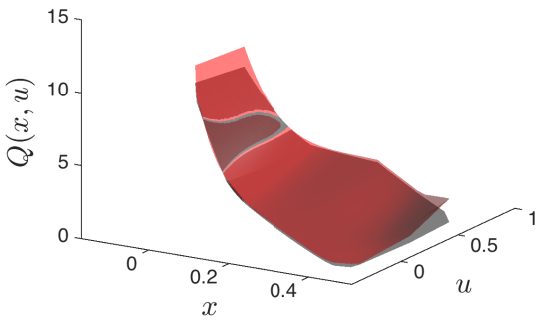
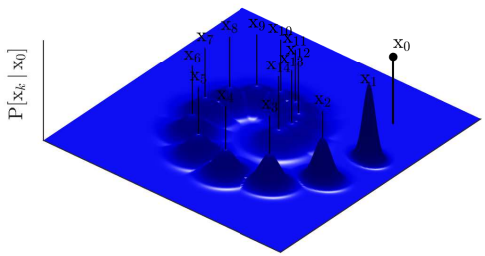
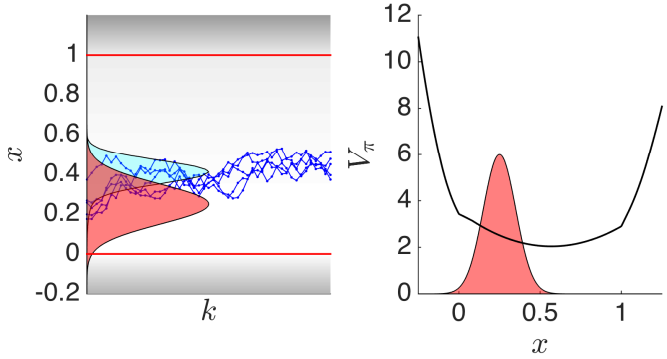
What are we going to discuss this week?

 # samples = 1000000

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$





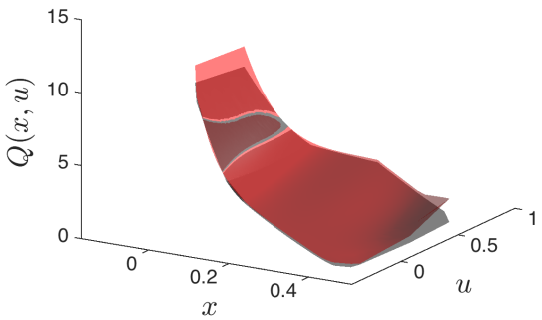


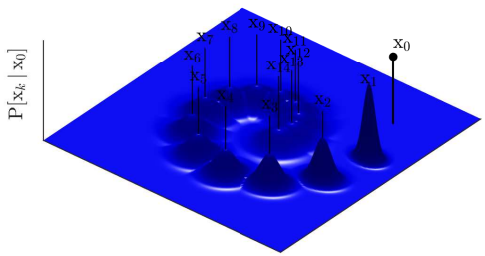
What are we going to discuss this week?

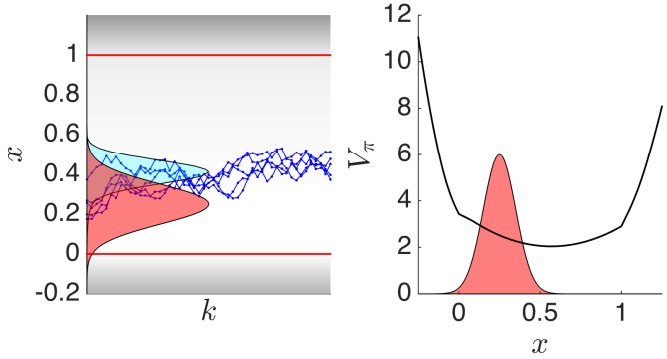
1. Optimal control on (almost) anything: Markov Decision Process & Bellman equations

 # samples = 1000000

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$





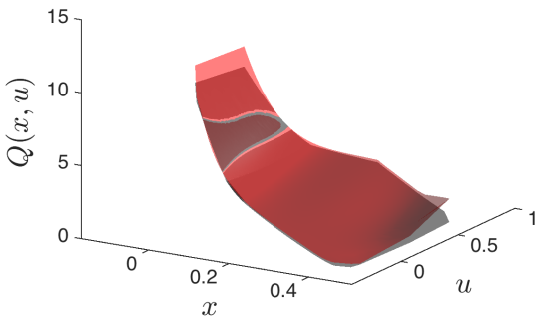


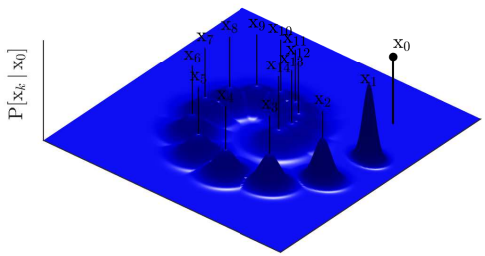
What are we going to discuss this week?

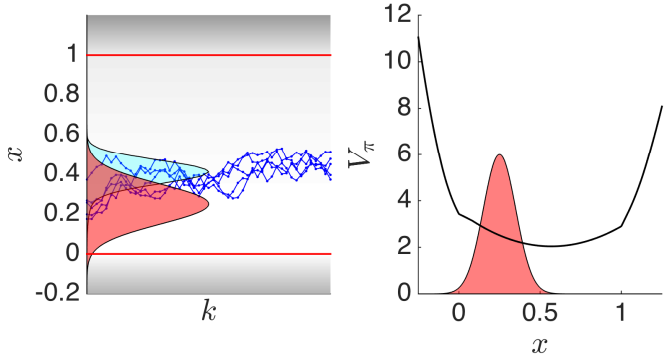
1. Optimal control on (almost) anything: Markov Decision Process & Bellman equations
2. Optimal control without models: Q-learning

 # samples = 1000000

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$





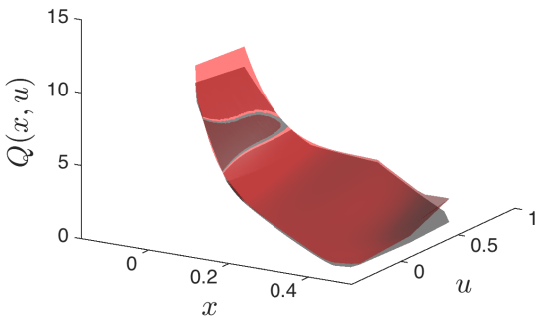


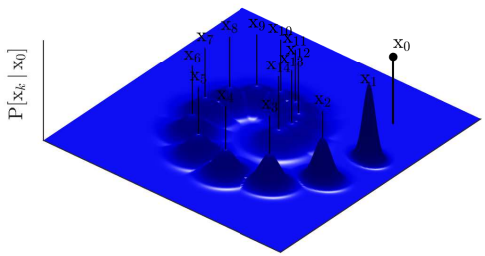
What are we going to discuss this week?

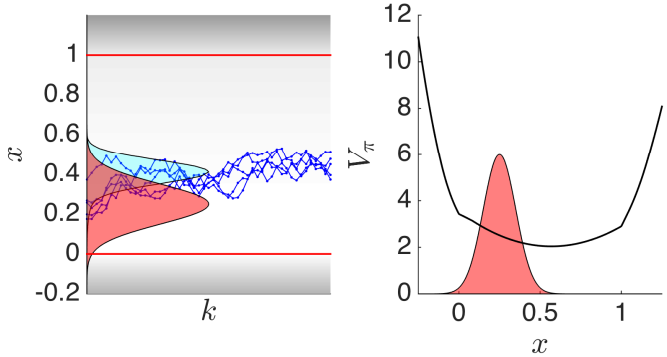
1. Optimal control on (almost) anything: Markov Decision Process & Bellman equations
2. Optimal control without models: Q-learning
3. Optimal control without models in practice: function approximation

 # samples = 1000000

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$





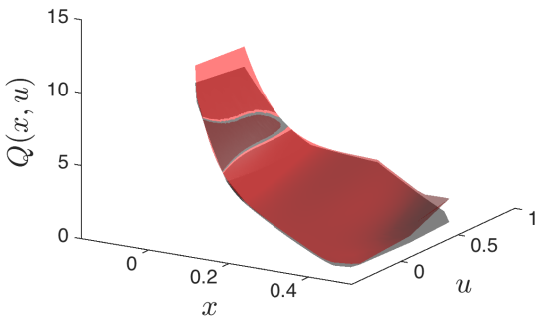


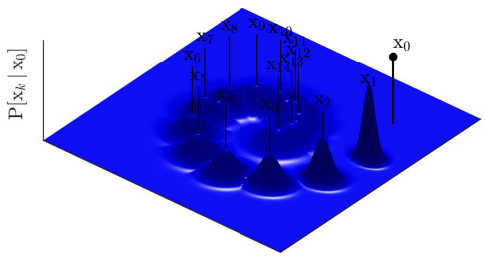
What are we going to discuss this week?

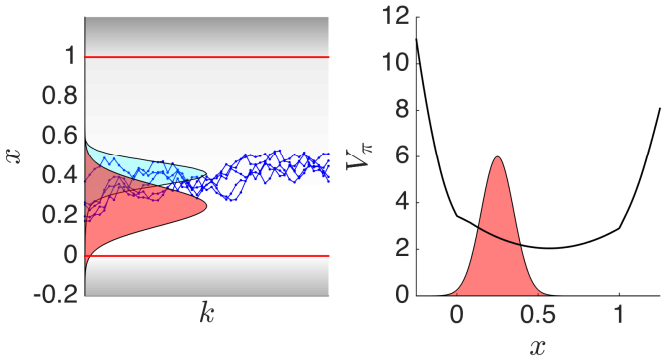
1. Optimal control on (almost) anything: Markov Decision Process & Bellman equations
2. Optimal control without models: Q-learning
3. Optimal control without models in practice: function approximation
4. Optimizing control policies: policy gradient methods

 # samples = 1000000

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$





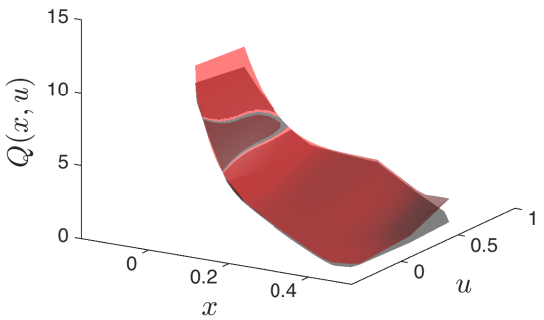


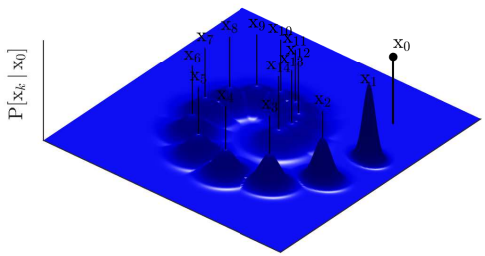
What are we going to discuss this week?

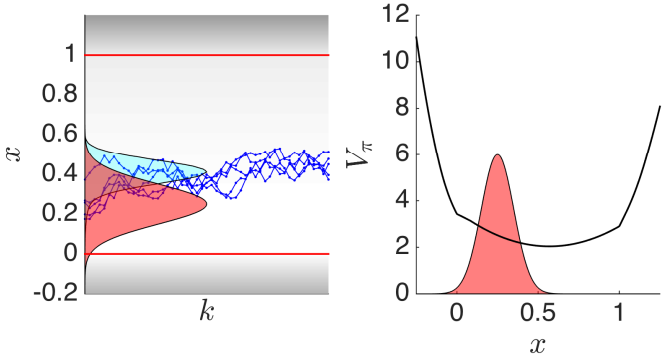
1. Optimal control on (almost) anything: Markov Decision Process & Bellman equations
2. Optimal control without models: Q-learning
3. Optimal control without models in practice: function approximation
4. Optimal control policies: policy gradient methods
5. Merging MPC and Learning: outlook on current research

 # samples = 1000000

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$







What are we going to discuss today?

"Classic" optimal control for given initial state $\mathbf{s}$:

$$\begin{aligned} \min_{\mathbf{x},\mathbf{u}} \quad & \int_0^T L(\mathbf{x},\mathbf{u})\,\mathrm{d}t \\ \text{s.t.} \quad & \mathbf{f}(\dot{\mathbf{x}},\mathbf{x},\mathbf{u}) = 0 \\ & \mathbf{h}(\mathbf{x},\mathbf{u}) \leq 0 \\ & \mathbf{x}(0) = \mathbf{s} \end{aligned}$$

Can we define & treat optimality for dynamic systems in a more abstract way?

What are we going to discuss today?

"Classic" optimal control for given initial state $\mathbf{s}$:

$$\begin{aligned} \min_{\mathbf{x},\mathbf{u}} \quad & \int_0^T L(\mathbf{x},\mathbf{u})\,\mathrm{d}t \\ \text{s.t.} \quad & \mathbf{f}(\dot{\mathbf{x}},\mathbf{x},\mathbf{u}) = 0 \\ & \mathbf{h}(\mathbf{x},\mathbf{u}) \leq 0 \\ & \mathbf{x}(0) = \mathbf{s} \end{aligned}$$

Can we define & treat optimality for dynamic systems in a more abstract way?

