TÉCNICO LISBOA

Planning, Learning and Decision Making

Lecture 18. Reinforcement learning: TD(λ)

Computing an average

- If we observe a new sample x_{N+1}

$$\bar{x}_{N+1} = \bar{x}_N + \frac{1}{N+1}\boxed{(x_{N+1} - \bar{x}_N)}$$

Error

Step

New estimate

Previous estimate

Computing a zero of a function

- Compute the sequence

$$x_{n+1} = x_n + \alpha_n \boxed{(H(x_n) - 0)}$$

New estimate

Previous estimate

Step

Error

Computing a FP of a function

- A fixed point of a function *F* is a point is the solution to

$$x = F(x)$$

or, equivalently,

$$H(x) = F(x) - x = 0$$

- We can use the approach for computing the zero of a function!

Computing a FP of a function

- Compute the sequence

$$x_{n+1} = x_n + \alpha_n \boxed{(F(x_n) - x_n)}$$

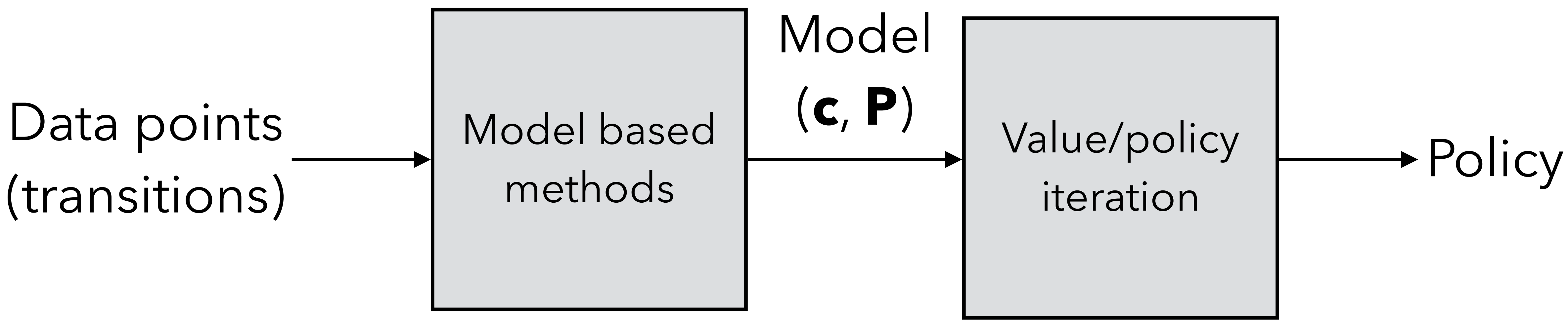
Error

Step

New estimate

Previous estimate

Model based RL



Computing J^π

- Given a sample (x_t, c_t, x_{t+1}), where the action was selected from π,

- Compute

$$\bar{\mathsf{P}}_{t+1}(y \mid x_t) = \bar{\mathsf{P}}_t(y \mid x_t) + \alpha(\mathbb{I}(\mathrm{x}_{t+1} = y) - \bar{\mathsf{P}}_t(y \mid x_t))$$

$$\bar{c}_{t+1}(x_t) = \bar{c}_t(x_t) + \alpha_t(c_t - \bar{c}_t(x_t))$$

- Compute

$$J_{t+1}(x_t) = \bar{c}_{t+1}(x_t) + \gamma \sum_{y \in \mathcal{X}} \bar{\mathbf{P}}_{t+1}(y \mid x_t) J_t(y)$$

Compute Q*

- Given a sample (x_t, a_t, c_t, x_{t+1})
- Compute

$$\bar{\mathsf{P}}_{t+1}(y \mid x_t, a_t) = \bar{\mathsf{P}}_t(y \mid x_t, a_t) + \alpha(\mathbb{I}(\mathrm{x}_{t+1} = y) - \bar{\mathsf{P}}_t(y \mid x_t, a_t))$$

$$\bar{c}_{t+1}(x_t, a_t) = \bar{c}_t(x_t, a_t) + \alpha_t(c_t - \bar{c}_t(x_t, a_t))$$

- Compute

$$Q_{t+1}(x_t, a_t) = \bar{c}_{t+1}(x_t, a_t) + \gamma \sum_{y \in \mathcal{X}} \bar{\mathbf{P}}_{t+1}(y \mid x_t, a_t) \min_{a' \in \mathcal{A}} Q_t(y, a')$$

