14th 2020

Rescorla-Wagner rule

Manuel Beiran
manuel.beiran@ens.fr

Remarks on Ex. 1:

Remarks on Ex. 1:

- your model (1): $$p_n = (\alpha + 1) p_{n-1}$$

a **map**: discrete-time evolution in the form: $x_n = A(x_{n-1})$

linear map

behavior determined by the value of $\alpha + 1$

- exponential positive growth $\alpha > 0$
- decay to 0 $-1 < \alpha < 0$
- oscillations $\alpha < -1$

Remarks on Ex. 1:

- your model (2): $$p_n = p_{n-1} + \overset{\beta}{0.001}\, p_{n-1}(\overset{\alpha}{200} - p_{n-1})$$
- the **logistic map**: $$x_n = r x_{n-1}(1 - x_{n-1})$$

$$r = 1 + \beta\alpha$$

Remarks on Ex. 1:

- your model (2): $$p_n = p_{n-1} + 0.001 p_{n-1}(200 - p_{n-1})$$
- the **logistic map**: $$x_n = r x_{n-1}(1 - x_{n-1})$$

Fixed point of the map: no time evolution, dynamics converges

$$x_n = A(x_{n-1}) = x_{n-1}$$

[Strogatz, *Non-linear dynamics and chaos*]

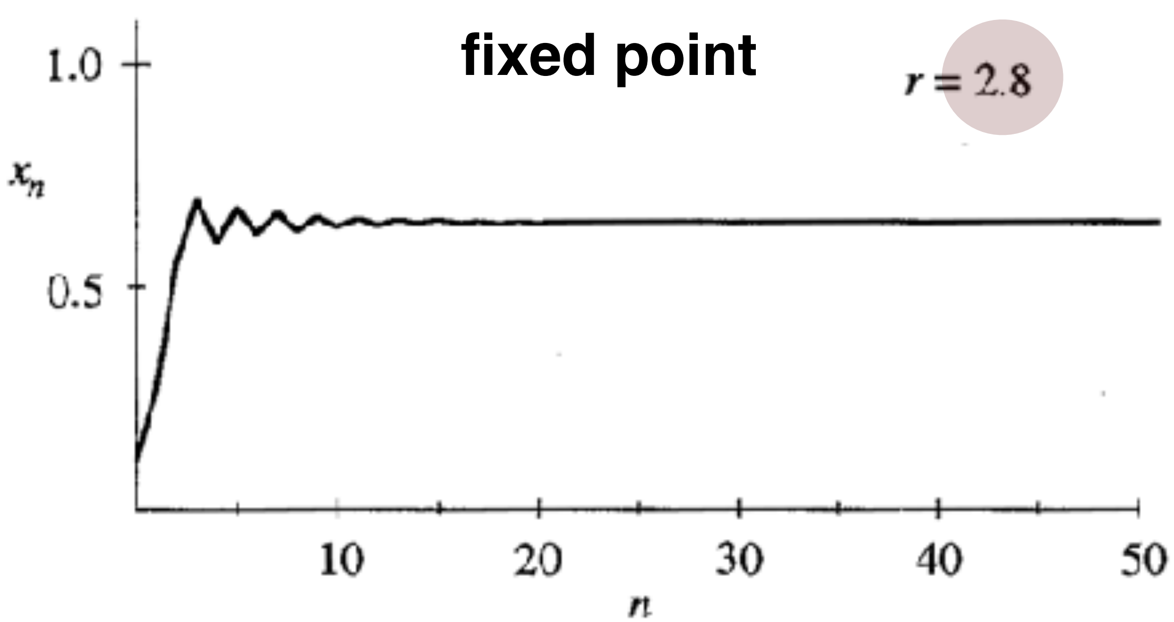
Remarks on Ex. 1:

- your model (2): $p_n = p_{n-1} + 0.001 p_{n-1}(200 - p_{n-1})$
- the **logistic map**: $x_n = r x_{n-1}(1 - x_{n-1})$

Fixed point of the map: no time evolution, dynamics converges

$$x_n = A(x_{n-1}) = x_{n-1}$$

fixed point

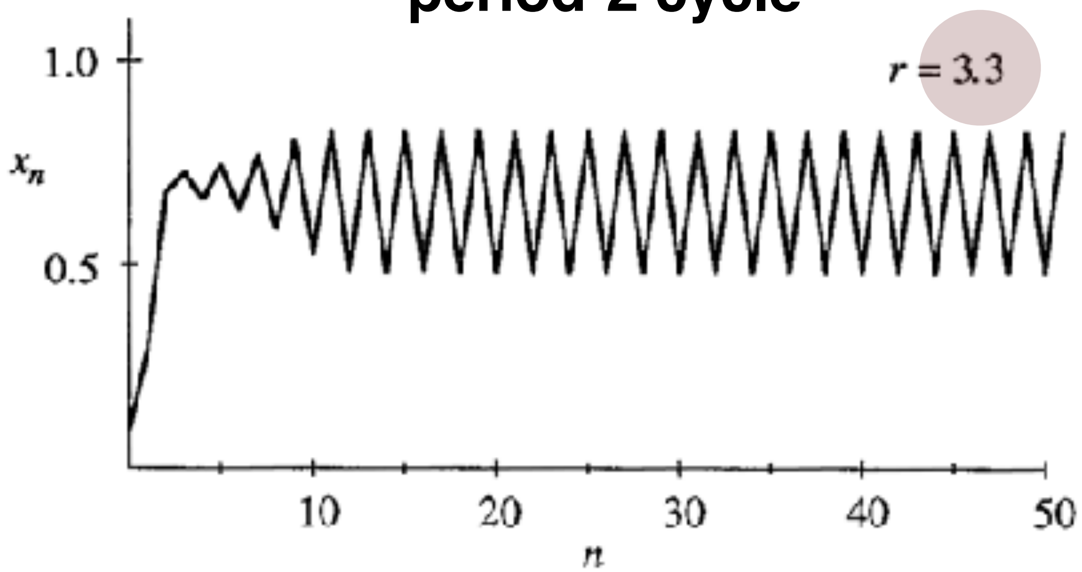


[Strogatz, *Non-linear dynamics and chaos*]

[May, RM (1976) *Simple mathematical models with very complicated dynamics*]