We want to address:

- Any set/space of state & input (functional, discrete, games, etc.)
- Any kind of dynamics (stochastic, mixed-integer, etc.)
- Any kind of stage cost (discontinuous, taking infinite values, stochastic, etc.)
- Infinite-horizon problems, state-dependent termination (game, target)
- Obtain input policies as opposed to input profiles

Markov Decision Processes: the Bellman equations and Dynamic Programming

Sébastien Gros

Cybernetic, NTNU
Elec. Eng., Chalmers

TUM lectures on RL

Outline

Notation & assumptions

- We will work in discrete time for simplicity

Notation & assumptions

- We will work in discrete time for simplicity
- State $\mathbf{x}$ represents the current "status" of a system

Notation & assumptions

- We will work in discrete time for simplicity
- State $\mathbf{x}$ represents the current "status" of a system
- Input $\mathbf{u}$ represents what we currently decide to do to the system (a.k.a. action)

Notation & assumptions

- We will work in discrete time for simplicity
- State $\mathbf{x}$ represents the current "status" of a system
- Input $\mathbf{u}$ represents what we currently decide to do to the system (a.k.a. action)
- Classic understanding applies but no restriction, e.g.:
  - $\mathbf{x}, \mathbf{u}$ can be discrete (e.g. position and movement in a game)
  - $\mathbf{x}, \mathbf{u}$ can belong to abstract vector spaces (e.g. functions)

Notation & assumptions

- We will work in discrete time for simplicity
- State $\mathbf{x}$ represents the current "status" of a system
- Input $\mathbf{u}$ represents what we currently decide to do to the system (a.k.a. action)
- Classic understanding applies but no restriction, e.g.:
  - $\mathbf{x}, \mathbf{u}$ can be discrete (e.g. position and movement in a game)
  - $\mathbf{x}, \mathbf{u}$ can belong to abstract vector spaces (e.g. functions)
- We will often use the objects $\mathbf{x}, \mathbf{u}, \mathbf{x}_+$ together, meaning

$$\underbrace{\mathbf{x}, \mathbf{u}}_{\text{current state \& input}} \longrightarrow \underbrace{\mathbf{x}_+}_{\text{next state}}$$

Notation & assumptions

- We will work in discrete time for simplicity
- State $\mathbf{x}$ represents the current "status" of a system
- Input $\mathbf{u}$ represents what we currently decide to do to the system (a.k.a. action)
- Classic understanding applies but no restriction, e.g.:
  - $\mathbf{x}, \mathbf{u}$ can be discrete (e.g. position and movement in a game)
  - $\mathbf{x}, \mathbf{u}$ can belong to abstract vector spaces (e.g. functions)
- We will often use the objects $\mathbf{x}, \mathbf{u}, \mathbf{x}_+$ together, meaning

$$\underbrace{\mathbf{x}, \mathbf{u}}_{\text{current state \& input}} \longrightarrow \underbrace{\mathbf{x}_+}_{\text{next state}}$$

- Stochastic: at current time, $\mathbf{x}_+$ is typically unknown, even if $\mathbf{x}, \mathbf{u}$ are known

Notation & assumptions

- We will work in discrete time for simplicity
- State $\mathbf{x}$ represents the current "status" of a system
- Input $\mathbf{u}$ represents what we currently decide to do to the system (a.k.a. action)
- Classic understanding applies but no restriction, e.g.:
  - $\mathbf{x}, \mathbf{u}$ can be discrete (e.g. position and movement in a game)
  - $\mathbf{x}, \mathbf{u}$ can belong to abstract vector spaces (e.g. functions)
- We will often use the objects $\mathbf{x}, \mathbf{u}, \mathbf{x}_+$ together, meaning

$$\underbrace{\mathbf{x}, \mathbf{u}}_{\text{current state \& input}} \longrightarrow \underbrace{\mathbf{x}_+}_{\text{next state}}$$

- Stochastic: at current time, $\mathbf{x}_+$ is typically unknown, even if $\mathbf{x}, \mathbf{u}$ are known
- We can consider ∞ and finite processes, termination time may be unknown. E.g.
  - Chess game: finite process terminating at an unknown time
  - Landing a plane: finite process terminating at a given time/state
  - Managing a smart building: infinite process

Notation & assumptions

- We will work in discrete time for simplicity
- State $\mathbf{x}$ represents the current "status" of a system
- Input $\mathbf{u}$ represents what we currently decide to do to the system (a.k.a. action)
- Classic understanding applies but no restriction, e.g.:
  - $\mathbf{x}, \mathbf{u}$ can be discrete (e.g. position and movement in a game)
  - $\mathbf{x}, \mathbf{u}$ can belong to abstract vector spaces (e.g. functions)
- We will often use the objects $\mathbf{x}, \mathbf{u}, \mathbf{x}_+$ together, meaning

$$\underbrace{\mathbf{x}, \mathbf{u}}_{\text{current state \& input}} \longrightarrow \underbrace{\mathbf{x}_+}_{\text{next state}}$$

- Stochastic: at current time, $\mathbf{x}_+$ is typically unknown, even if $\mathbf{x}, \mathbf{u}$ are known
- We can consider ∞ and finite processes, termination time may be unknown. E.g.
  - Chess game: finite process terminating at an unknown time
  - Landing a plane: finite process terminating at a given time/state
  - Managing a smart building: infinite process
- What do we keep from "classic" optimal control?
  - Causality
  - Performance is minimizing the sum of L

Notation

Conditional probability $\mathbb{P}[\mathbf{y} \,|\, \mathbf{x}]$ is a function of $\mathbf{x}$ and $\mathbf{y}$

Probability distribution (discrete or density) of random variable $\mathbf{y}$ for $\mathbf{x}$ known, e.g.:

$$\mathbb{P}[\mathbf{y} \,|\, \mathbf{x}] = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2\sigma}\|\mathbf{y}-\boldsymbol{\nu}(\mathbf{x})\|^2}$$

Notation

Conditional probability $\mathbb{P}[\mathbf{y} \mid \mathbf{x}]$ is a function of $\mathbf{x}$ and $\mathbf{y}$

Probability distribution (discrete or density) of random variable $\mathbf{y}$ for $\mathbf{x}$ known, e.g.:

$$\mathbb{P}[\mathbf{y} \mid \mathbf{x}] = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1}{2\sigma}\|\mathbf{y}-\boldsymbol{\nu}(\mathbf{x})\|^2}$$

Expected value $\mathbb{E}[\mathbf{v}(\mathbf{y})]$

For discrete distributions: $\mathbb{E}[\mathbf{v}(\mathbf{y})] = \sum_{\mathbf{y}\in D} \mathbf{v}(\mathbf{y})\mathbb{P}[\mathbf{y}]$

For densities: $\mathbb{E}[\mathbf{v}(\mathbf{y})] = \int_{\mathbf{y}\in D} \mathbf{v}(\mathbf{y}) \underbrace{\mathbb{P}[\mathbf{y}]\,\mathrm{d}\mathbf{y}}_{=\mu_{\mathbb{P}[\mathbf{y}]}(\mathbf{y})}$ (may need Lebesgue int.)

Notation

Conditional probability $\mathbb{P}[\mathbf{y} \,|\, \mathbf{x}]$ is a function of $\mathbf{x}$ and $\mathbf{y}$

Probability distribution (discrete or density) of random variable $\mathbf{y}$ for $\mathbf{x}$ known, e.g.:

$$\mathbb{P}[\mathbf{y} \,|\, \mathbf{x}] = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1}{2\sigma}\|\mathbf{y}-\boldsymbol{\nu}(\mathbf{x})\|^2}$$

Expected value $\mathbb{E}[\mathbf{v}(\mathbf{y})]$

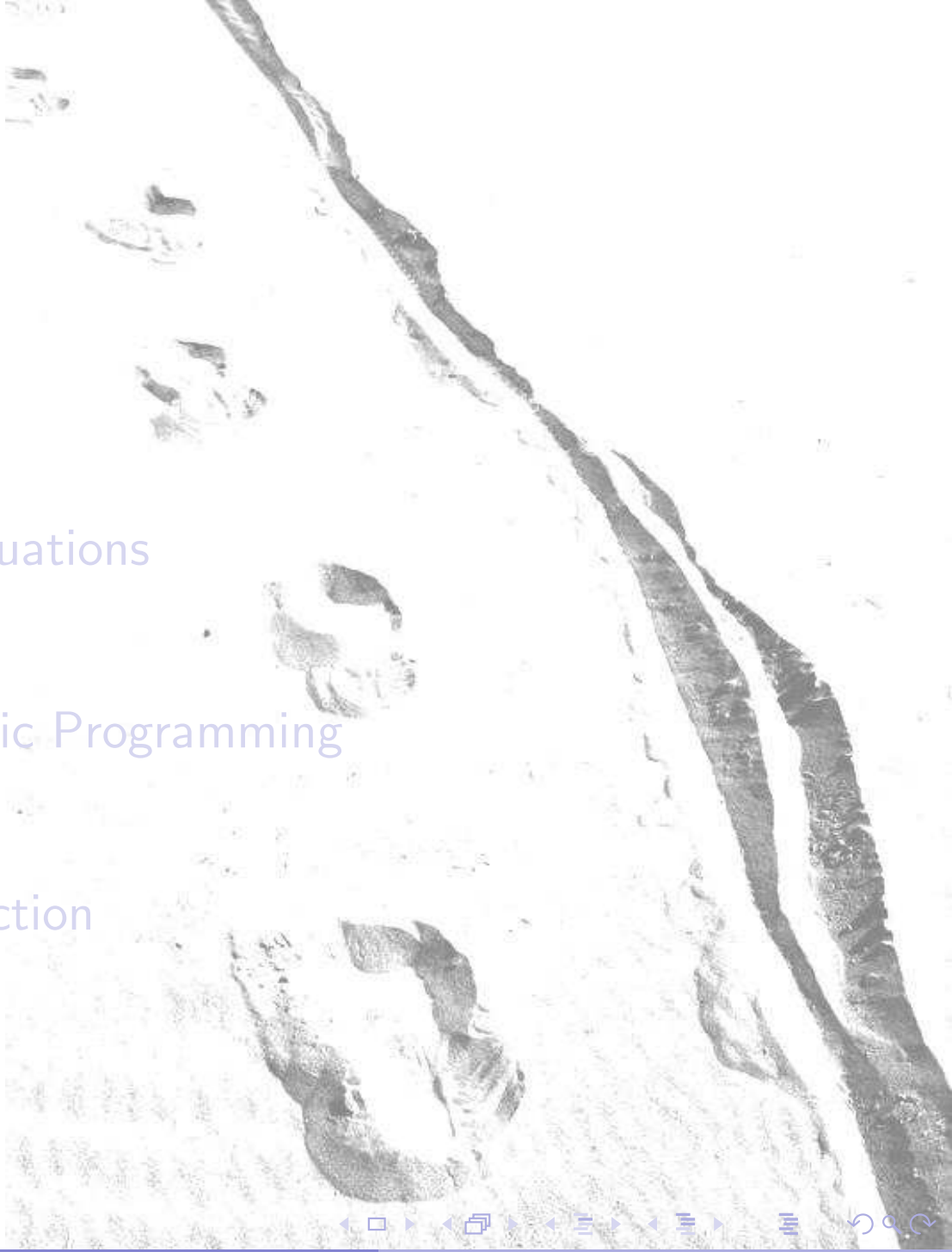
For discrete distributions: $\mathbb{E}[\mathbf{v}(\mathbf{y})] = \sum_{\mathbf{y}\in D} \mathbf{v}(\mathbf{y})\mathbb{P}[\mathbf{y}]$

For densities: $\mathbb{E}[\mathbf{v}(\mathbf{y})] = \int_{\mathbf{y}\in D} \mathbf{v}(\mathbf{y}) \underbrace{\mathbb{P}[\mathbf{y}]\,\mathrm{d}\mathbf{y}}_{=\mu_{\mathbb{P}[\mathbf{y}]}(\mathbf{y})}$ (may need Lebesgue int.)

Conditional expected value $\mathbb{E}[\mathbf{v}(\mathbf{y}) \,|\, \mathbf{x}]$ is a function of $\mathbf{x}$

For densities: $\mathbb{E}[\mathbf{v}(\mathbf{y}) \,|\, \mathbf{x}] = \int_{\mathbf{y}\in D} \mathbf{v}(\mathbf{y})\mathbb{P}[\mathbf{y} \,|\, \mathbf{x}]\,\mathrm{d}\mathbf{y}$

Outline



State transitions - how to describe dynamics?

State transition

$$\underbrace{\mathbf{x}, \mathbf{u}}_{\text{current state \& input}} \longrightarrow \underbrace{\mathbf{x}_+}_{\text{next state}}$$

State transitions - how to describe dynamics?

State transition

$$\underbrace{\mathbf{x}, \mathbf{u}}_{\text{current state \& input}} \longrightarrow \underbrace{\mathbf{x}_+}_{\text{next state}}$$

- In classic control, models are $\mathbf{x}_+ = \mathbf{f}(\mathbf{x}, \mathbf{u})$ for some function $\mathbf{f}$

State transitions - how to describe dynamics?

State transition

$$\underbrace{\mathbf{x}, \mathbf{u}}_{\text{current state \& input}} \longrightarrow \underbrace{\mathbf{x}_+}_{\text{next state}}$$

- In classic control, models are $\mathbf{x}_+ = \mathbf{f}(\mathbf{x}, \mathbf{u})$ for some function $\mathbf{f}$
- More generally state transitions are stochastic, given by conditional distributions

$$\mathbb{P}[\mathbf{x}_+ \,|\, \mathbf{x}, \mathbf{u}]$$

Meaning: probability of transiting to state $\mathbf{x}_+$ provided that the system is in state $\mathbf{x}$ and subject to input $\mathbf{u}$

State transitions - how to describe dynamics?

State transition

$$\underbrace{\mathbf{x}, \mathbf{u}}_{\text{current state \& input}} \longrightarrow \underbrace{\mathbf{x}_+}_{\text{next state}}$$

- In classic control, models are $\mathbf{x}_+ = \mathbf{f}(\mathbf{x}, \mathbf{u})$ for some function $\mathbf{f}$
- More generally state transitions are stochastic, given by conditional distributions

$$\mathbb{P}[\mathbf{x}_+ \,|\, \mathbf{x}, \mathbf{u}]$$

  Meaning: probability of transiting to state $\mathbf{x}_+$ provided that the system is in state $\mathbf{x}$ and subject to input $\mathbf{u}$
- Classic model $\mathbf{x}_+ = \mathbf{f}(\mathbf{x}, \mathbf{u})$ is a special case of stochastic transition, by defining:

$$\mathbb{P}[\mathbf{x}_+ \,|\, \mathbf{x}, \mathbf{u}] = \delta(\mathbf{x}_+ - \mathbf{f}(\mathbf{x}, \mathbf{u}))$$

where δ is a Dirac function

State transitions - how to describe dynamics?

State transition

$$\underbrace{\mathbf{x}, \mathbf{u}}_{\text{current state \& input}} \longrightarrow \underbrace{\mathbf{x}_+}_{\text{next state}}$$

- In classic control, models are $\mathbf{x}_+ = \mathbf{f}(\mathbf{x}, \mathbf{u})$ for some function $\mathbf{f}$
- More generally state transitions are stochastic, given by conditional distributions

$$\mathbb{P}[\mathbf{x}_+ \,|\, \mathbf{x}, \mathbf{u}]$$

  Meaning: probability of transiting to state $\mathbf{x}_+$ provided that the system is in state $\mathbf{x}$ and subject to input $\mathbf{u}$
- Classic model $\mathbf{x}_+ = \mathbf{f}(\mathbf{x}, \mathbf{u})$ is a special case of stochastic transition, by defining:

$$\mathbb{P}[\mathbf{x}_+ \,|\, \mathbf{x}, \mathbf{u}] = \delta(\mathbf{x}_+ - \mathbf{f}(\mathbf{x}, \mathbf{u}))$$

  where δ is a Dirac function
- Process noise $\mathbf{x}_+ = \mathbf{f}(\mathbf{x}, \mathbf{u}) + \mathbf{e}$ where e.g. $\mathbf{e} \sim \mathcal{N}(0, \Sigma)$, then

$$\mathbb{P}[\mathbf{x}_+ \,|\, \mathbf{x}, \mathbf{u}] \propto e^{-\frac{1}{2}(\mathbf{x}_+ - \mathbf{f}(\mathbf{x},\mathbf{u}))^\top \Sigma (\mathbf{x}_+ - \mathbf{f}(\mathbf{x},\mathbf{u}))}$$

Stage cost - Assessing Performance

Stage cost function: $L(\mathbf{x}, \mathbf{u}) \in \mathbb{R}$

- $L(\mathbf{x}, \mathbf{u}) = +\infty$ for infeasible states and inputs (constraints in the cost)
- $L(\mathbf{x}, \mathbf{u})$ *may not* be lower-bounded

Stage cost - Assessing Performance

Stage cost function: $L(\mathbf{x}, \mathbf{u}) \in \mathbb{R}$

- $L(\mathbf{x}, \mathbf{u}) = +\infty$ for infeasible states and inputs (constraints in the cost)
- $L(\mathbf{x}, \mathbf{u})$ *may not* be lower-bounded