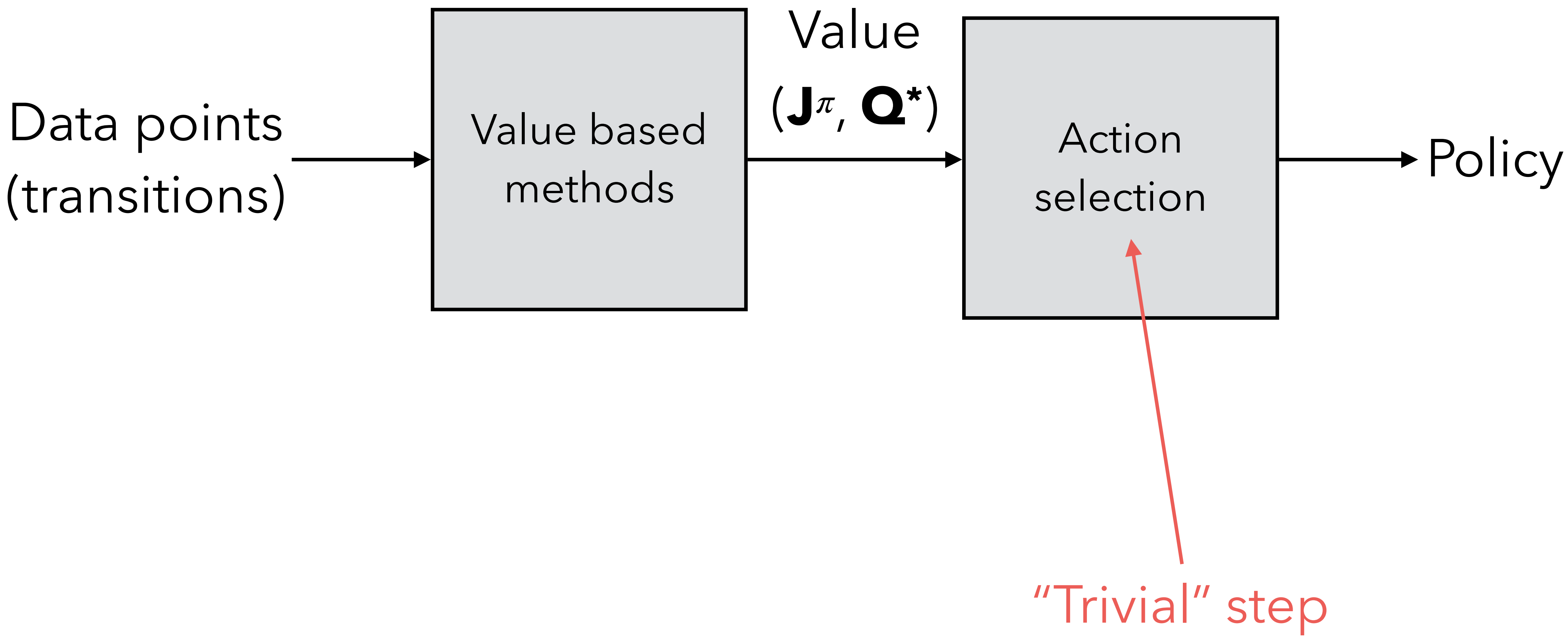
Does this work?

Theorem: Both approaches converge w.p.1 to J^π and Q^*, respectively, as long as every state (for J^π) or every state-action pair (for Q^*) is visited infinitely often.

Value based RL

Value based RL

- Value-based methods:



Computing J^π

- We have that

$$J^\pi(x) = c_\pi(x) + \gamma \sum_{y \in \mathcal{X}} \mathsf{P}_\pi(y \mid x) J^\pi(y)$$

which, back in lecture 7, we wrote as

$$\boldsymbol{J}^\pi = \mathbf{T}_\pi \boldsymbol{J}^\pi$$

$\boldsymbol{J}^\pi$ is a fixed point

Computing J^π

- Alternatively, for each state *x*, we can write

$$J^\pi(x) = \mathbb{E}_\pi\left[\mathrm{c} + \gamma J^\pi(\mathrm{y})\right]$$

Random variables

Computing J$^\pi$

- Alternatively, for each state *x*, we can write

$$\boxed{\mathbb{E}_\pi\left[\mathrm{c} + \gamma J^\pi(\mathrm{y}) - J^\pi(x)\right]} = 0$$