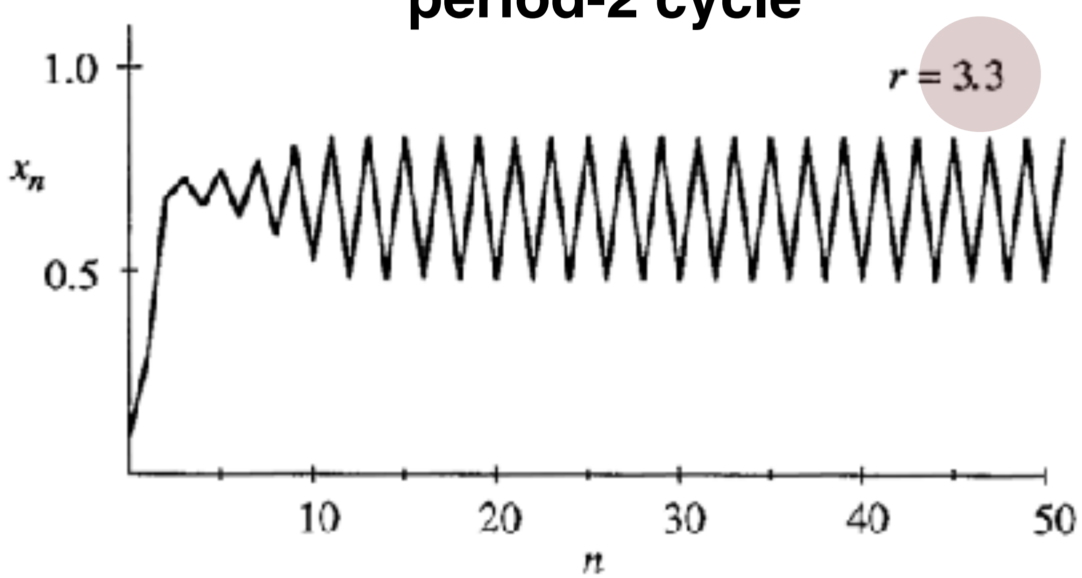
- the **logistic map**: $$x_n = r x_{n-1}(1 - x_{n-1})$$