Examples

LQR

$$L(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\mathbf{x}^\top Q\mathbf{x} + \frac{1}{2}\mathbf{u}^\top R\mathbf{u}$$

Stage cost - Assessing Performance

Stage cost function: $L(\mathbf{x}, \mathbf{u}) \in \mathbb{R}$

- $L(\mathbf{x}, \mathbf{u}) = +\infty$ for infeasible states and inputs (constraints in the cost)
- $L(\mathbf{x}, \mathbf{u})$ *may not* be lower-bounded

Examples

LQR

$$L(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\mathbf{x}^\top Q\mathbf{x} + \frac{1}{2}\mathbf{u}^\top R\mathbf{u}$$

MPC

$$\frac{1}{2}\mathbf{x}^\top Q\mathbf{x} + \frac{1}{2}\mathbf{u}^\top R\mathbf{u}$$
$$\text{s.t.} \quad \mathbf{x} \in \mathcal{X}, \quad \mathbf{u} \in \mathcal{U}$$

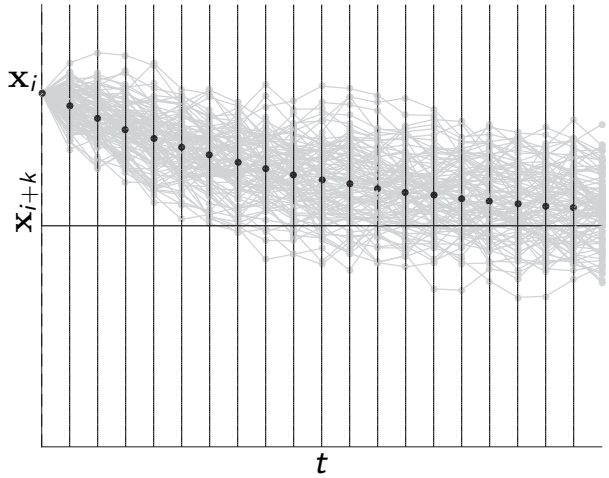
yields:

$$L(\mathbf{x}, \mathbf{u}) = \begin{cases} \frac{1}{2}\mathbf{x}^\top Q\mathbf{x} + \frac{1}{2}\mathbf{u}^\top R\mathbf{u} & \text{if } \mathbf{x} \in \mathcal{X},\ \mathbf{u} \in \mathcal{U} \\ \infty & \text{otherwise} \end{cases}$$

MDP, return and discount - How to assess long-term performance?

The triplet $(\mathbb{P}[\mathbf{x}_+ \,|\, \mathbf{x}, \mathbf{u}], \gamma, L)$ define a Markov Decision Process

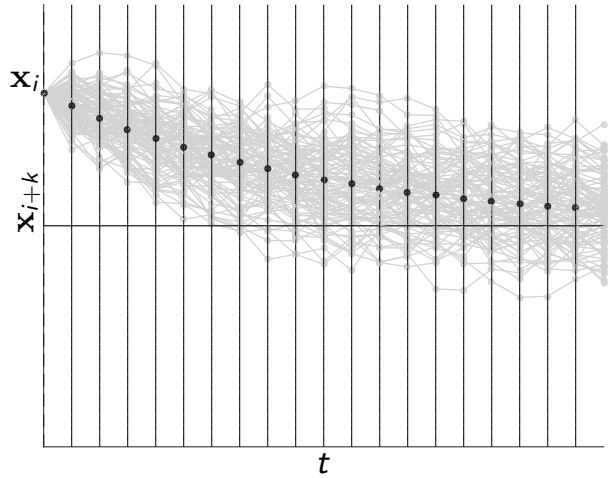
- State and input $\mathbf{x}, \mathbf{u}$
- State transition (model) $\mathbb{P}[\mathbf{x}_+ \,|\, \mathbf{x}, \mathbf{u}]$
- Stage cost L
- Discount γ



MDP, return and discount - How to assess long-term performance?

The triplet $(\mathbb{P}[\mathbf{x}_+ \,|\, \mathbf{x}, \mathbf{u}], \gamma, L)$ define a Markov Decision Process

- State and input $\mathbf{x}, \mathbf{u}$
- State transition (model) $\mathbb{P}[\mathbf{x}_+ \,|\, \mathbf{x}, \mathbf{u}]$
- Stage cost L
- Discount γ



Cost-to-go (a.k.a. return)

$$J = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_{i+k}, \mathbf{u}_{i+k})\right]$$

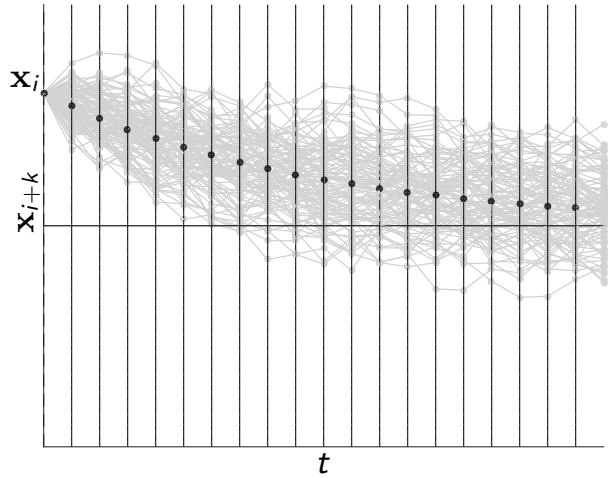
where

- $\mathbf{x}_{i+k}, \mathbf{u}_{i+k}$ is the state trajectory and input sequence
- $\gamma \leq 1$ is the discount factor

MDP, return and discount - How to assess long-term performance?

The triplet $(\mathbb{P}[\mathbf{x}_+ \,|\, \mathbf{x}, \mathbf{u}], \gamma, L)$ define a Markov Decision Process

- State and input $\mathbf{x}, \mathbf{u}$
- State transition (model) $\mathbb{P}[\mathbf{x}_+ \,|\, \mathbf{x}, \mathbf{u}]$
- Stage cost L
- Discount γ



Cost-to-go (a.k.a. return)

$$J = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_{i+k}, \mathbf{u}_{i+k})\right]$$

Why a discount factor?

where

- $\mathbf{x}_{i+k}, \mathbf{u}_{i+k}$ is the state trajectory and input sequence
- $\gamma \leq 1$ is the discount factor

Policy (decisions) taken such that J is minimized

What is $\mathbb{E}$ expecting over (distribution)?

Policy - How to define decisions?

Deterministic

$$\mathbf{u} = \boldsymbol{\pi}(\mathbf{x})$$

Meaning: what input $\mathbf{u}$ do we select *provided* that we are in state $\mathbf{x}$

Policy - How to define decisions?

Deterministic

$$\mathbf{u} = \boldsymbol{\pi}(\mathbf{x})$$

Meaning: what input $\mathbf{u}$ do we select *provided* that we are in state $\mathbf{x}$

Example LQR:

$$L(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\mathbf{x}^\top Q\mathbf{x} + \frac{1}{2}\mathbf{u}^\top R\mathbf{u}$$

$$\mathbf{x}_+ = A\mathbf{x} + B\mathbf{u}$$

yields: $\mathbf{u} = -K\mathbf{x}$
hence: $\boldsymbol{\pi}(\mathbf{x}) = -K\mathbf{x}$

Policy - How to define decisions?

Deterministic

$$\mathbf{u} = \boldsymbol{\pi}(\mathbf{x})$$

Meaning: what input $\mathbf{u}$ do we select *provided* that we are in state $\mathbf{x}$

Stochastic

$$\boldsymbol{\pi}[\mathbf{u} \,|\, \mathbf{x}]$$

Example LQR:

$$L(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\mathbf{x}^\top Q\mathbf{x} + \frac{1}{2}\mathbf{u}^\top R\mathbf{u}$$

$$\mathbf{x}_+ = A\mathbf{x} + B\mathbf{u}$$

yields: $\mathbf{u} = -K\mathbf{x}$

hence: $\boldsymbol{\pi}(\mathbf{x}) = -K\mathbf{x}$

Policy - How to define decisions?

Deterministic

$$\mathbf{u} = \boldsymbol{\pi}(\mathbf{x})$$

Meaning: what input $\mathbf{u}$ do we select *provided* that we are in state $\mathbf{x}$

Stochastic

$$\boldsymbol{\pi}[\mathbf{u} \,|\, \mathbf{x}]$$

Meaning: probability of selecting input $\mathbf{u}$ *provided* that we are in state $\mathbf{x}$

Example LQR:

$$L(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\mathbf{x}^\top Q\mathbf{x} + \frac{1}{2}\mathbf{u}^\top R\mathbf{u}$$

$$\mathbf{x}_+ = A\mathbf{x} + B\mathbf{u}$$

yields: $\mathbf{u} = -K\mathbf{x}$

hence: $\boldsymbol{\pi}(\mathbf{x}) = -K\mathbf{x}$

Policy - How to define decisions?

Deterministic

$$\mathbf{u} = \boldsymbol{\pi}(\mathbf{x})$$

Meaning: what input $\mathbf{u}$ do we select *provided* that we are in state $\mathbf{x}$

Example LQR:

$$L(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\mathbf{x}^\top Q\mathbf{x} + \frac{1}{2}\mathbf{u}^\top R\mathbf{u}$$

$$\mathbf{x}_+ = A\mathbf{x} + B\mathbf{u}$$

yields: $\mathbf{u} = -K\mathbf{x}$

hence: $\boldsymbol{\pi}(\mathbf{x}) = -K\mathbf{x}$

Stochastic

$$\boldsymbol{\pi}[\mathbf{u} \,|\, \mathbf{x}]$$

Meaning: probability of selecting input $\mathbf{u}$ *provided* that we are in state $\mathbf{x}$

Why?

Policy - How to define decisions?

Deterministic

$$\mathbf{u} = \boldsymbol{\pi}(\mathbf{x})$$

Meaning: what input $\mathbf{u}$ do we select *provided* that we are in state $\mathbf{x}$

Example LQR:

$$L(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\mathbf{x}^\top Q\mathbf{x} + \frac{1}{2}\mathbf{u}^\top R\mathbf{u}$$
$$\mathbf{x}_+ = A\mathbf{x} + B\mathbf{u}$$

yields: $\mathbf{u} = -K\mathbf{x}$
hence: $\boldsymbol{\pi}(\mathbf{x}) = -K\mathbf{x}$

Stochastic

$$\boldsymbol{\pi}[\mathbf{u} \,|\, \mathbf{x}]$$

Meaning: probability of selecting input $\mathbf{u}$ *provided* that we are in state $\mathbf{x}$

Why?

- unsure of the policy, "try" different decisions

Policy - How to define decisions?

Deterministic

$$\mathbf{u} = \boldsymbol{\pi}(\mathbf{x})$$

Meaning: what input $\mathbf{u}$ do we select *provided* that we are in state $\mathbf{x}$

Example LQR:

$$L(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\mathbf{x}^\top Q\mathbf{x} + \frac{1}{2}\mathbf{u}^\top R\mathbf{u}$$

$$\mathbf{x}_+ = A\mathbf{x} + B\mathbf{u}$$

yields: $\mathbf{u} = -K\mathbf{x}$
hence: $\boldsymbol{\pi}(\mathbf{x}) = -K\mathbf{x}$

Stochastic

$$\boldsymbol{\pi}[\mathbf{u} \,|\, \mathbf{x}]$$

Meaning: probability of selecting input $\mathbf{u}$ *provided* that we are in state $\mathbf{x}$

Why?

- unsure of the policy, "try" different decisions
- taking random actions can be better, e.g. Rock-Paper-Scissor. Not an MDP though, but a POMDP

Policy - How to define decisions?

Deterministic

$$\mathbf{u} = \boldsymbol{\pi}(\mathbf{x})$$

Meaning: what input $\mathbf{u}$ do we select *provided* that we are in state $\mathbf{x}$

Example LQR:

$$L(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\mathbf{x}^\top Q\mathbf{x} + \frac{1}{2}\mathbf{u}^\top R\mathbf{u}$$

$$\mathbf{x}_+ = A\mathbf{x} + B\mathbf{u}$$

yields: $\mathbf{u} = -K\mathbf{x}$
hence: $\boldsymbol{\pi}(\mathbf{x}) = -K\mathbf{x}$

Stochastic

$$\boldsymbol{\pi}[\mathbf{u} \,|\, \mathbf{x}]$$

Meaning: probability of selecting input $\mathbf{u}$ *provided* that we are in state $\mathbf{x}$

Why?

- unsure of the policy, "try" different decisions
- taking random actions can be better, e.g. Rock-Paper-Scissor. Not an MDP though, but a POMDP
- more on that later...

Policy - How to define decisions?

Deterministic

$$\mathbf{u} = \boldsymbol{\pi}(\mathbf{x})$$

Meaning: what input $\mathbf{u}$ do we select *provided* that we are in state $\mathbf{x}$

Example LQR:

$$L(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\mathbf{x}^\top Q\mathbf{x} + \frac{1}{2}\mathbf{u}^\top R\mathbf{u}$$

$$\mathbf{x}_+ = A\mathbf{x} + B\mathbf{u}$$

yields: $\mathbf{u} = -K\mathbf{x}$
hence: $\boldsymbol{\pi}(\mathbf{x}) = -K\mathbf{x}$

Stochastic

$$\boldsymbol{\pi}[\mathbf{u} \,|\, \mathbf{x}]$$

Meaning: probability of selecting input $\mathbf{u}$ *provided* that we are in state $\mathbf{x}$

Why?

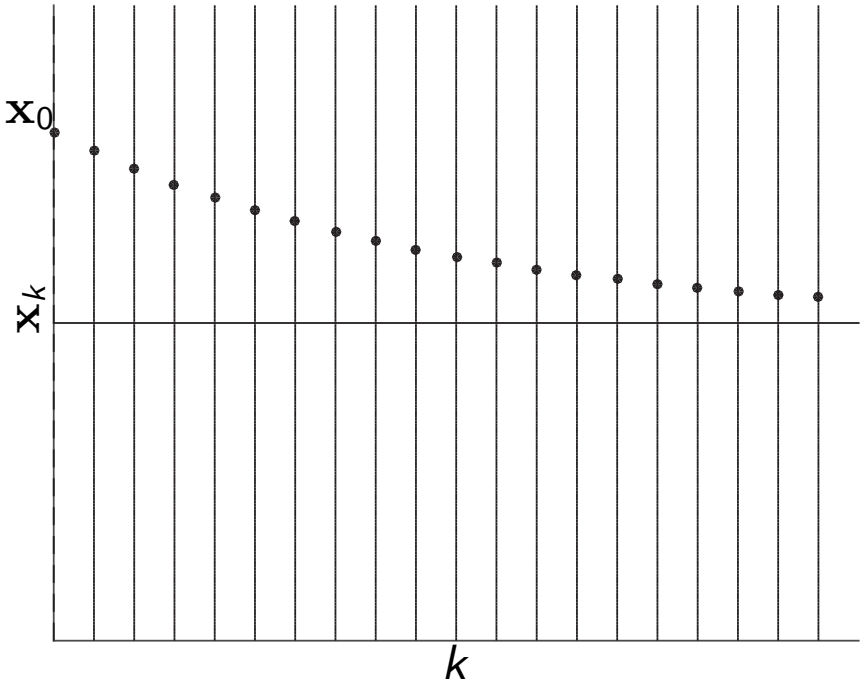
- unsure of the policy, "try" different decisions
- taking random actions can be better, e.g. Rock-Paper-Scissor. Not an MDP though, but a POMDP
- more on that later...