J$^\pi$ is the zero of
this function of *J*

Computing J^{π}

- Using the stochastic approximation/computation of the mean recipe

$$J_{t+1}(x_t) = J_t(x_t) + \alpha_t[c_t + \gamma J_t(x_{t+1}) - J_t(x_t)]$$

Can be seen as

$$J_{t+1}(x_t) = J_t(x_t) + \alpha_t[\mathbf{T}_{\pi} J_t(x_t) - J_t(x_t) + \varepsilon]$$

Temporal difference

- This algorithm is called TD-learning (temporal-difference learning) or TD(0)
- The quantity

$$c_t + \gamma J_t(x_{t+1}) - \boxed{J_t(x_t)}$$

Current estimate

Temporal difference

- This algorithm is called TD-learning (temporal-difference learning) or TD(0)
- The quantity

$$\boxed{c_t + \gamma J_t(x_{t+1})} - J_t(x_t)$$

Estimate with information from **next** time step

Temporal difference

- This algorithm is called TD-learning (temporal-difference learning) or TD(0)
- The quantity

$$\boxed{c_t + \gamma J_t(x_{t+1}) - J_t(x_t)}$$

Difference between current estimate and next time-step estimate

Temporal difference

- This algorithm is called TD-learning (temporal-difference learning) or TD(0)
- The quantity

$$c_t + \gamma J_t(x_{t+1}) - J_t(x_t)$$

  is known as **temporal difference**

- It corresponds to the current “estimation error”

Why TD(0)? Why the 0?

… let's play.

Temporal difference

- In vector form,

$$\boldsymbol{J}^{\pi} = \boldsymbol{c}_{\pi} + \gamma \mathsf{P}_{\pi} \boxed{\boldsymbol{J}^{\pi}}$$

We can replace this one

Temporal difference

- In vector form,

$$\boldsymbol{J}^{\pi} = \boldsymbol{c}_{\pi} + \gamma \mathsf{P}_{\pi}[\boldsymbol{c}_{\pi} + \gamma \mathsf{P}_{\pi} \boldsymbol{J}^{\pi}]$$

Temporal difference

- In vector form,

$$\boldsymbol{J}^\pi = \boldsymbol{c}_\pi + \gamma \mathsf{P}_\pi \boldsymbol{c}_\pi + \gamma^2 \mathsf{P}_\pi^2 \boxed{\boldsymbol{J}^\pi}$$

We can replace this one

Temporal difference

- In vector form,

$$\boldsymbol{J}^{\pi} = \boldsymbol{c}_{\pi} + \gamma \mathsf{P}_{\pi}\boldsymbol{c}_{\pi} + \gamma^2 \mathsf{P}_{\pi}^2[\boldsymbol{c}_{\pi} + \gamma \mathsf{P}_{\pi}\boldsymbol{J}^{\pi}]$$

Temporal difference

- In vector form,

$$\boldsymbol{J}^{\pi} = \boldsymbol{c}_{\pi} + \gamma \mathsf{P}_{\pi}\boldsymbol{c}_{\pi} + \gamma^2 \mathsf{P}_{\pi}^2 \boldsymbol{c}_{\pi} + \gamma^3 \mathsf{P}_{\pi}^3 \boldsymbol{J}^{\pi}$$

… many steps later…

Fixed points

- In vector form,

$$\boldsymbol{J}^\pi = \sum_{n=0}^{N} \gamma^n \mathsf{P}_\pi^n \boldsymbol{c}_\pi + \gamma^{N+1} \mathsf{P}_\pi^{N+1} \boldsymbol{J}^\pi$$

Fixed points

- So we have all these versions:

$$\boldsymbol{J}^\pi = \boldsymbol{c}_\pi + \gamma \mathsf{P}_\pi \boldsymbol{J}^\pi$$

$$\boldsymbol{J}^\pi = \boldsymbol{c}_\pi + \gamma \mathsf{P}_\pi \boldsymbol{c}_\pi + \gamma^2 \mathsf{P}_\pi^2 \boldsymbol{J}^\pi$$

$$\boldsymbol{J}^\pi = \boldsymbol{c}_\pi + \gamma \mathsf{P}_\pi \boldsymbol{c}_\pi + \gamma^2 \mathsf{P}_\pi^2 \boldsymbol{c}_\pi + \gamma^3 \mathsf{P}_\pi^3 \boldsymbol{J}^\pi$$