period-2 cycle

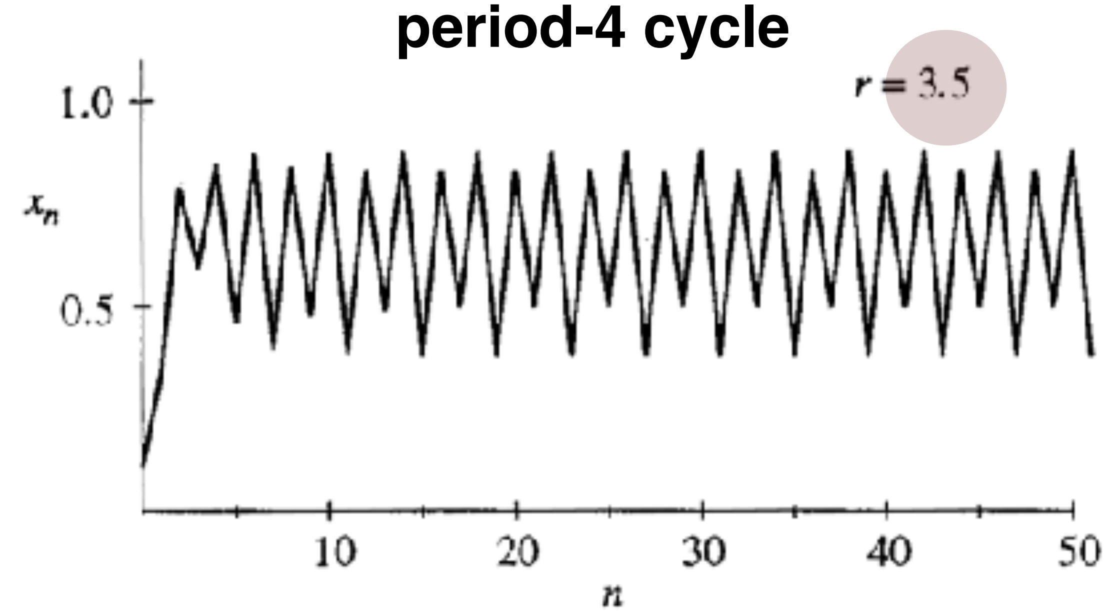


- the **logistic map**: $$x_n = r x_{n-1}(1 - x_{n-1})$$

period-2 cycle

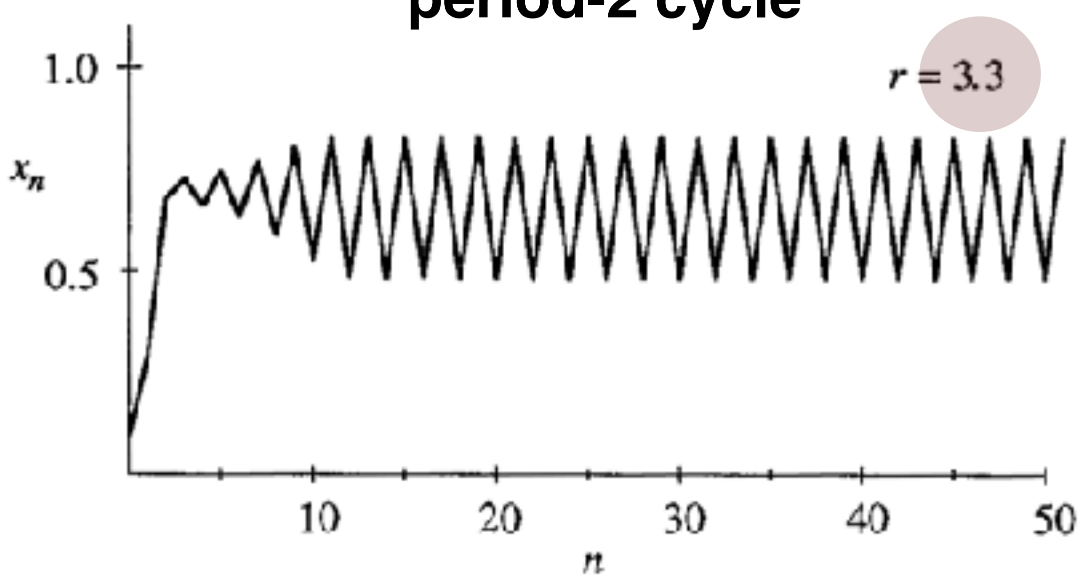


period-4 cycle

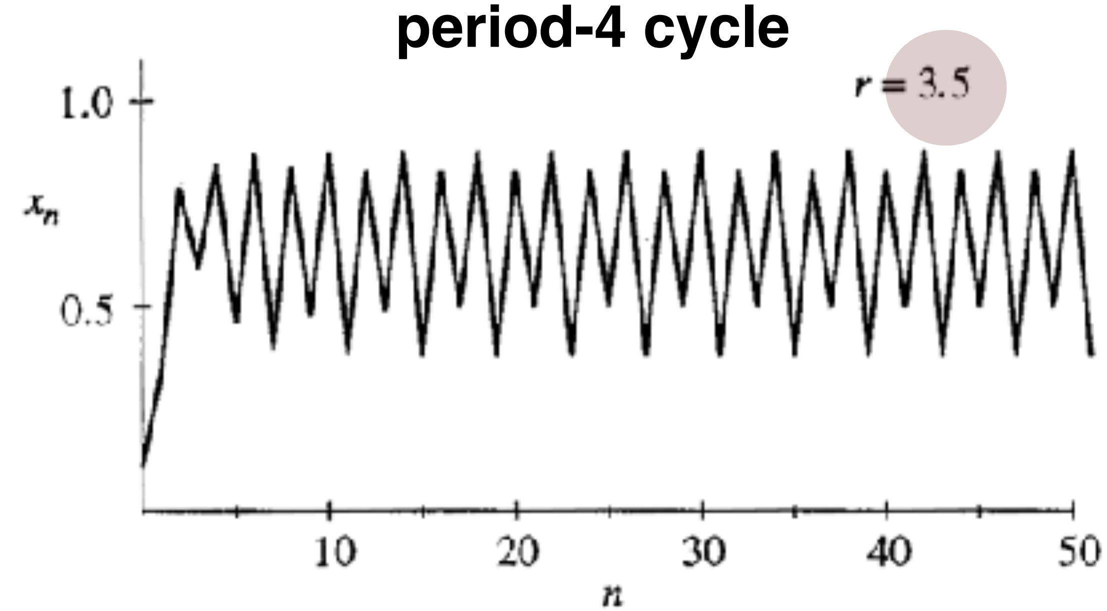


- the **logistic map**: $$x_n = rx_{n-1}(1 - x_{n-1})$$

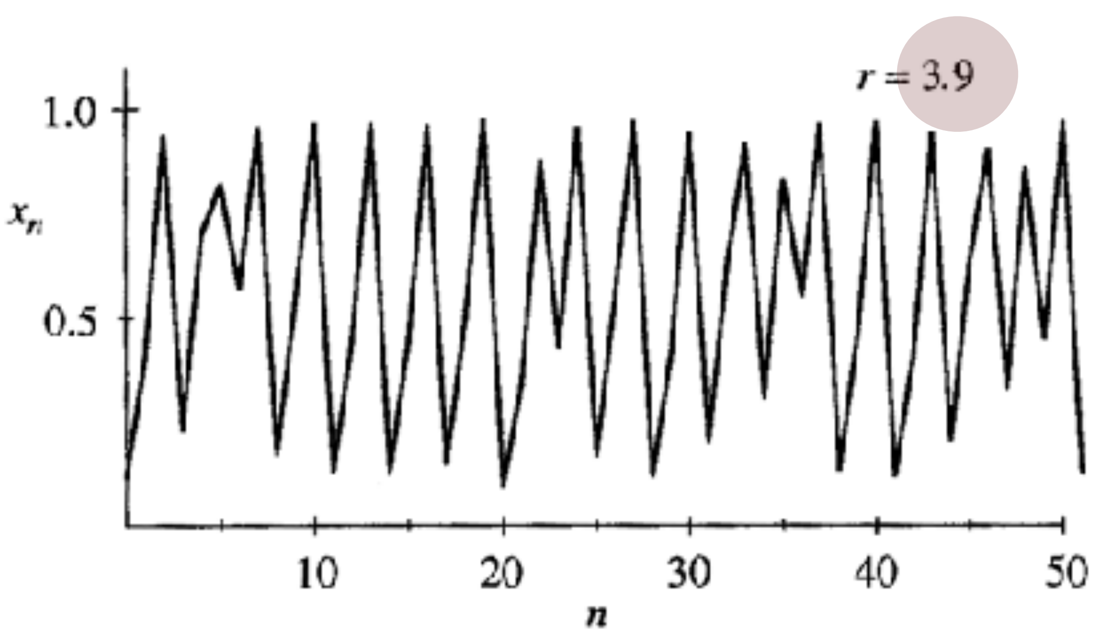
period-2 cycle



period-4 cycle