Remarks

- MDPs always have a deterministic optimal policy
- Optimal policy may not be unique
- If several actions are equally good, then a random choice between them is still optimum, hence some MDPs admit stochastic optimal policies

Unpacking the Markov Chain

Unpacking the Markov Chain

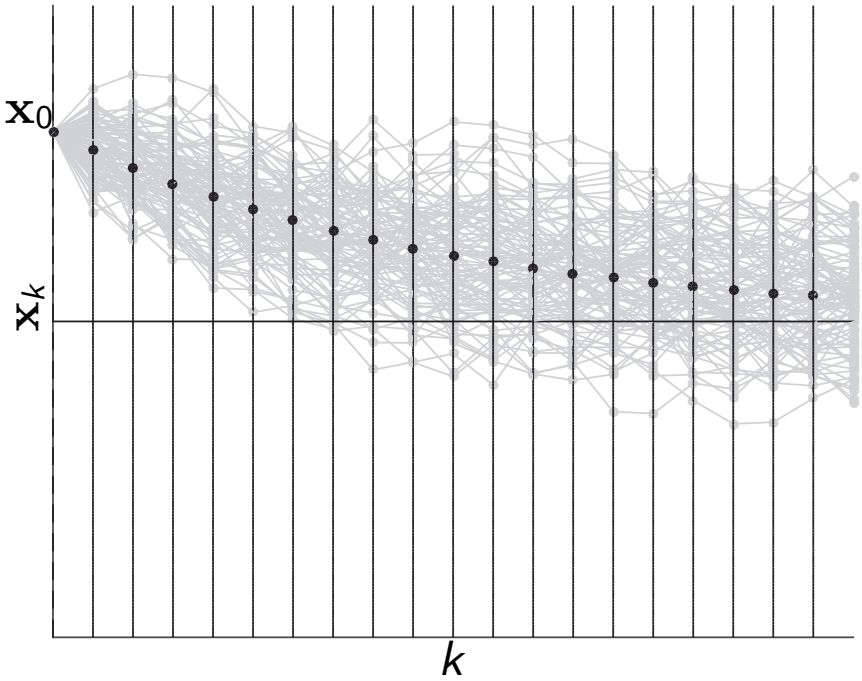


Deterministic state transition

$$\mathbf{x}_+ = \mathbf{f}(\mathbf{x}, \mathbf{u})$$

$$\mathbb{P}[\mathbf{x}_+ \mid \mathbf{x}, \mathbf{u}] = \delta(\mathbf{x}_+ - \mathbf{f}(\mathbf{x}, \mathbf{u}))$$

Unpacking the Markov Chain



Stochastic state transition

$$\mathbf{x}, \mathbf{u} \to \mathbf{x}_+ \quad \text{defined by} \quad \mathbb{P}\left[\mathbf{x}_+ \,|\, \mathbf{x}, \mathbf{u}\right]$$

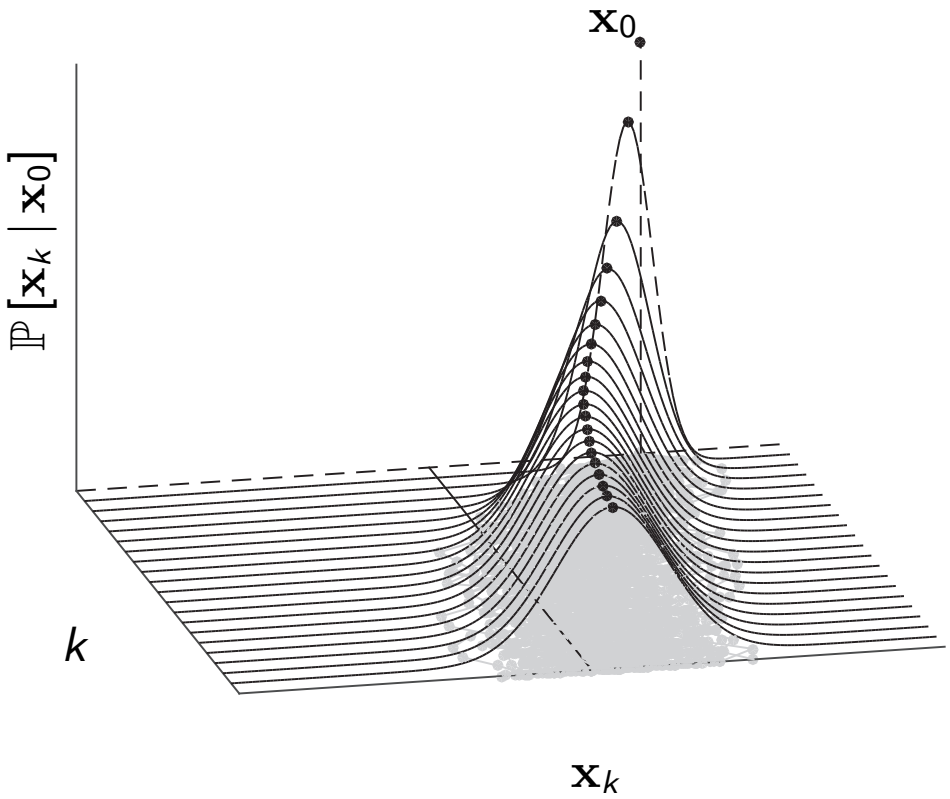
Policy

$$\mathbf{u} = \boldsymbol{\pi}(\mathbf{x}), \quad \text{or} \quad \boldsymbol{\pi}[\mathbf{u} \,|\, \mathbf{x}]$$

Transtion $\mathbf{x}_0 \to \mathbf{x}_k$ is stochastic:

$$\mathbb{P}\left[\mathbf{x}_k \,|\, \mathbf{x}_0\right]$$

Unpacking the Markov Chain



Stochastic state transition

$$\mathbf{x}, \mathbf{u} \to \mathbf{x}_+ \quad \text{defined by} \quad \mathbb{P}[\mathbf{x}_+ \,|\, \mathbf{x}, \mathbf{u}]$$

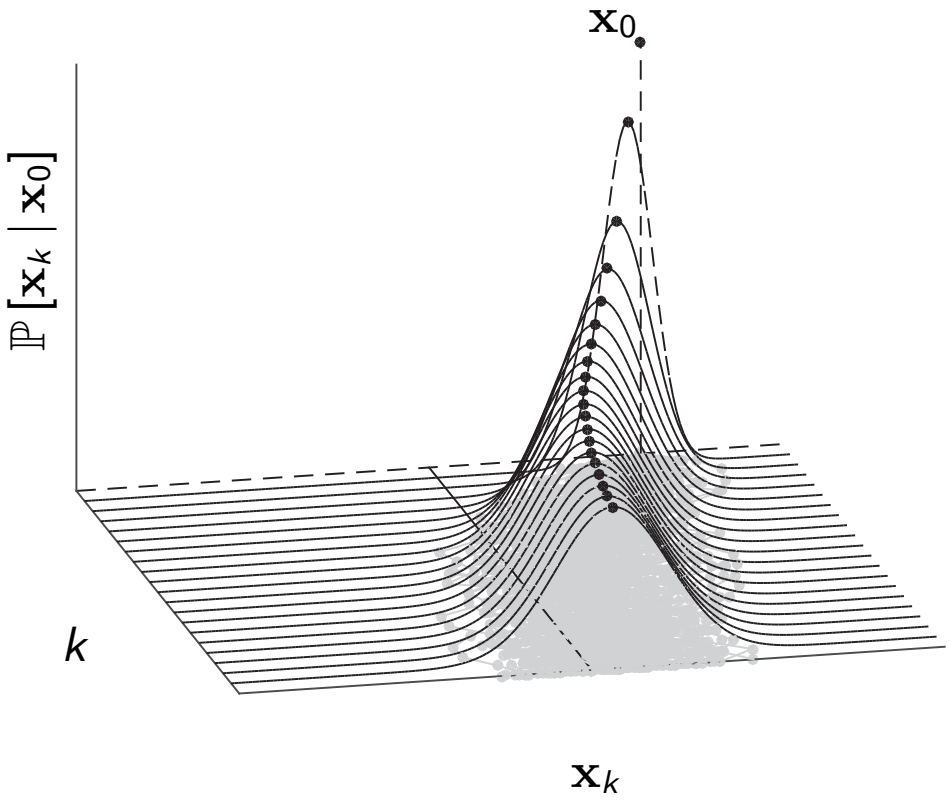
Policy

$$\mathbf{u} = \boldsymbol{\pi}(\mathbf{x}), \quad \text{or} \quad \boldsymbol{\pi}[\mathbf{u} \,|\, \mathbf{x}]$$

Transtion $\mathbf{x}_0 \to \mathbf{x}_k$ is stochastic:

$$\mathbb{P}[\mathbf{x}_k \,|\, \mathbf{x}_0]$$

Unpacking the Markov Chain



Stochastic state transition

$$\mathbf{x}, \mathbf{u} \to \mathbf{x}_+ \quad \text{defined by} \quad \mathbb{P}[\mathbf{x}_+ \,|\, \mathbf{x}, \mathbf{u}]$$

Policy

$$\mathbf{u} = \boldsymbol{\pi}(\mathbf{x}), \quad \text{or} \quad \boldsymbol{\pi}[\mathbf{u} \,|\, \mathbf{x}]$$

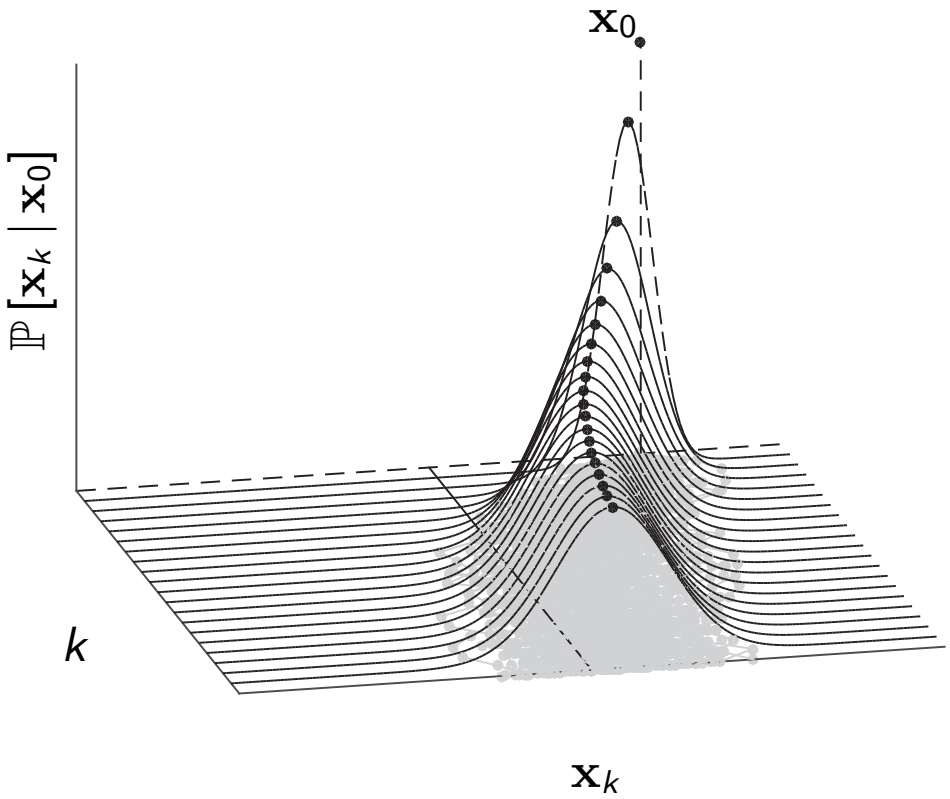
Transtion $\mathbf{x}_0 \to \mathbf{x}_k$ is stochastic:

$$\mathbb{P}[\mathbf{x}_k \,|\, \mathbf{x}_0]$$

1-step distribution:

$$\mathbb{P}[\mathbf{x}_1 \,|\, \mathbf{x}_0,\ \mathbf{u}_0 = \boldsymbol{\pi}(\mathbf{x}_0)] \equiv \mathbb{P}[\mathbf{x}_1 \,|\, \mathbf{x}_0,\ \boldsymbol{\pi}(\mathbf{x}_0)]$$

Unpacking the Markov Chain



Stochastic state transition

$$\mathbf{x}, \mathbf{u} \to \mathbf{x}_+ \quad \text{defined by} \quad \mathbb{P}[\mathbf{x}_+ \,|\, \mathbf{x}, \mathbf{u}]$$

Policy

$$\mathbf{u} = \boldsymbol{\pi}(\mathbf{x}), \quad \text{or} \quad \boldsymbol{\pi}[\mathbf{u} \,|\, \mathbf{x}]$$

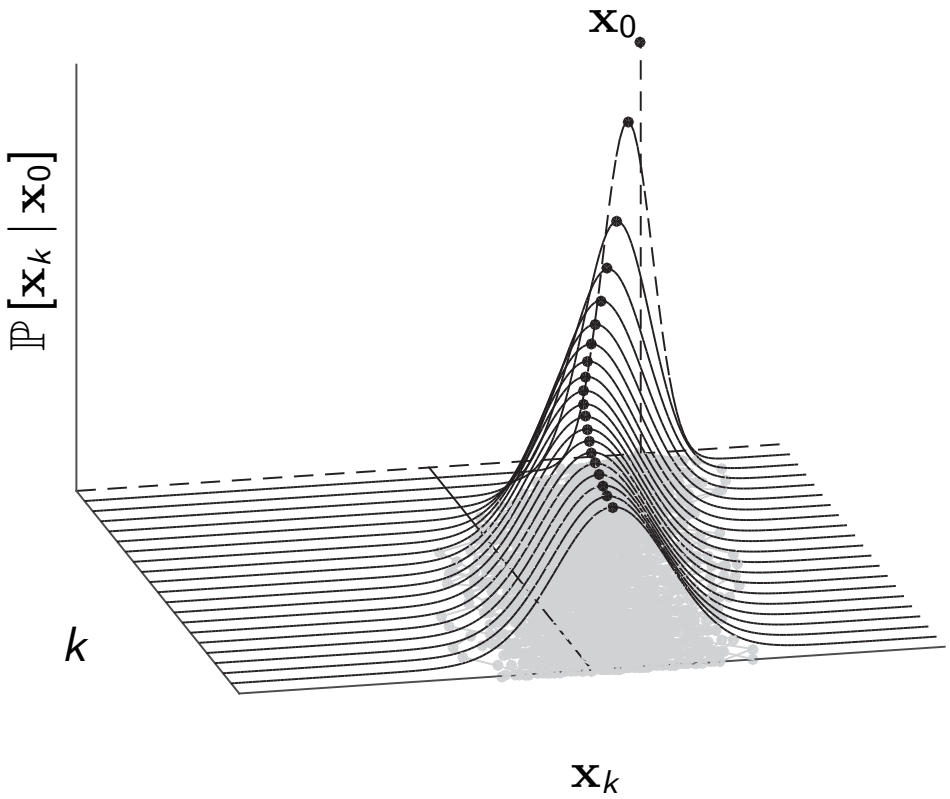
Transtion $\mathbf{x}_0 \to \mathbf{x}_k$ is stochastic:

$$\mathbb{P}[\mathbf{x}_k \,|\, \mathbf{x}_0]$$

2-step distribution:

$$\mathbb{P}[\mathbf{x}_2 \,|\, \mathbf{x}_0] = \int \mathbb{P}[\mathbf{x}_2 \,|\, \mathbf{x}_1,\, \boldsymbol{\pi}(\mathbf{x}_1)]\, \mathbb{P}[\mathbf{x}_1 \,|\, \mathbf{x}_0,\, \mathbf{u}_0]\, \mathrm{d}\mathbf{x}_1$$