$$\vdots$$

$$\boldsymbol{J}^\pi = \sum_{n=0}^{N} \gamma^n \mathsf{P}_\pi^n \boldsymbol{c}_\pi + \gamma^{N+1} \mathsf{P}_\pi^{N+1} \boldsymbol{J}^\pi$$

$$\vdots$$

Variations…

- We can build an algorithm out of each…

$$J_{t+1}(x_t) = J_t(x_t) + \alpha_t[c_t + \gamma J_t(x_{t+1}) - J_t(x_t)]$$

$$J_{t+1}(x_t) = J_t(x_t) + \alpha_t[c_t + \gamma c_{t+1} + \gamma^2 J_t(x_{t+2}) - J_t(x_t)]$$

$$J_{t+1}(x_t) = J_t(x_t) + \alpha_t[c_t + \gamma c_{t+1} + \gamma^2 c_{t+2} + \gamma^3 J_t(x_{t+3}) - J_t(x_t)]$$

$$\vdots$$

$$J_{t+1}(x_t) = J_t(x_t) + \alpha_t \left[\sum_{n=0}^{N} \gamma^n c_{t+n} + \gamma^{N+1} J_t(x_{t+N+1}) - J_t(x_t)\right]$$

$$\vdots$$

Should we do this?

$$J_{t+1}(x_t) = J_t(x_t) + \alpha_t \left[\boxed{\sum_{n=0}^{N} \gamma^n c_{t+n}} + \gamma^{N+1} J_t(x_{t+N+1}) - J_t(x_t) \right]$$

- Good points:
  - Each update uses informations from multiple steps

Should we do this?

$$J_{t+1}(x_t) = J_t(x_t) + \alpha_t \left[\sum_{n=0}^{N} \gamma^n c_{t+n} + \gamma^{N+1} J_t(x_{t+N+1}) - J_t(x_t) \right]$$

- Good points:
  - Each update uses informations from multiple steps
  - Updates are more informative
  - Converges (potentially) faster

Should we do this?

$$J_{t+1}(x_t) = J_t(x_t) + \alpha_t \left[\sum_{n=0}^{N} \gamma^n c_{t+n} + \gamma^{N+1} J_t(\boxed{x_{t+N+1}}) - J_t(x_t) \right]$$

- Bad points:
  - Updates now require “looking into the distant future”

Should we do this?

$$J_{t+1}(x_t) = J_t(x_t) + \alpha_t \left[\sum_{n=0}^{N} \gamma^n c_{t+n} + \gamma^{N+1} J_t(x_{t+N+1}) - J_t(x_t) \right]$$

- Bad points:
  - Updates now require "looking into the distant future"
  - Updates requiring tracking "long transitions":

$$(x_t, c_t, \boxed{x_{t+1}}, c_{t+1}, \ldots, c_{t+N}, x_{t+N+1})$$

  - Updates discard information about intermediate states
  - Which *N* should we choose?

Revisited fixed point

- So we have all these versions:

$$(1-\lambda) \xrightarrow{\text{Multiply}} \boldsymbol{J}^\pi = \boldsymbol{c}_\pi + \gamma \mathsf{P}_\pi \boldsymbol{J}^\pi$$

$$(1-\lambda)\lambda \xrightarrow{\text{Multiply}} \boldsymbol{J}^\pi = \boldsymbol{c}_\pi + \gamma \mathsf{P}_\pi \boldsymbol{c}_\pi + \gamma^2 \mathsf{P}_\pi^2 \boldsymbol{J}^\pi$$

$$(1-\lambda)\lambda^2 \xrightarrow{\text{Multiply}} \boldsymbol{J}^\pi = \boldsymbol{c}_\pi + \gamma \mathsf{P}_\pi \boldsymbol{c}_\pi + \gamma^2 \mathsf{P}_\pi^2 \boldsymbol{c}_\pi + \gamma^3 \mathsf{P}_\pi^3 \boldsymbol{J}^\pi$$

$$\vdots \qquad\qquad \vdots$$

$$(1-\lambda)\lambda^N \xrightarrow{\text{Multiply}} \boldsymbol{J}^\pi = \sum_{n=0}^{N} \gamma^n \mathsf{P}_\pi^n \boldsymbol{c}_\pi + \gamma^{N+1} \mathsf{P}_\pi^{N+1} \boldsymbol{J}^\pi$$

$$\vdots$$

Revisited fixed point

- We get:

Add them all

$$(1-\lambda)\boldsymbol{J}^\pi = (1-\lambda)[\boldsymbol{c}_\pi + \gamma \mathsf{P}_\pi \boldsymbol{J}^\pi]$$

$$(1-\lambda)\lambda\boldsymbol{J}^\pi = (1-\lambda)\lambda[\boldsymbol{c}_\pi + \gamma \mathsf{P}_\pi \boldsymbol{c}_\pi + \gamma^2 \mathsf{P}_\pi^2 \boldsymbol{J}^\pi]$$

$$(1-\lambda)\lambda^2\boldsymbol{J}^\pi = (1-\lambda)\lambda^2[\boldsymbol{c}_\pi + \gamma \mathsf{P}_\pi \boldsymbol{c}_\pi + \gamma^2 \mathsf{P}_\pi^2 \boldsymbol{c}_\pi + \gamma^3 \mathsf{P}_\pi^3 \boldsymbol{J}^\pi]$$

$$\vdots$$

$$(1-\lambda)\lambda^N\boldsymbol{J}^\pi = (1-\lambda)\lambda^N \left[\sum_{n=0}^{N} \gamma^n \mathsf{P}_\pi^n \boldsymbol{c}_\pi + \gamma^{N+1} \mathsf{P}_\pi^{N+1} \boldsymbol{J}^\pi\right]$$

$$\vdots$$

Revisited fixed point

- We get:

$$\underbrace{\boxed{(1-\lambda)\sum_{N=1}^{\infty}\lambda^N}}_{=1}\boldsymbol{J}^\pi = (1-\lambda)\sum_{N=1}^{\infty}\lambda^N\left[\sum_{n=0}^{N}\gamma^n \mathsf{P}_\pi^n \boldsymbol{c}_\pi + \gamma^{N+1}\mathsf{P}_\pi^{N+1}\boldsymbol{J}^\pi\right]$$

Revisited fixed point

- We get:

$$\boldsymbol{J}^{\pi} = (1-\lambda)\sum_{N=1}^{\infty}\lambda^N\left[\sum_{n=0}^{N}\gamma^n\mathsf{P}_{\pi}^{n}\boldsymbol{c}_{\pi} + \gamma^{N+1}\mathsf{P}_{\pi}^{N+1}\boldsymbol{J}^{\pi}\right]$$

Chewing on this for a bit

$$\boldsymbol{J}^{\pi} = \sum_{n=0}^{\infty}\lambda^n\gamma^n\mathsf{P}_{\pi}^{n}\left[\boldsymbol{c}_{\pi} + \gamma\mathsf{P}_{\pi}\boldsymbol{J}_{\pi} - \boldsymbol{J}_{\pi}\right] + \boldsymbol{J}_{\pi}$$

Revisited fixed point

- We get:

$$\boldsymbol{J}^{\pi} = (1-\lambda)\sum_{N=1}^{\infty}\lambda^{N}\left[\sum_{n=0}^{N}\gamma^{n}\mathsf{P}_{\pi}^{n}\boldsymbol{c}_{\pi} + \gamma^{N+1}\mathsf{P}_{\pi}^{N+1}\boldsymbol{J}^{\pi}\right]$$

$$\Big\downarrow \text{ Chewing on this for a bit}$$

$$\sum_{n=0}^{\infty}\lambda^{n}\gamma^{n}\mathsf{P}_{\pi}^{n}\left[\boldsymbol{c}_{\pi} + \gamma\mathsf{P}_{\pi}\boldsymbol{J}_{\pi} - \boldsymbol{J}_{\pi}\right] = 0$$

Finally…

- We have a new algorithm:

$$J_{t+1}(x_t) = J_t(x_t) + \alpha_t \sum_{n=0}^{\infty} \lambda^n \gamma^n [c_{t+n} + \gamma J_t(\boxed{x_{t+n+1}}) - J_t(x_{t+n})]$$

- We no longer ignore intermediate states
- We no longer need to select an *N*

However…

- We now need an infinite trajectory!

DON'T
PANIC

What does this mean?