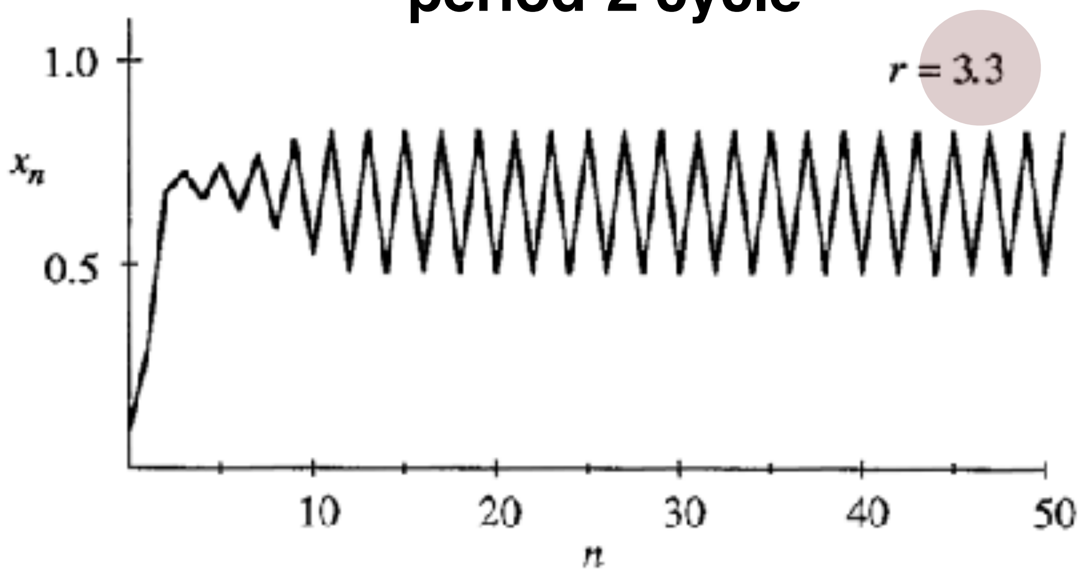


period-doubling bifurcation to chaos

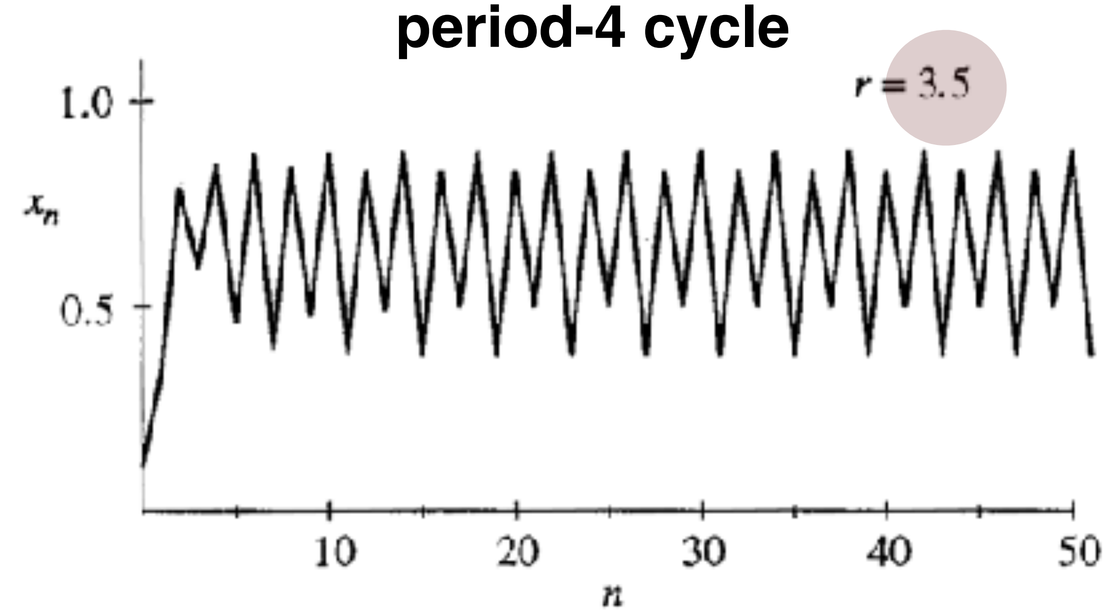


- the **logistic map**: $$x_n = r x_{n-1}(1 - x_{n-1})$$

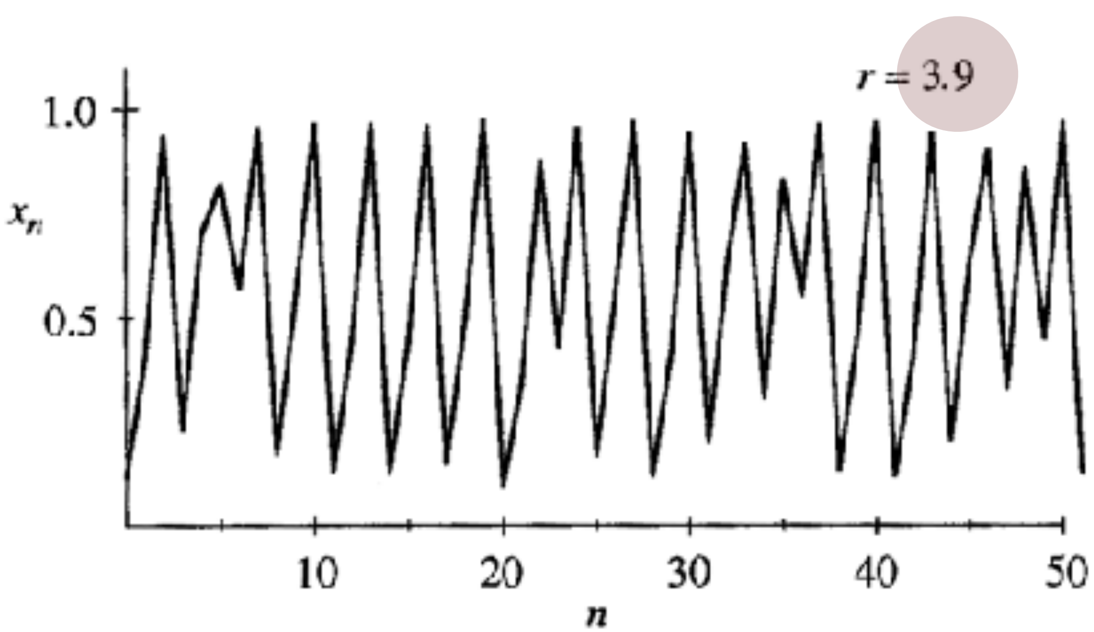
period-2 cycle



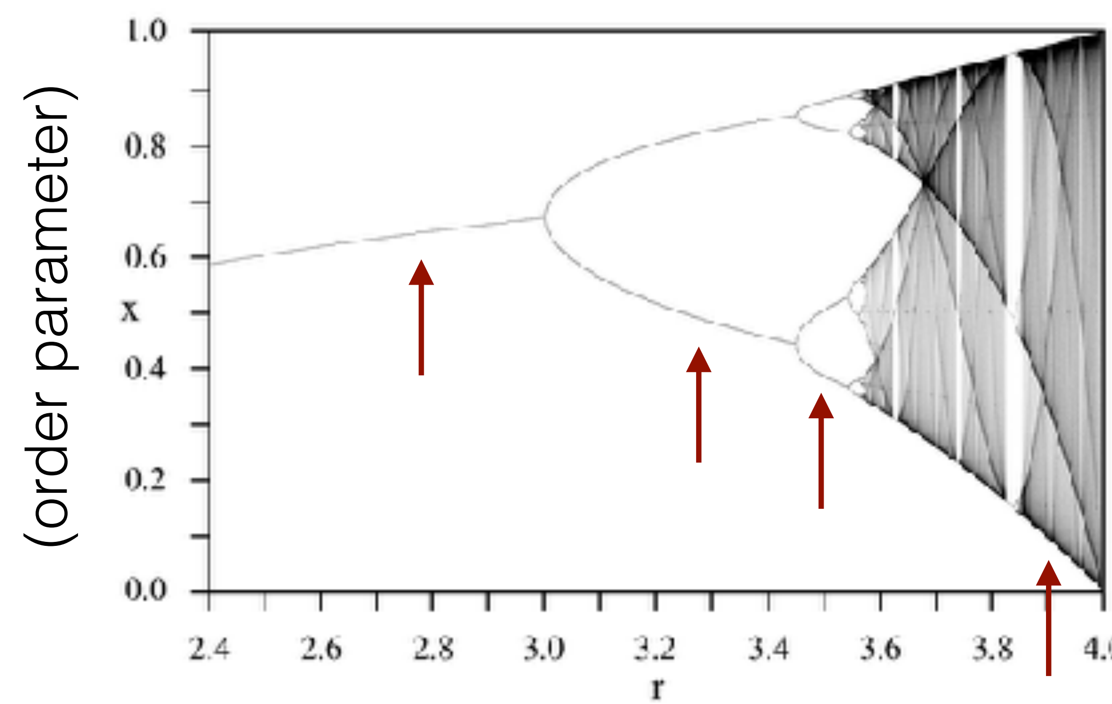
period-4 cycle



period-doubling bifurcation to chaos



bifurcation diagram:



Ex. 2: Computational models of behaviour

see:

Dayan and Abbott, *Theoretical Neuroscience*, 9.1 - 9.2
C06 course

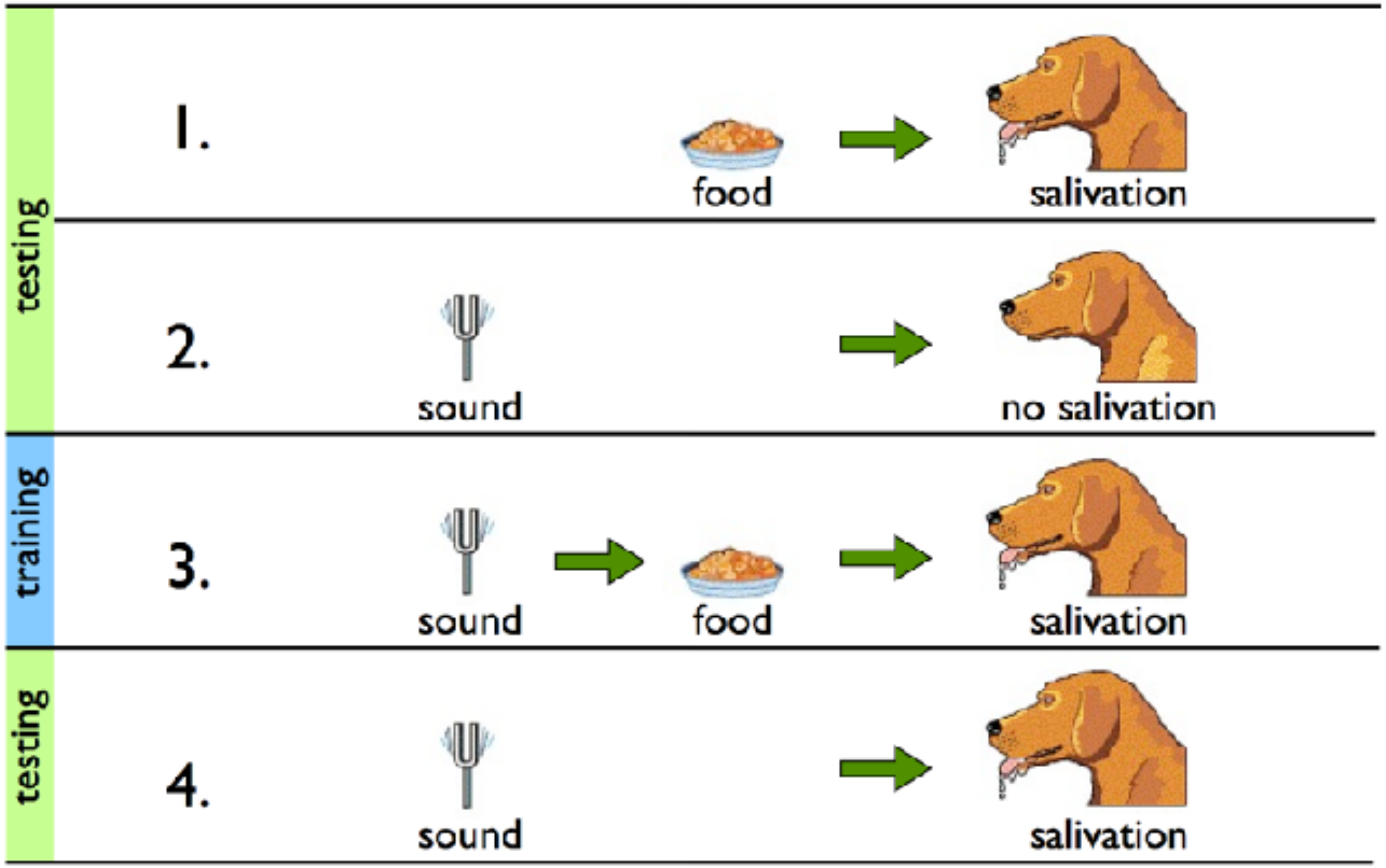
Study the ability of animals of taking actions according only to the received **reward** and **punishment:**

REINFORCEMENT LEARNING

Experiments of: - **classical** (Pavlovian) conditioning;
- **instrumental** conditioning