Unpacking the Markov Chain



Stochastic state transition

$$\mathbf{x}, \mathbf{u} \to \mathbf{x}_+ \quad \text{defined by} \quad \mathbb{P}[\mathbf{x}_+ \,|\, \mathbf{x}, \mathbf{u}]$$

Policy

$$\mathbf{u} = \boldsymbol{\pi}(\mathbf{x}), \quad \text{or} \quad \boldsymbol{\pi}[\mathbf{u} \,|\, \mathbf{x}]$$

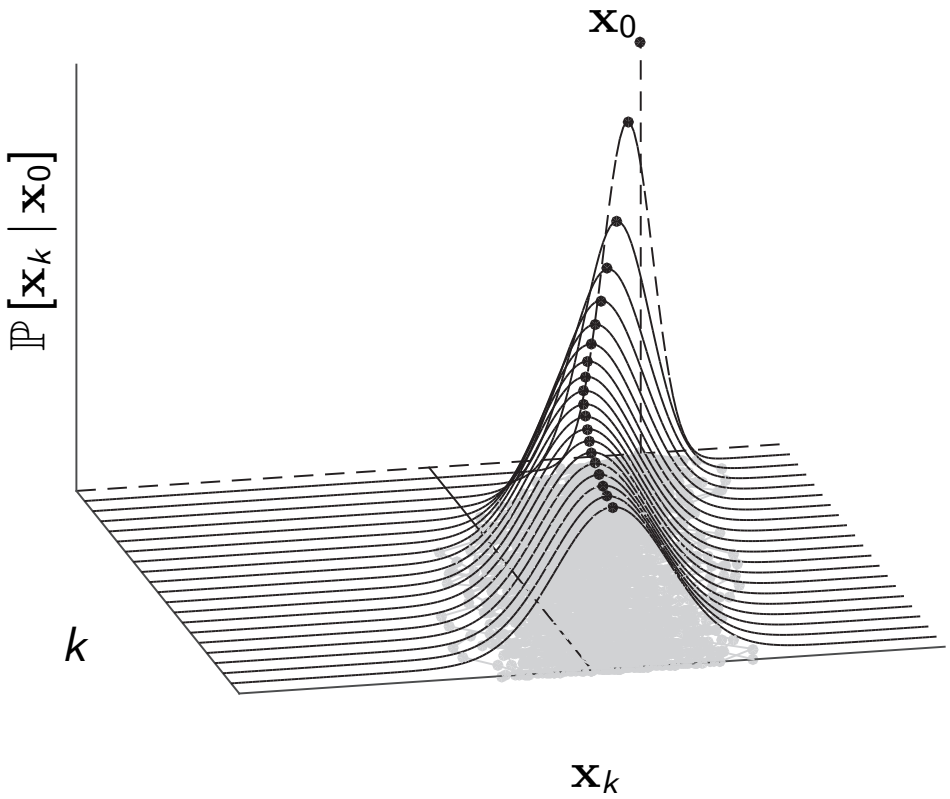
Transtion $\mathbf{x}_0 \to \mathbf{x}_k$ is stochastic:

$$\mathbb{P}[\mathbf{x}_k \,|\, \mathbf{x}_0]$$

3-step distribution:

$$\mathbb{P}[\mathbf{x}_3 \,|\, \mathbf{x}_0] = \int \mathbb{P}[\mathbf{x}_3 \,|\, \mathbf{x}_2,\, \boldsymbol{\pi}(\mathbf{x}_2)]\, \mathbb{P}[\mathbf{x}_2 \,|\, \mathbf{x}_1,\, \boldsymbol{\pi}(\mathbf{x}_1)]\, \mathbb{P}[\mathbf{x}_1 \,|\, \mathbf{x}_0,\, \boldsymbol{\pi}(\mathbf{x}_0)]\, \mathrm{d}\mathbf{x}_1 \mathrm{d}\mathbf{x}_2$$

Unpacking the Markov Chain



Stochastic state transition

$$\mathbf{x}, \mathbf{u} \to \mathbf{x}_+ \quad \text{defined by} \quad \mathbb{P}[\mathbf{x}_+ \,|\, \mathbf{x}, \mathbf{u}]$$

Policy

$$\mathbf{u} = \boldsymbol{\pi}(\mathbf{x}), \quad \text{or} \quad \boldsymbol{\pi}[\mathbf{u} \,|\, \mathbf{x}]$$

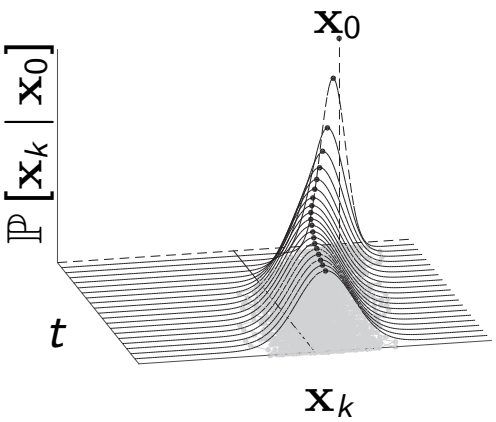
Transtion $\mathbf{x}_0 \to \mathbf{x}_k$ is stochastic:

$$\mathbb{P}[\mathbf{x}_k \,|\, \mathbf{x}_0]$$

n-step distribution:

$$\mathbb{P}[\mathbf{x}_k \,|\, \mathbf{x}_0] = \int \prod_{i=0}^{k-1} \mathbb{P}[\mathbf{x}_{i+1} \,|\, \mathbf{x}_i,\, \boldsymbol{\pi}(\mathbf{x}_i)]\, \mathrm{d}\mathbf{x}_1 \ldots \mathrm{d}\mathbf{x}_{k-1}$$

Unpacking the Markov Chain



Stationary distribution: how often are the different states visited under a given policy, after the Markov chain reaches its stationary dynamics?

$$\mathbb{P}[\mathbf{x}_\infty \mid \mathbf{x}_0]$$

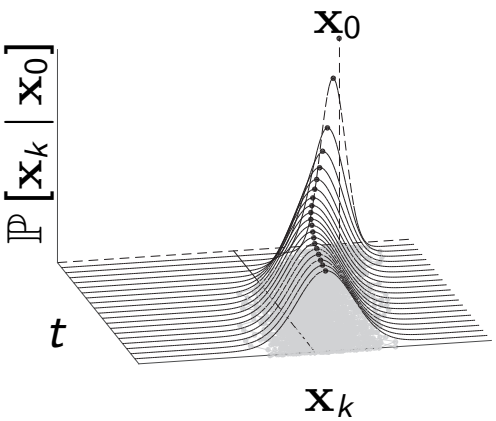
Markov chain distribution: $\tau_{\boldsymbol{\pi}}$ describes the distribution of trajectories, i.e. the probability distribution that specific trajectories $\mathbf{x}_{0,\ldots,\infty}$ are observed under policy $\boldsymbol{\pi}$

- Distribution $\mathbf{x}_0 \to \mathbf{x}_k$

$$\mathbb{P}[\mathbf{x}_k \mid \mathbf{x}_0] = \int \prod_{i=0}^{k-1} \mathbb{P}[\mathbf{x}_{i+1} \mid \mathbf{x}_i,\ \boldsymbol{\pi}(\mathbf{x}_i)] \cdot \mathrm{d}\mathbf{x}_1 \ldots \mathrm{d}\mathbf{x}_{k-1}$$

- Expectation over distribution $\tau_{\boldsymbol{\pi}}$, e.g. performance of policy $\boldsymbol{\pi}$:

Unpacking the Markov Chain



Stationary distribution: how often are the different states visited under a given policy, after the Markov chain reaches its stationary dynamics?

$$\mathbb{P}[\mathbf{x}_\infty \mid \mathbf{x}_0]$$

Markov chain distribution: $\tau_{\boldsymbol{\pi}}$ describes the distribution of trajectories, i.e. the probability distribution that specific trajectories $\mathbf{x}_{0,\ldots,\infty}$ are observed under policy $\boldsymbol{\pi}$

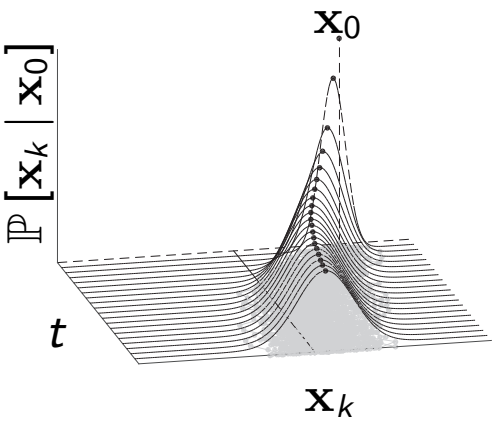
- Distribution $\mathbf{x}_0 \to \mathbf{x}_k$

$$\mathbb{P}[\mathbf{x}_k \mid \mathbf{x}_0] = \int \prod_{i=0}^{k-1} \mathbb{P}[\mathbf{x}_{i+1} \mid \mathbf{x}_i,\ \boldsymbol{\pi}(\mathbf{x}_i)] \cdot \mathrm{d}\mathbf{x}_1 \ldots \mathrm{d}\mathbf{x}_{k-1}$$

- Expectation over distribution $\tau_{\boldsymbol{\pi}}$, e.g. performance of policy $\boldsymbol{\pi}$:

$$J(\boldsymbol{\pi}) = \mathbb{E}_{\tau_{\boldsymbol{\pi}}}[L(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x}))] = \int \sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \boldsymbol{\pi}(\mathbf{x}_k)) \mathbb{P}[\mathbf{x}_k \mid \mathbf{x}_0] \mathbb{P}[\mathbf{x}_0] \, \mathrm{d}\mathbf{x}_0 \mathrm{d}\mathbf{x}_k$$

Unpacking the Markov Chain



Stationary distribution: how often are the different states visited under a given policy, after the Markov chain reaches its stationary dynamics?

$$\mathbb{P}[\mathbf{x}_\infty \mid \mathbf{x}_0]$$

Markov chain distribution: $\tau_{\boldsymbol{\pi}}$ describes the distribution of trajectories, i.e. the probability distribution that specific trajectories $\mathbf{x}_{0,\ldots,\infty}$ are observed under policy $\boldsymbol{\pi}$

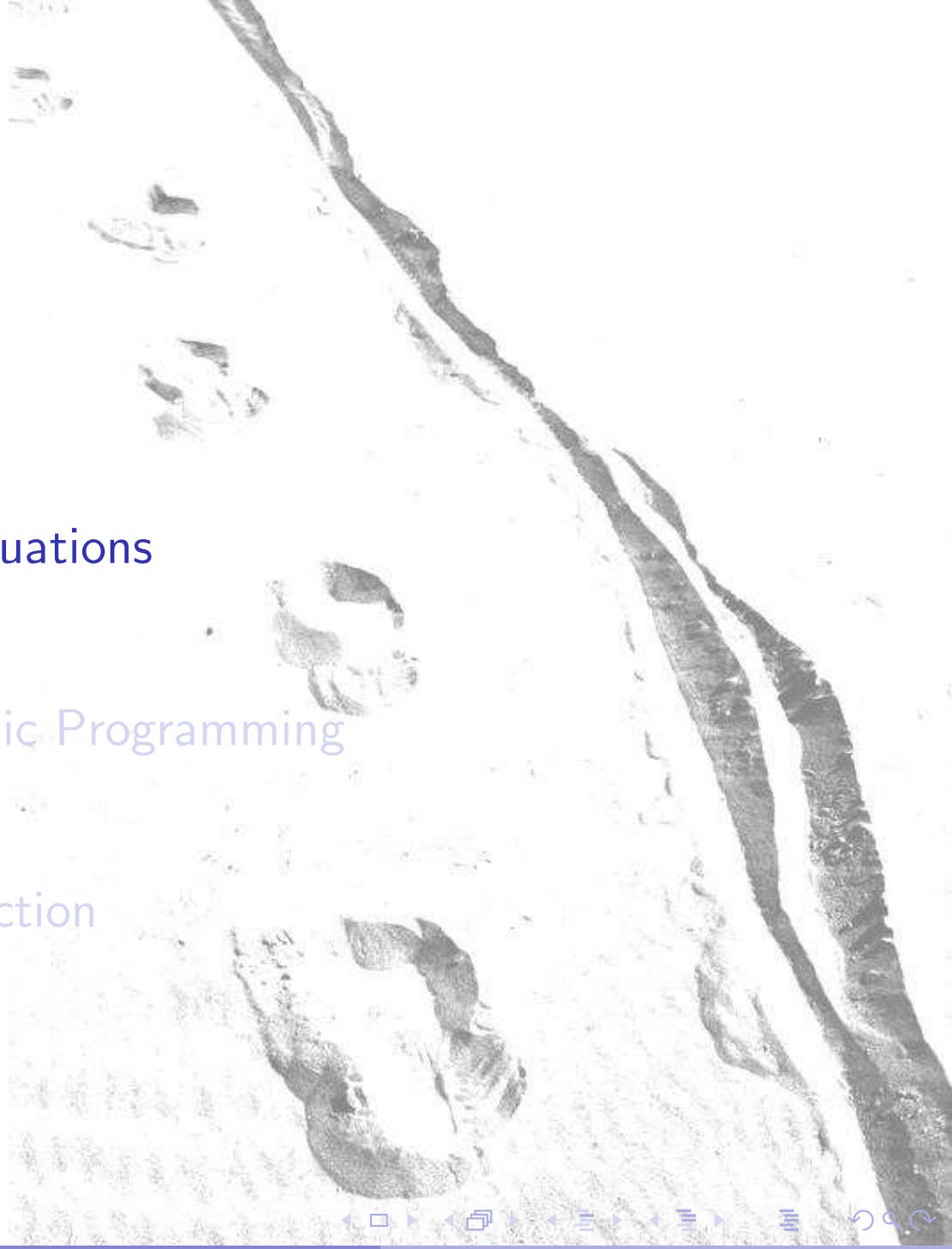
- Distribution $\mathbf{x}_0 \to \mathbf{x}_k$

$$\mathbb{P}[\mathbf{x}_k \mid \mathbf{x}_0] = \int \prod_{i=0}^{k-1} \mathbb{P}[\mathbf{x}_{i+1} \mid \mathbf{x}_i,\ \boldsymbol{\pi}(\mathbf{x}_i)] \cdot \mathrm{d}\mathbf{x}_1 \ldots \mathrm{d}\mathbf{x}_{k-1}$$