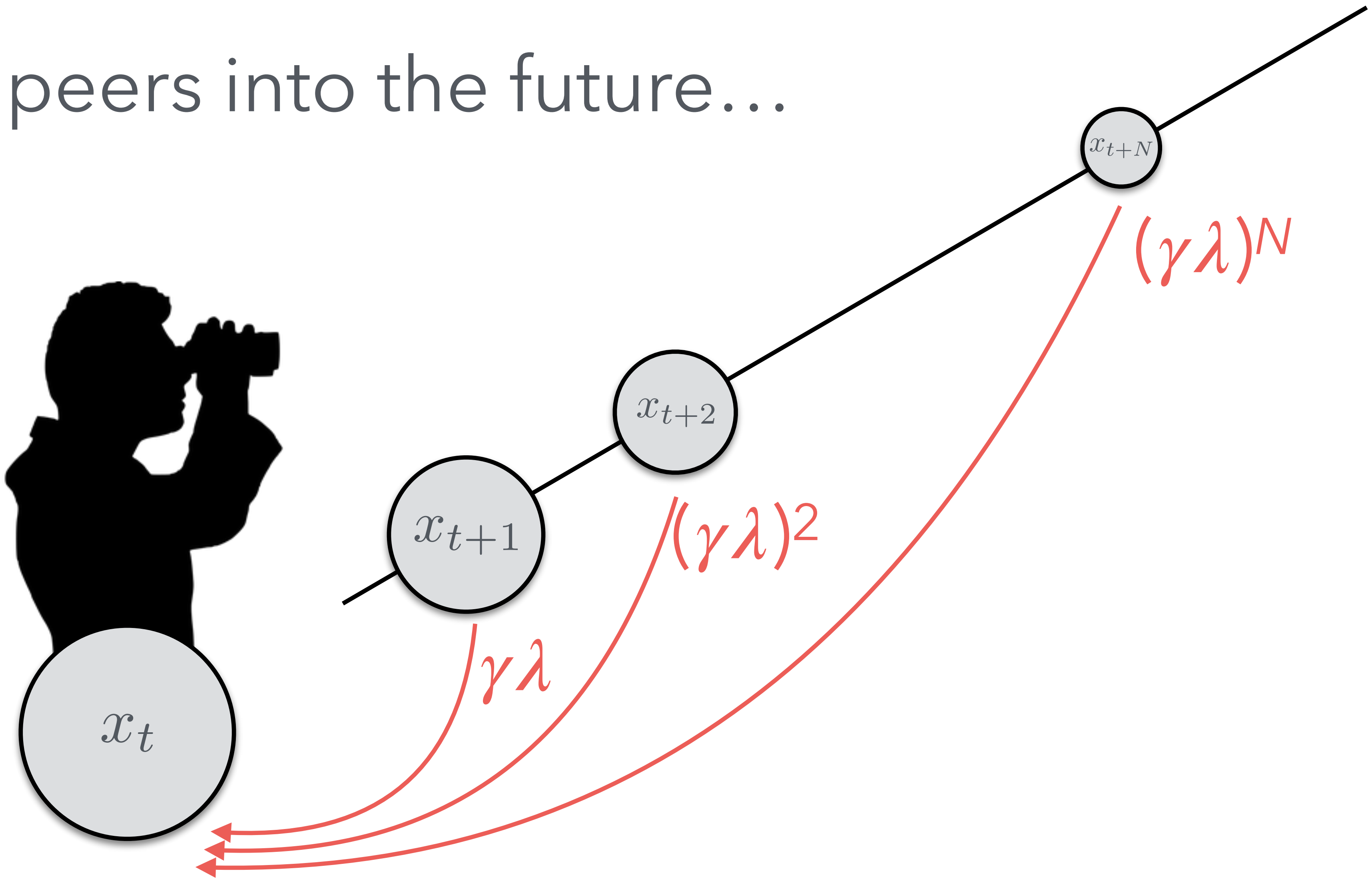
- Let's look at this carefully:

$$J_{t+1}(\boxed{x_t}) = J_t(x_t) + \alpha_t \sum_{n=0}^{\infty} \boxed{\lambda^n \gamma^n} [c_{t+n} + \gamma J_t(\boxed{x_{t+n+1}}) - J_t(x_{t+n})]$$

  - All states visited in the future contribute to current value
  - States further away contribute less (they are weighted down by $\gamma < 1$ and $\lambda \leq 1$)