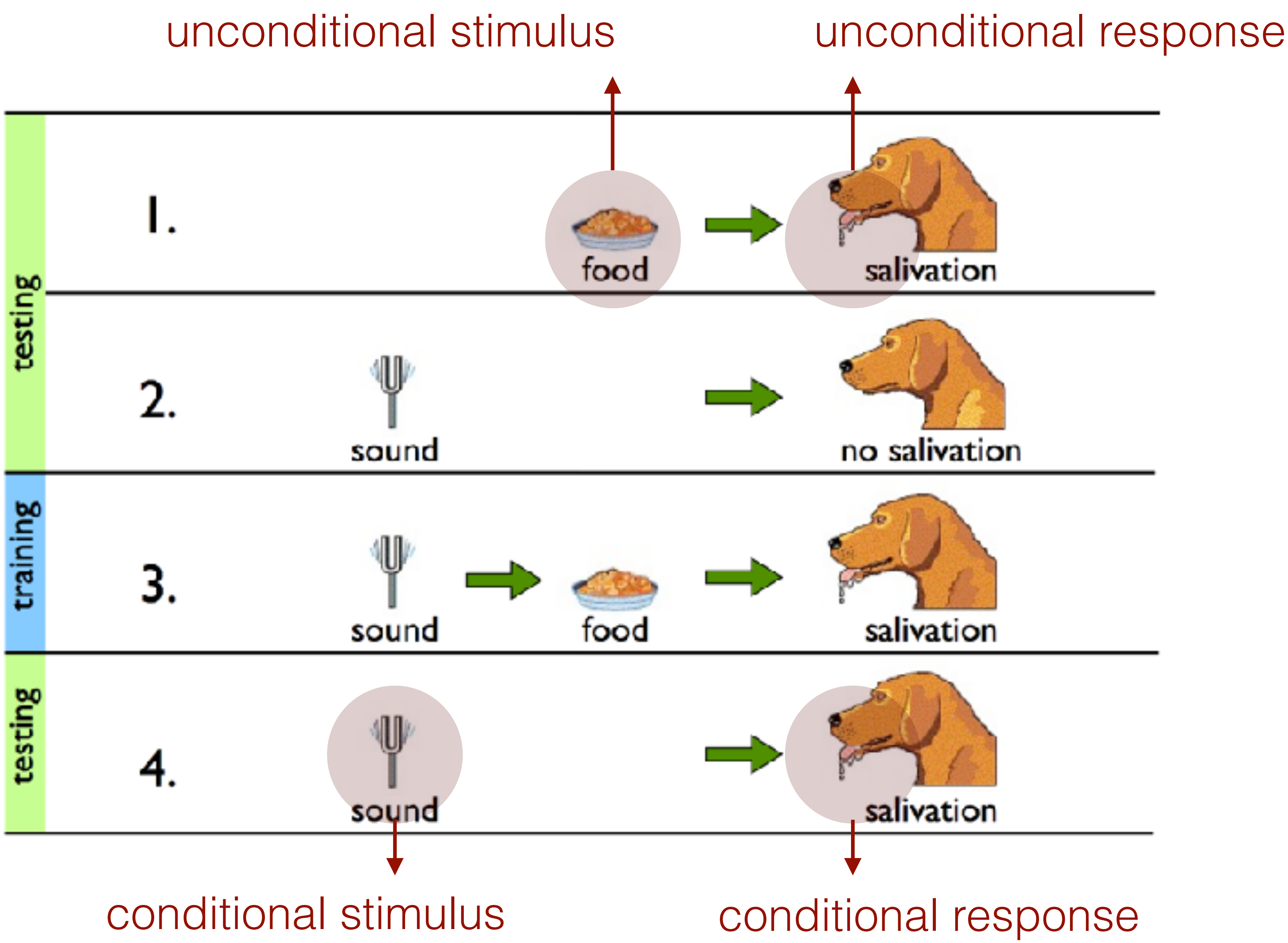
Ex. 2.1: Classical conditioning

Pavlovian experiment:

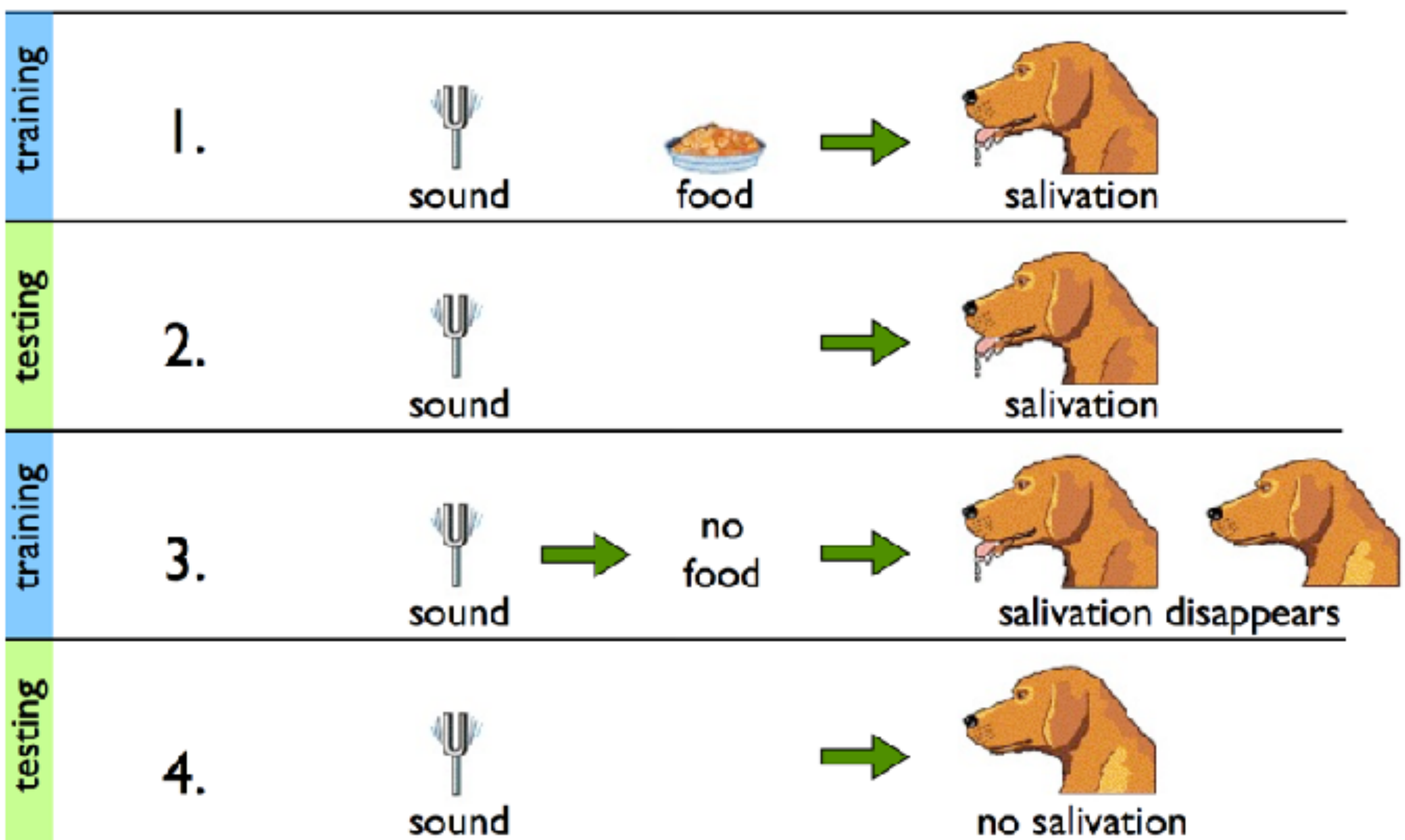


Ex. 2.1: Classical conditioning

Pavlovian experiment:



Extinction:


training
1.
sound
food
salivation
testing
2.
sound
salivation
training
3.
sound
no
food
salivation disappears
testing
4.
sound
no salivation

Computational model: **Rescorla-Wagner rule**

u_i stimulus ( ) in trial i: $u_i = 0$ or $u_i = 1$

r_i reward ( ) in trial i: $r_i = 0$ or $r_i = 1$

v_i reward that the dog expects ( ) in trial i

Robert Rescorla

Allan R Wagner

- the reward prediction is **linear** with the stimulus: $v = wu$

free parameter

- the animal wants to learn to **predict** the reward.

Measure the ability of the dog to predict the reward!
Prediction error in the i-th trial:

$$\delta_i = \underbrace{r_i}_{\text{actual reward}} - \underbrace{v_i}_{\text{predicted reward}}$$

$\delta_i > 0$: more reward than predicted
$\delta_i < 0$: less reward than predicted

"Loss" in the i-th trial:

$$L_i = \delta_i^2 = (r_i - v_i)^2$$

→ Minimize this loss function to maximize the ability to predict the reward !!

Gradient descent minimization:

"Loss" in the i-th trial:

$$L_i = (r_i - wu_i)^2$$

Update parameter w to decrease loss!

$$w \rightarrow w - \epsilon \frac{d}{dw} L_i \qquad \frac{d}{dw} L_i = -2u_i\delta_i$$