- Expectation over distribution $\tau_{\boldsymbol{\pi}}$, e.g. performance of policy $\boldsymbol{\pi}$:

$$J(\boldsymbol{\pi}) = \sum_{k=0}^{\infty} \int \gamma^k L(\mathbf{x}_k, \boldsymbol{\pi}(\mathbf{x}_k)) \prod_{i=1}^{k} \mathbb{P}[\mathbf{x}_i \mid \mathbf{x}_{i-1},\ \boldsymbol{\pi}(\mathbf{x}_{i-1})]\, \mathbb{P}[\mathbf{x}_0] \cdot \mathrm{d}\mathbf{x}_0 \ldots \mathrm{d}\mathbf{x}_{k-1}$$

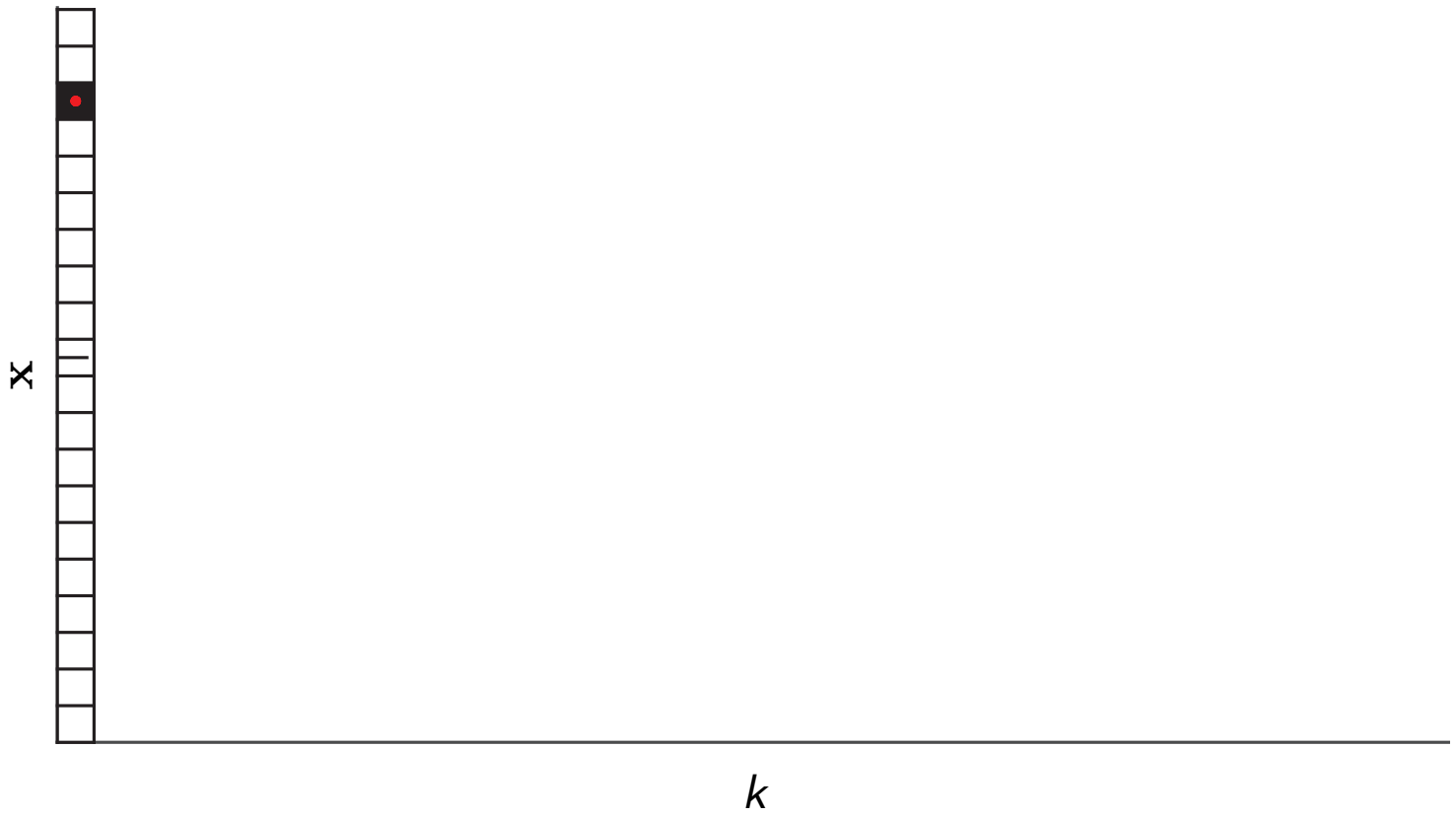
Outline



Let's use the following example (for illustration purposes)

- State $\mathbf{x} \in \mathbb{N}$ and input $\mathbf{u} \in \mathbb{N}$ locked on a grid
- Input $\mathbf{u}$ can only move $\mathbf{x}$ on the grid
- Stage cost: $L(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\mathbf{x}^2 + \frac{1}{2}\mathbf{u}^2$
- Discount $\gamma = 0.9$

Dynamics: $\mathbf{x}_+ = \mathbf{x} + \mathbf{u} + \text{round}(\mathbf{e})$ where $\mathbf{e} \sim \mathcal{N}(0, 1)$

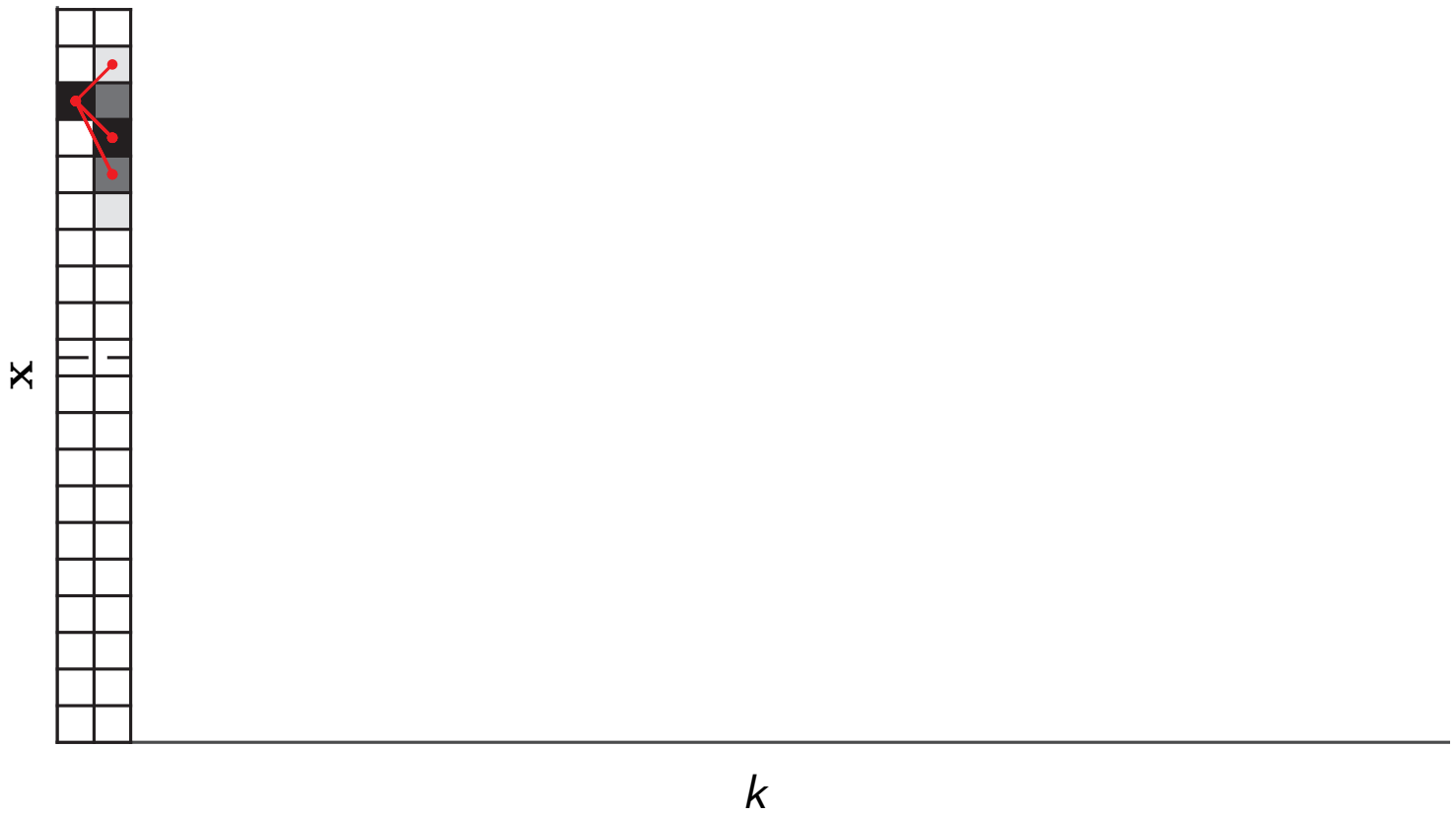


e.g. for
$\mathbf{u} = \text{round}\left(-\frac{\mathbf{x}}{10}\right)$

Let's use the following example (for illustration purposes)

- State $\mathbf{x} \in \mathbb{N}$ and input $\mathbf{u} \in \mathbb{N}$ locked on a grid
- Input $\mathbf{u}$ can only move $\mathbf{x}$ on the grid
- Stage cost: $L(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\mathbf{x}^2 + \frac{1}{2}\mathbf{u}^2$
- Discount $\gamma = 0.9$

Dynamics: $\mathbf{x}_+ = \mathbf{x} + \mathbf{u} + \text{round}(\mathbf{e})$ where $\mathbf{e} \sim \mathcal{N}(0, 1)$

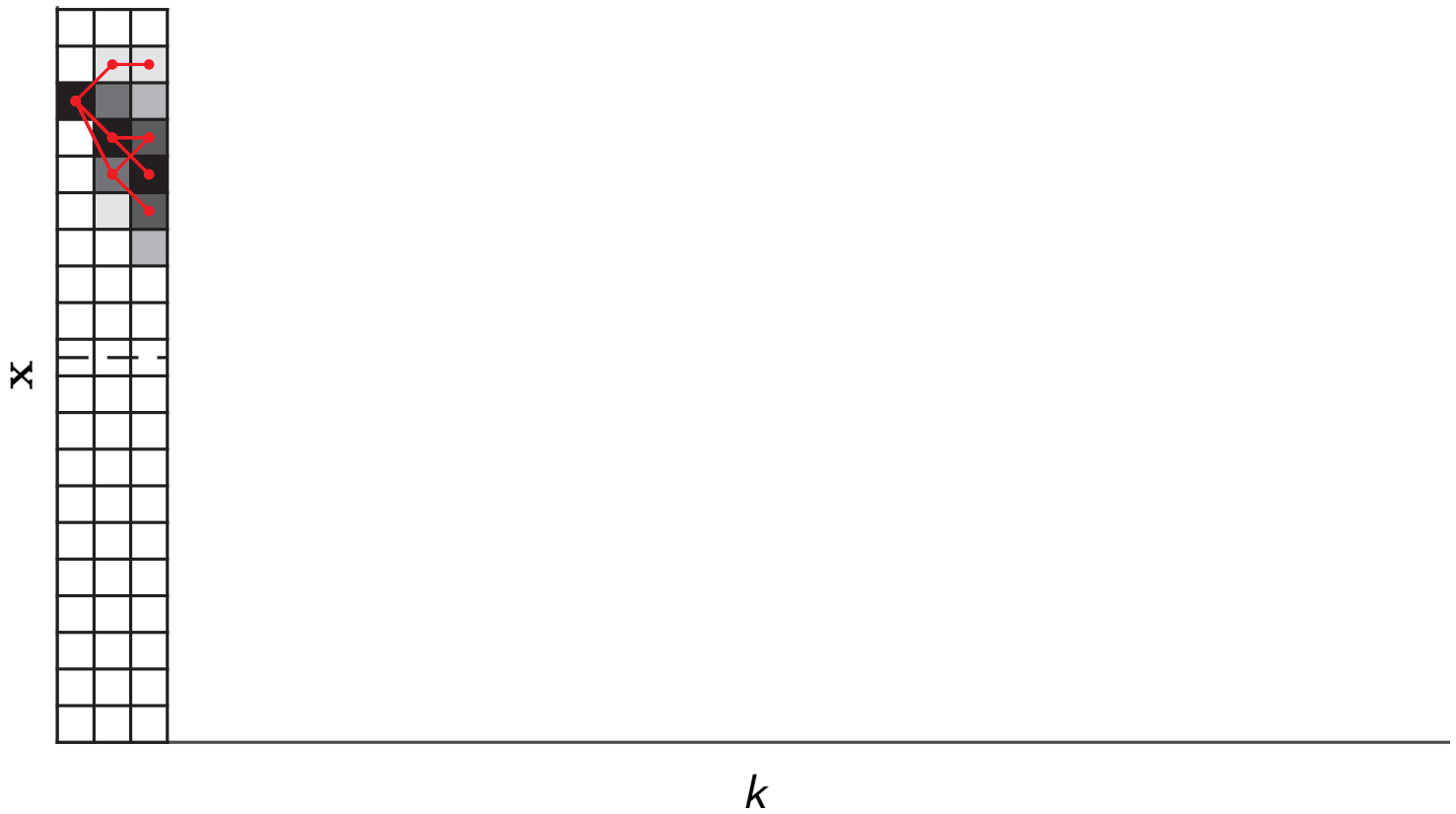


e.g. for
$\mathbf{u} = \text{round}\left(-\frac{\mathbf{x}}{10}\right)$

Let's use the following example (for illustration purposes)

- State $\mathbf{x} \in \mathbb{N}$ and input $\mathbf{u} \in \mathbb{N}$ locked on a grid
- Input $\mathbf{u}$ can only move $\mathbf{x}$ on the grid
- Stage cost: $L(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\mathbf{x}^2 + \frac{1}{2}\mathbf{u}^2$
- Discount $\gamma = 0.9$

Dynamics: $\mathbf{x}_+ = \mathbf{x} + \mathbf{u} + \mathrm{round}(\mathbf{e})$ where $\mathbf{e} \sim \mathcal{N}(0, 1)$