Forward view

Agent peers into the future…

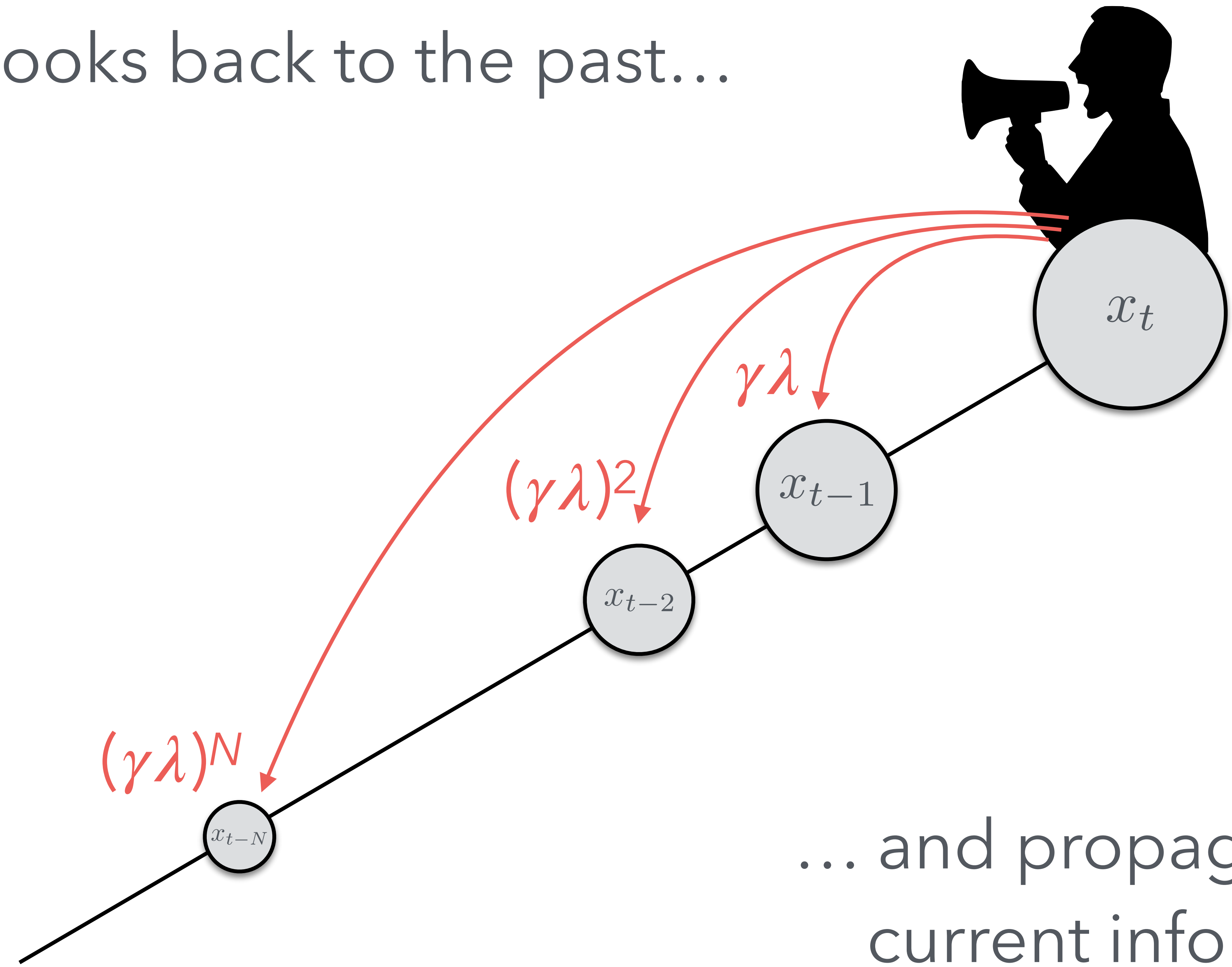


… and weights all future information

… but we can look at this the other way around…

Backward view

Agent looks back to the past…



… and propagates back current information

What does this mean?

- We track how much current state contributes to previous states:
  - We store how long ago previous states were visited
  - Weight current temporal difference accordingly

What does this mean?