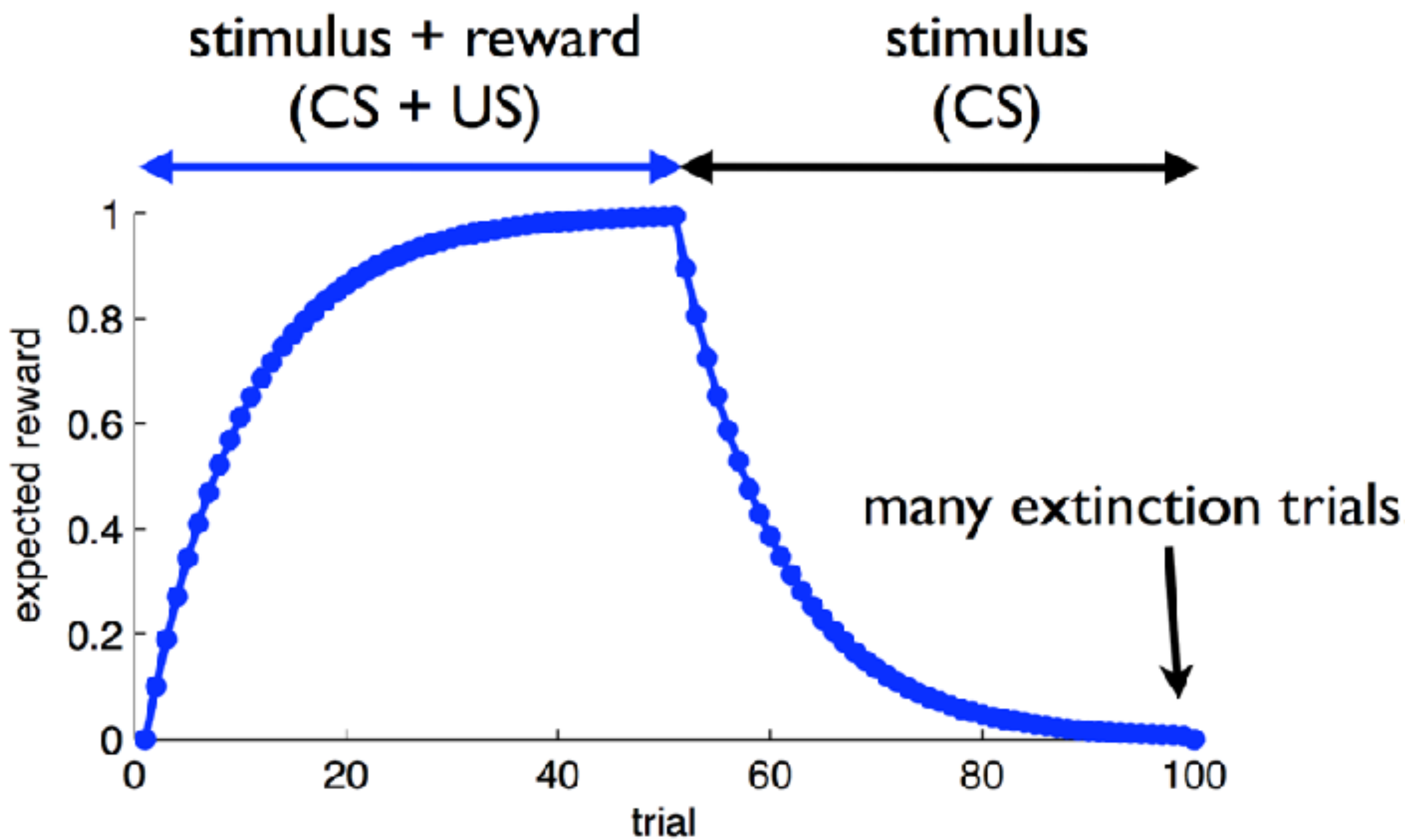
Rescorla-Wagner-rule

$$w \rightarrow w + \epsilon\delta_i u_i$$

learning rate

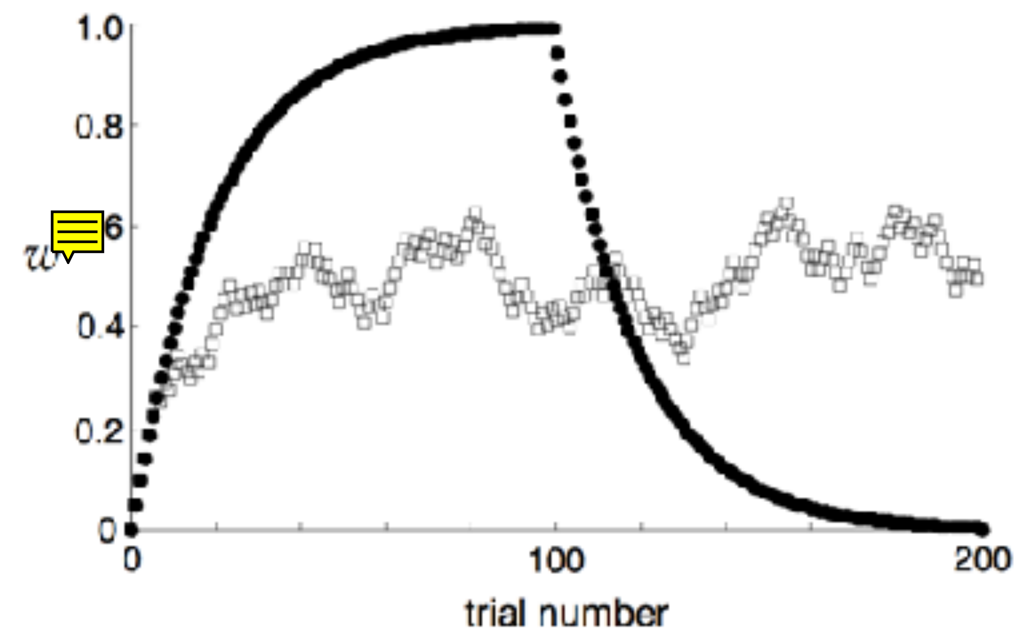
"delta-rule"

Learning **converges** to $w = 1$:



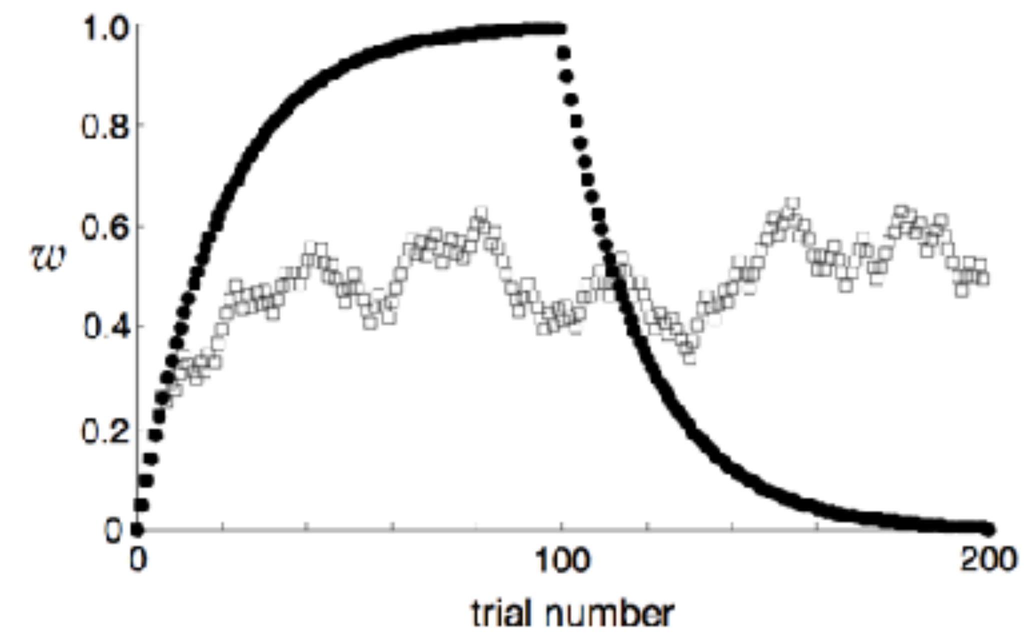
Rescorla-Wagner rule can explain other phenomena:

- **partial reinforcement**:
  the reward is delivered with a certain probability *p*



Rescorla-Wagner rule can explain other phenomena:

- **partial reinforcement**:
  the reward is delivered with a certain probability *p*



- With many stimuli and a single reward:

$$\vec{u} = (u_1, u_2)$$
$$\vec{w} = (w_1, w_2)$$

$$v = \vec{u}\vec{w} = u_1 w_1 + u_2 w_2$$

$$\vec{w} \rightarrow \vec{w} + \epsilon \delta \vec{u}$$
$$\delta = r - v$$

- **blocking**:
the animal cannot learn the association between the second stimulus and the reward if the reward is already predicted by the first one:

1. $u_1 \to r$
2. $u_1 + u_2 \to r$

$\Rightarrow$ $\cancel{u_2 \to r}$

- **blocking**:
the animal cannot learn the association between the second stimulus and the reward if the reward is already predicted by the first one:

1. $u_1 \to r$

2. $u_1 + u_2 \to r$

$\Rightarrow$ $\cancel{u_2 \to r}$

It cannot explain:

- **secondary conditioning**:
animals predict reward if the stimulus is associated to a second stimulus which already predict reward:

1. $u_1 \to r$

2. $u_1 + u_2 \to 0$

$\Rightarrow$ $\cancel{u_2 \to r}$

you need a prediction error to learn!

Programming: some more tricks

- Python built-in functions: enumerate
- Random numbers: numpy.random
- Plotting (scatter plot, remove box)

Tips for analysis and scientific writing

- Always send a PDF file
- Be careful with the units (and indicate them in your report)
- Instead of writing “I changed the initial conditions in Fig 1 …”,

  write: “Different initial conditions lead to … (see Fig 1)