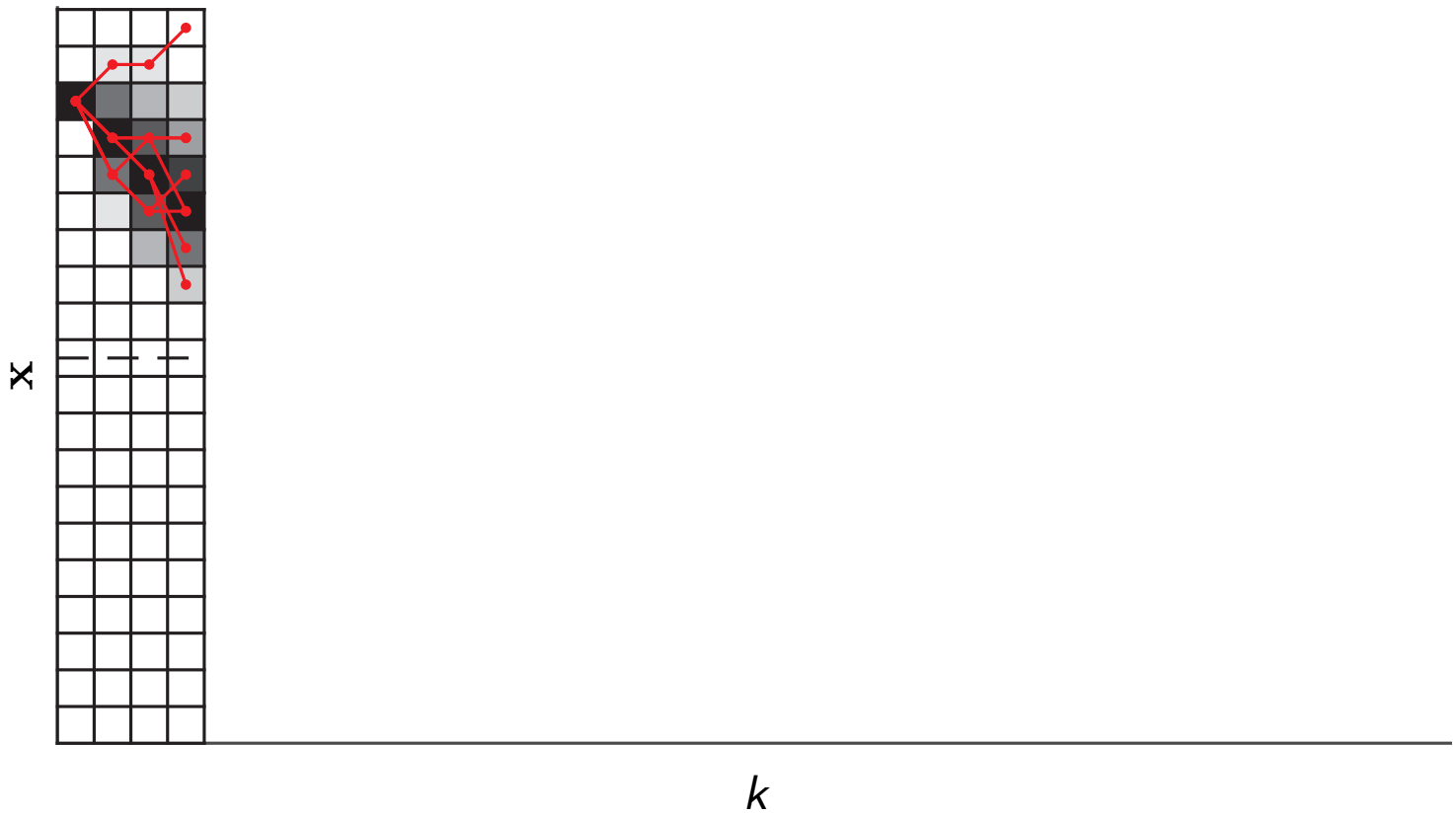


e.g. for
$\mathbf{u} = \mathrm{round}\left(-\frac{\mathbf{x}}{10}\right)$

Let's use the following example (for illustration purposes)

- State $\mathbf{x} \in \mathbb{N}$ and input $\mathbf{u} \in \mathbb{N}$ locked on a grid
- Input $\mathbf{u}$ can only move $\mathbf{x}$ on the grid
- Stage cost: $L(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\mathbf{x}^2 + \frac{1}{2}\mathbf{u}^2$
- Discount $\gamma = 0.9$

Dynamics: $\mathbf{x}_+ = \mathbf{x} + \mathbf{u} + \text{round}(\mathbf{e})$ where $\mathbf{e} \sim \mathcal{N}(0, 1)$

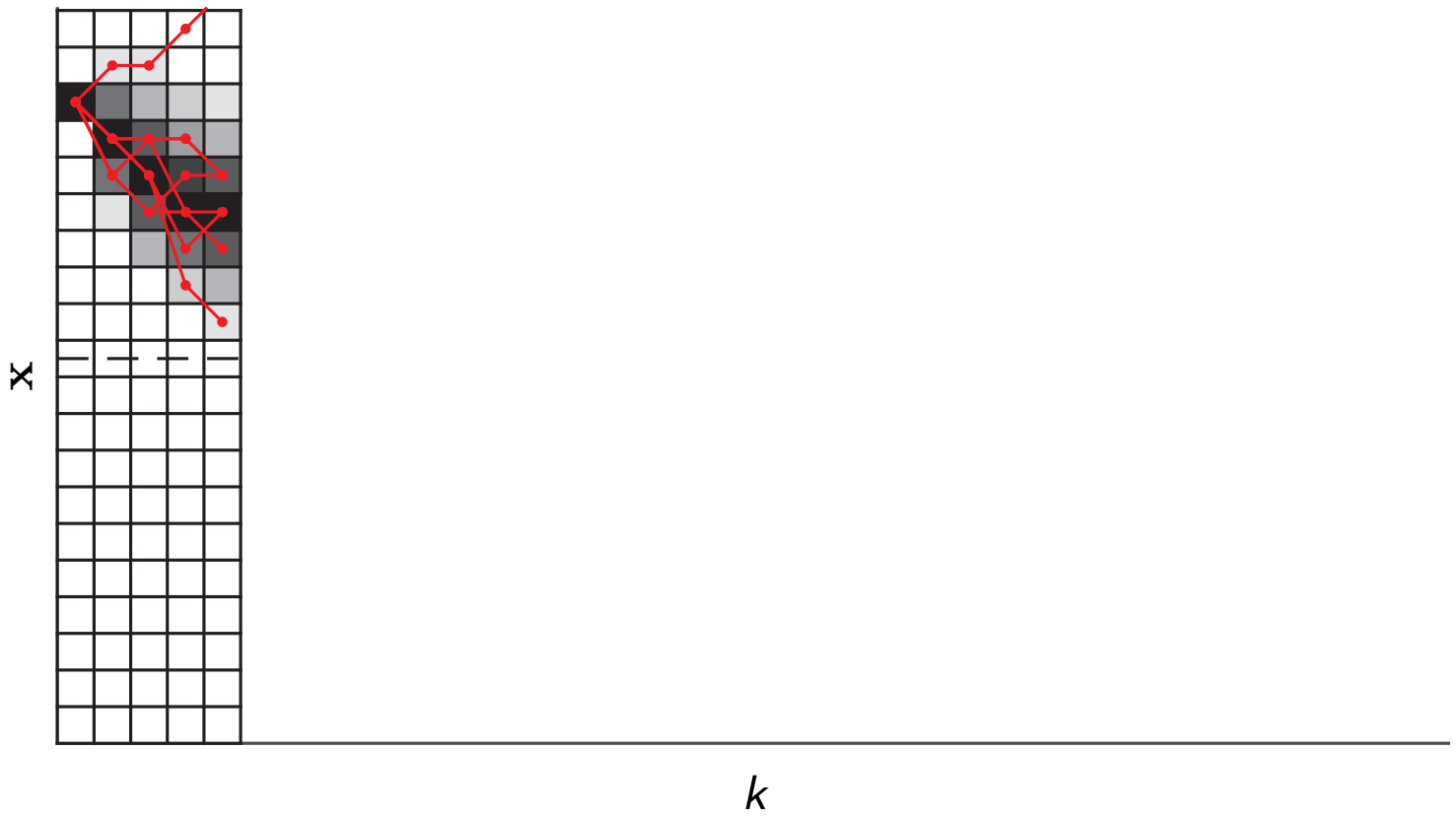


e.g. for
$\mathbf{u} = \text{round}\left(-\frac{\mathbf{x}}{10}\right)$

Let's use the following example (for illustration purposes)

- State $\mathbf{x} \in \mathbb{N}$ and input $\mathbf{u} \in \mathbb{N}$ locked on a grid
- Input $\mathbf{u}$ can only move $\mathbf{x}$ on the grid
- Stage cost: $L(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\mathbf{x}^2 + \frac{1}{2}\mathbf{u}^2$
- Discount $\gamma = 0.9$

Dynamics: $\mathbf{x}_+ = \mathbf{x} + \mathbf{u} + \text{round}(\mathbf{e})$ where $\mathbf{e} \sim \mathcal{N}(0, 1)$

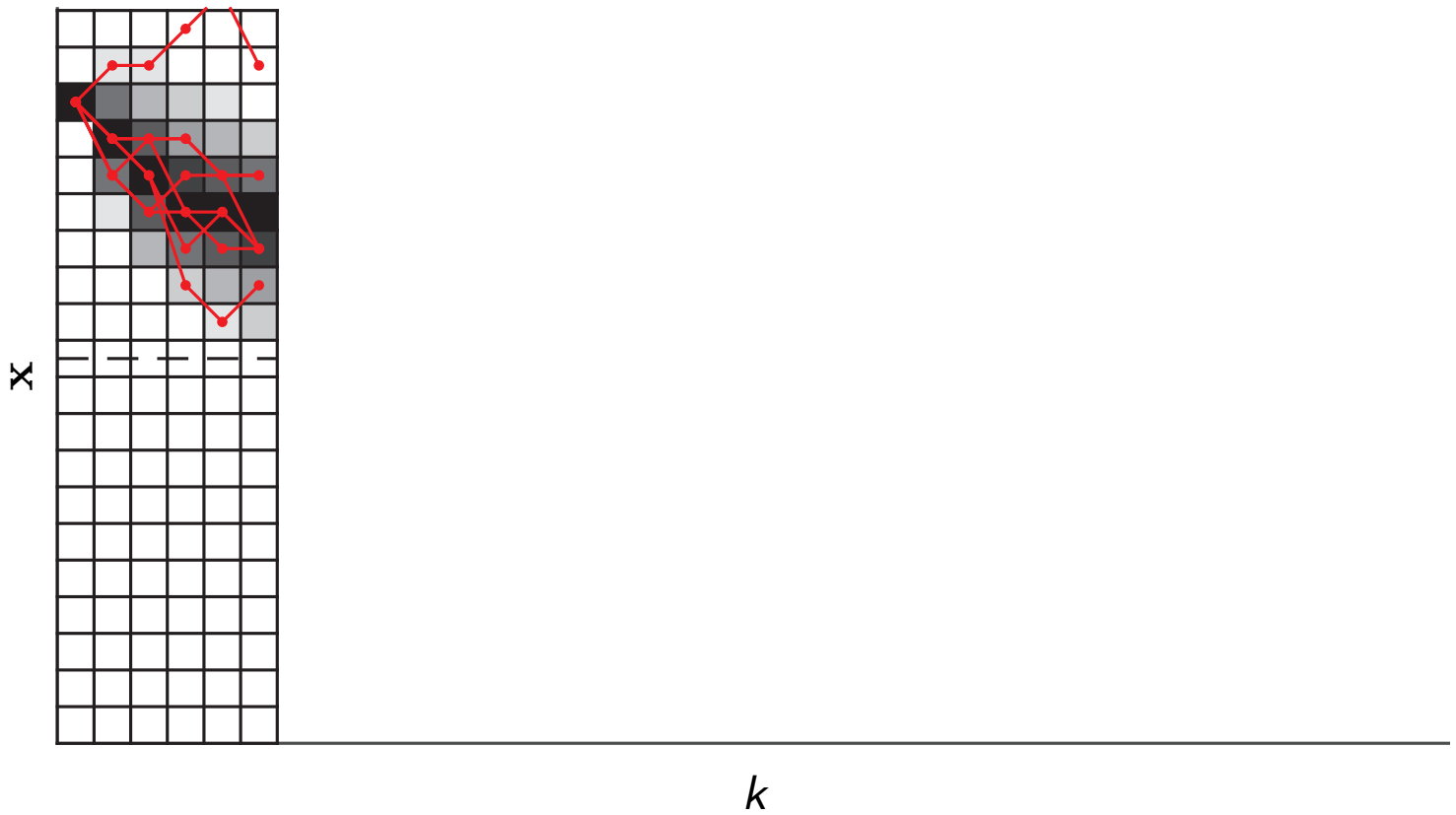


e.g. for
$\mathbf{u} = \text{round}\left(-\frac{\mathbf{x}}{10}\right)$

Let's use the following example (for illustration purposes)

- State $\mathbf{x} \in \mathbb{N}$ and input $\mathbf{u} \in \mathbb{N}$ locked on a grid
- Input $\mathbf{u}$ can only move $\mathbf{x}$ on the grid
- Stage cost: $L(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\mathbf{x}^2 + \frac{1}{2}\mathbf{u}^2$
- Discount $\gamma = 0.9$

Dynamics: $\mathbf{x}_+ = \mathbf{x} + \mathbf{u} + \text{round}(\mathbf{e})$ where $\mathbf{e} \sim \mathcal{N}(0, 1)$

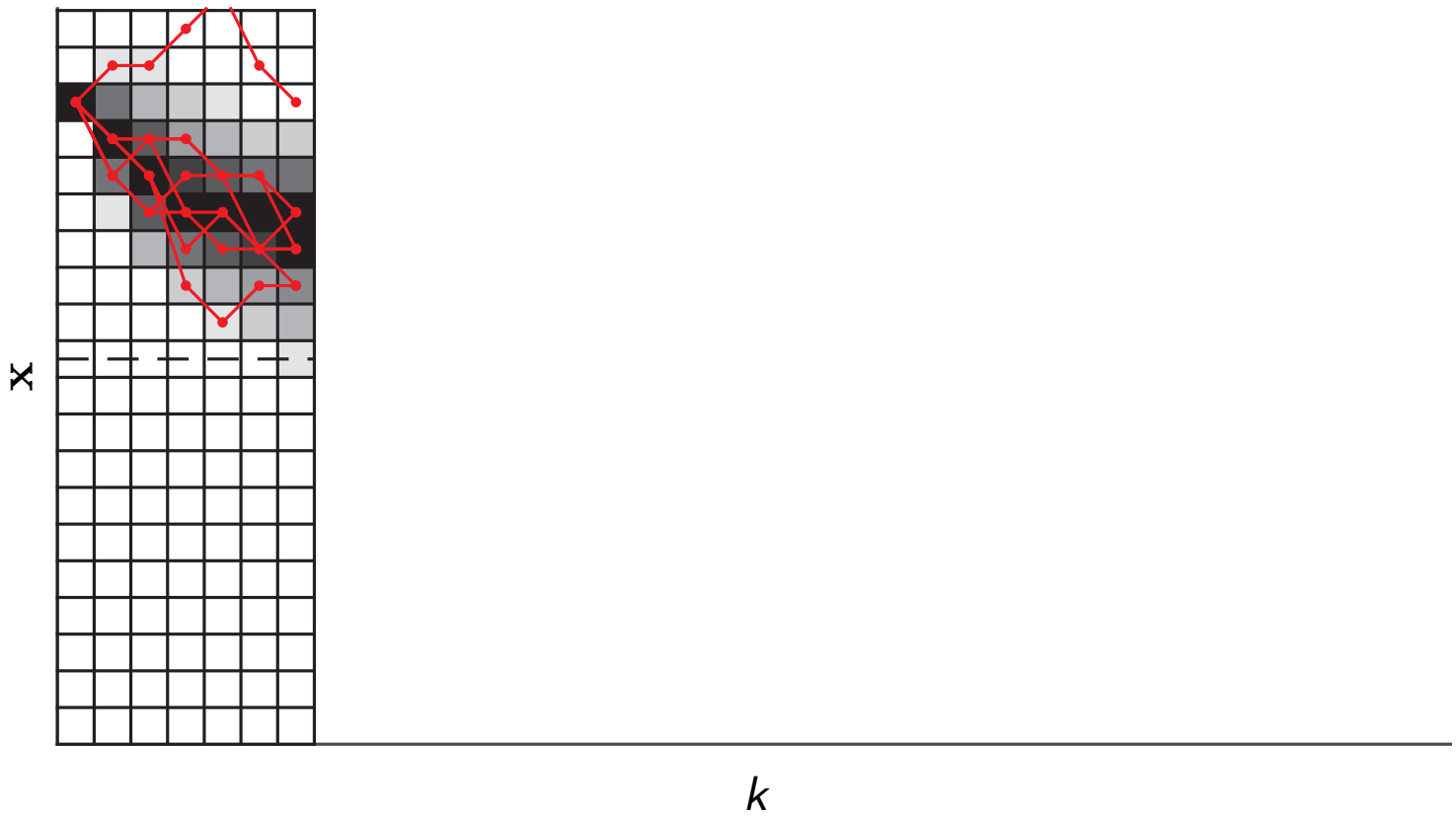


e.g. for
$\mathbf{u} = \text{round}\left(-\frac{\mathbf{x}}{10}\right)$

Let's use the following example (for illustration purposes)