- Algorithmically,

$$J_{t+1}(x) = J_t(x) + \alpha_t \boxed{z_{t+1}(x)} [c_t + \gamma J_t(x_{t+1}) - J_t(x_t)]$$

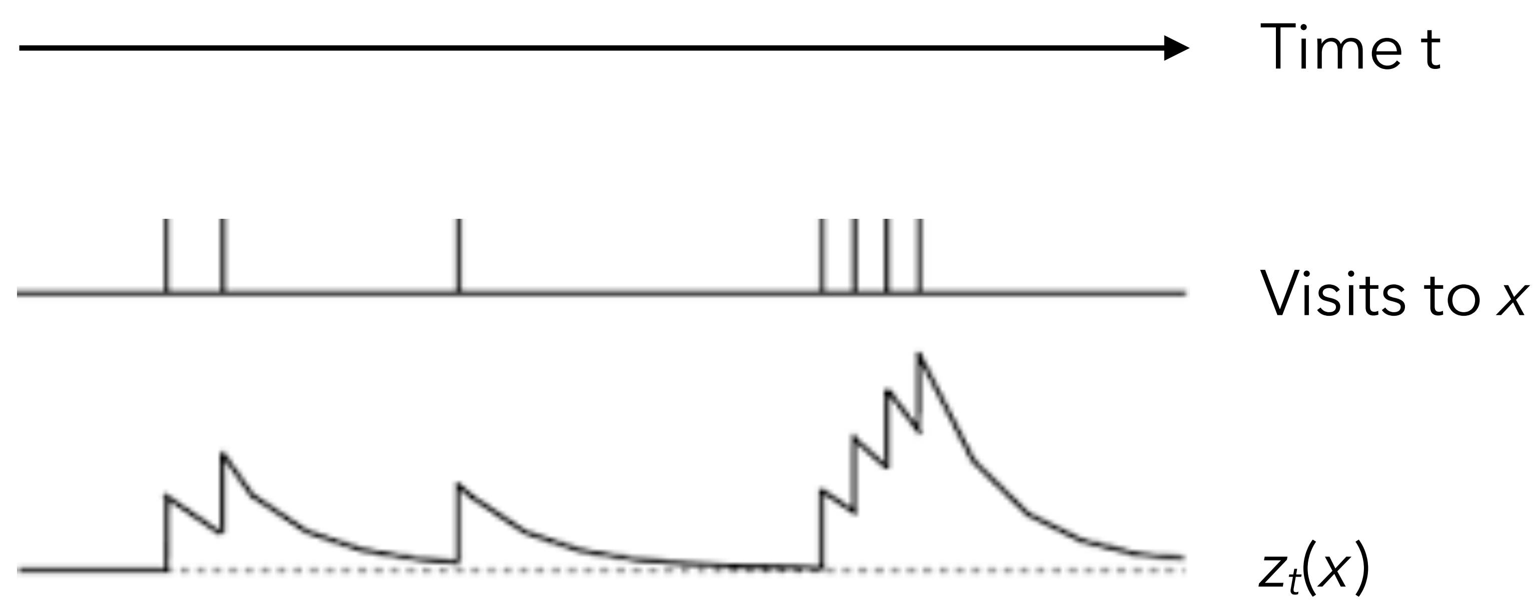
This factor traces
how much *x* should
"receive" from x_t

What does this mean?

- Algorithmically,

$$J_{t+1}(x) = J_t(x) + \alpha_t \boxed{z_{t+1}(x)} [c_t + \gamma J_t(x_{t+1}) - J_t(x_t)]$$

Eligibility trace



Temporal difference revisited

- Algorithmically,

$$J_{t+1}(x) = J_t(x) + \alpha_t z_{t+1}(x)[c_t + \gamma J_t(x_{t+1}) - J_t(x_t)]$$

$$z_{t+1}(x) = \lambda\gamma z_t(x) + \mathbb{I}(x = x_t)$$

- In this algorithm:
  - Each update uses informations from multiple steps
  - No looking in the future
  - No "long transitions" required

TD(λ)

- Given a sample (x_t, c_t, x_{t+1}), where the action was selected from π,
- Compute

$$z_{t+1}(x) = \lambda\gamma z_t(x) + \mathbb{I}(x = x_t)$$

$$J_{t+1}(x) = J_t(x) + \alpha_t z_{t+1}(x)[c_t + \gamma J_t(x_{t+1}) - J_t(x_t)]$$

- For λ = 0, we get TD(0) (the previous algorithm)

... hence the 0 in TD(0)

Does this work?

Theorem: For any $0 \leq \lambda \leq 1$, as long as every state is visited infinitely often, TD(λ) converges to J^π w.p.1.