- State $\mathbf{x} \in \mathbb{N}$ and input $\mathbf{u} \in \mathbb{N}$ locked on a grid
- Input $\mathbf{u}$ can only move $\mathbf{x}$ on the grid
- Stage cost: $L(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\mathbf{x}^2 + \frac{1}{2}\mathbf{u}^2$
- Discount $\gamma = 0.9$

Dynamics: $\mathbf{x}_+ = \mathbf{x} + \mathbf{u} + \text{round}(\mathbf{e})$ where $\mathbf{e} \sim \mathcal{N}(0, 1)$

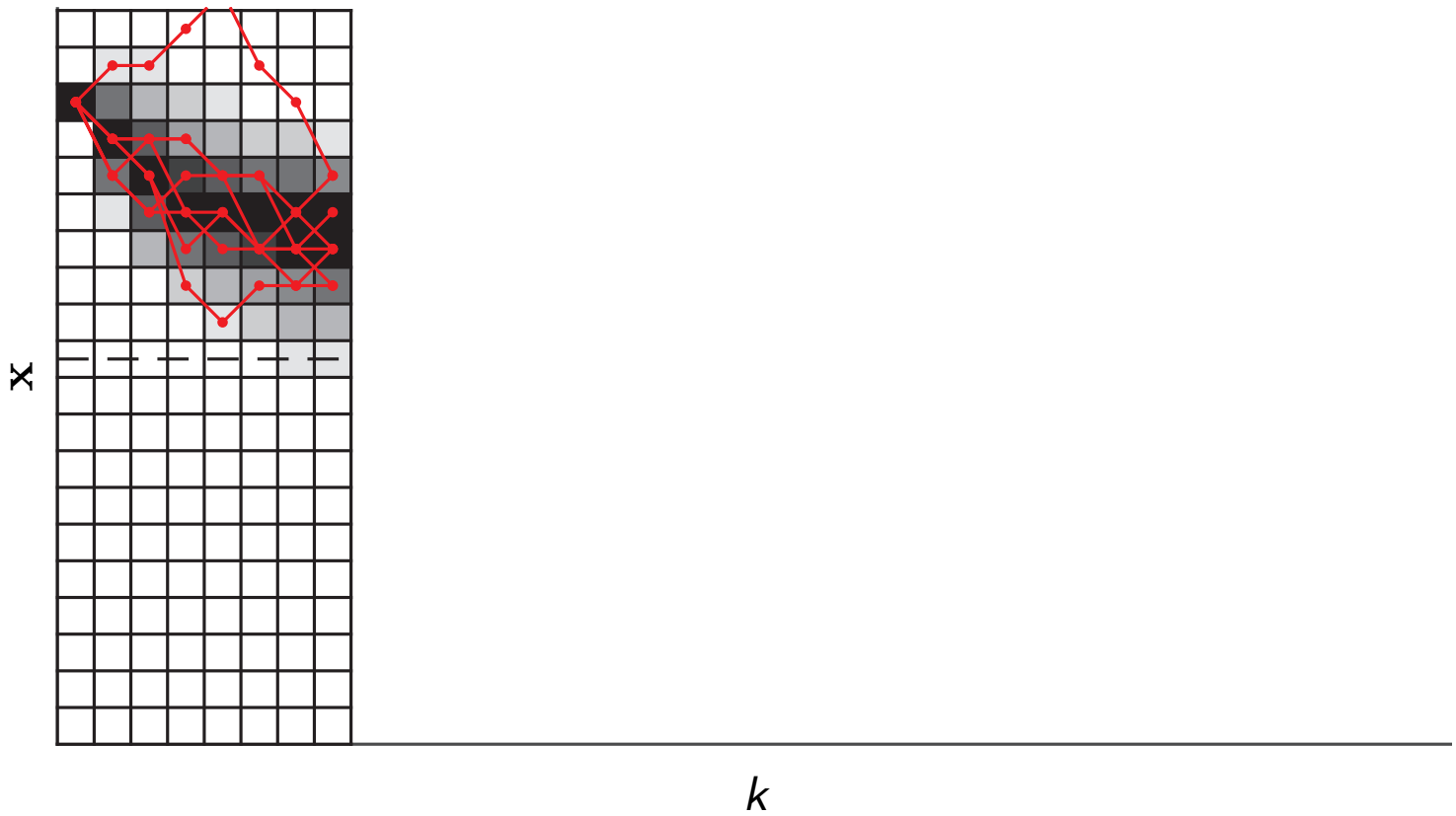


e.g. for
$\mathbf{u} = \text{round}\left(-\frac{\mathbf{x}}{10}\right)$

Let's use the following example (for illustration purposes)

- State $\mathbf{x} \in \mathbb{N}$ and input $\mathbf{u} \in \mathbb{N}$ locked on a grid
- Input $\mathbf{u}$ can only move $\mathbf{x}$ on the grid
- Stage cost: $L(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\mathbf{x}^2 + \frac{1}{2}\mathbf{u}^2$
- Discount $\gamma = 0.9$

Dynamics: $\mathbf{x}_+ = \mathbf{x} + \mathbf{u} + \text{round}(\mathbf{e})$ where $\mathbf{e} \sim \mathcal{N}(0, 1)$

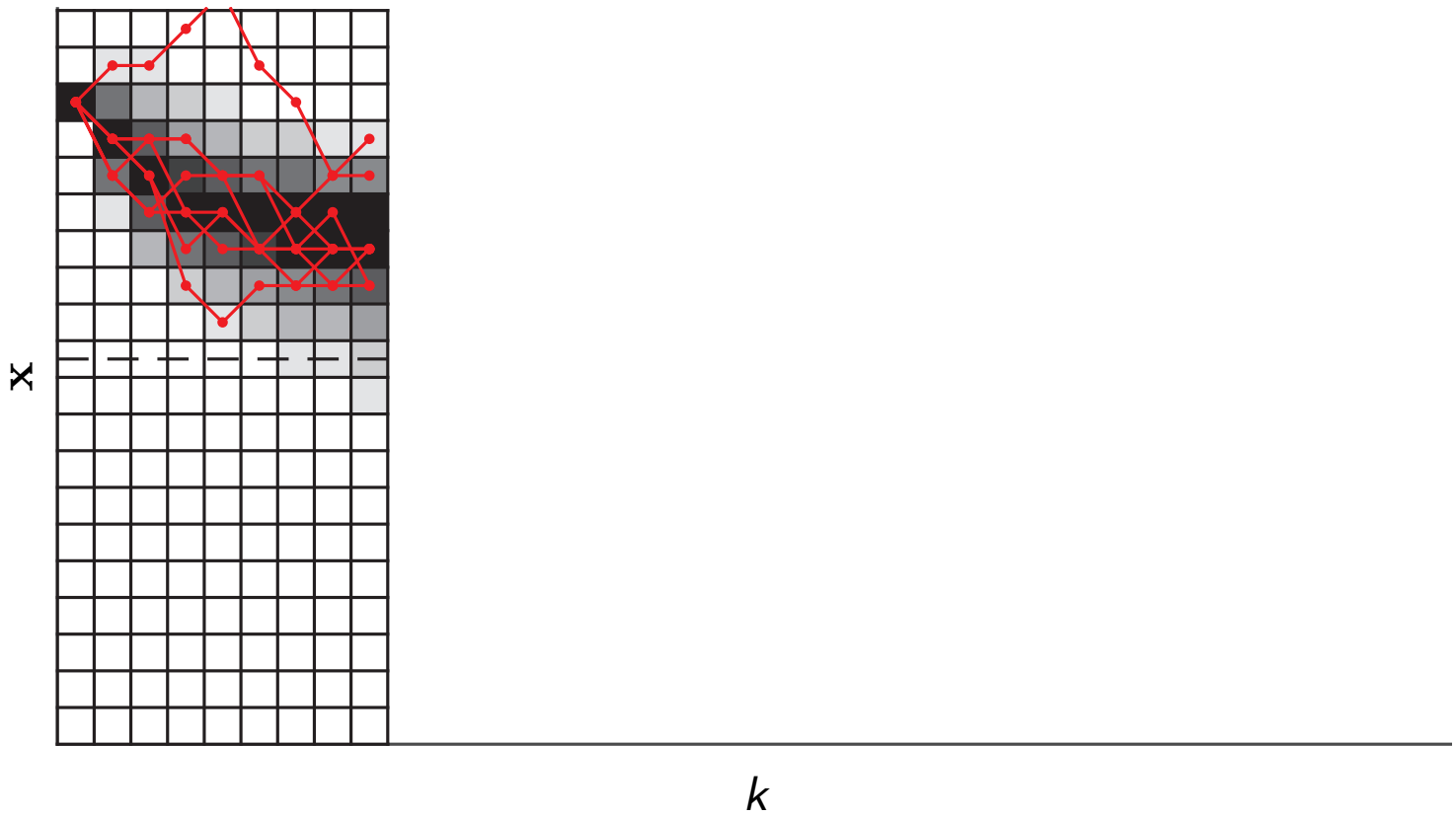


e.g. for
$\mathbf{u} = \text{round}\left(-\frac{\mathbf{x}}{10}\right)$

Let's use the following example (for illustration purposes)

- State $\mathbf{x} \in \mathbb{N}$ and input $\mathbf{u} \in \mathbb{N}$ locked on a grid
- Input $\mathbf{u}$ can only move $\mathbf{x}$ on the grid
- Stage cost: $L(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\mathbf{x}^2 + \frac{1}{2}\mathbf{u}^2$
- Discount $\gamma = 0.9$

Dynamics: $\mathbf{x}_+ = \mathbf{x} + \mathbf{u} + \text{round}(\mathbf{e})$ where $\mathbf{e} \sim \mathcal{N}(0, 1)$

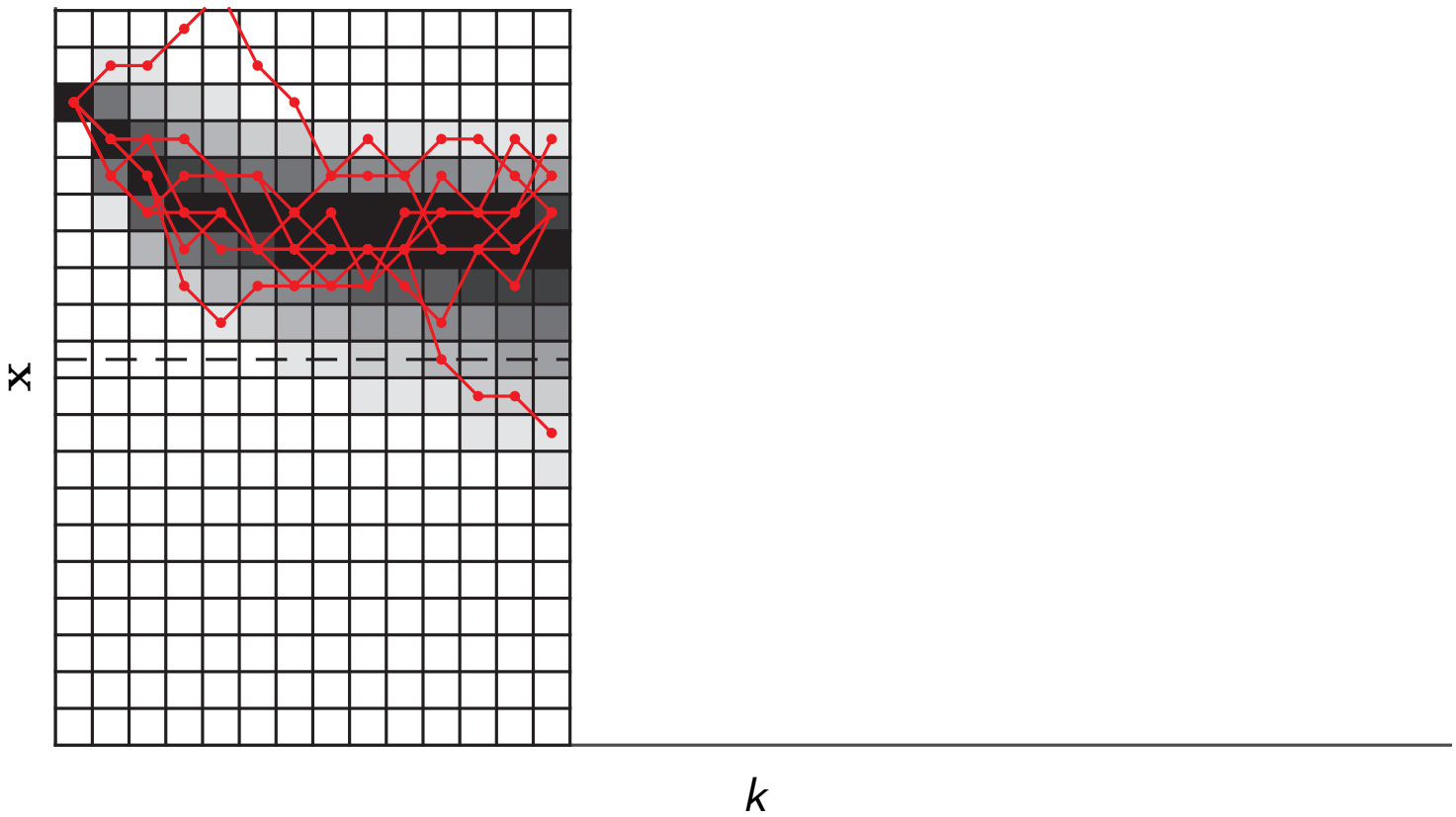


e.g. for
$\mathbf{u} = \text{round}\left(-\frac{\mathbf{x}}{10}\right)$

Let's use the following example (for illustration purposes)

- State $\mathbf{x} \in \mathbb{N}$ and input $\mathbf{u} \in \mathbb{N}$ locked on a grid
- Input $\mathbf{u}$ can only move $\mathbf{x}$ on the grid
- Stage cost: $L(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\mathbf{x}^2 + \frac{1}{2}\mathbf{u}^2$
- Discount $\gamma = 0.9$

Dynamics: $\mathbf{x}_+ = \mathbf{x} + \mathbf{u} + \text{round}(\mathbf{e})$ where $\mathbf{e} \sim \mathcal{N}(0, 1)$

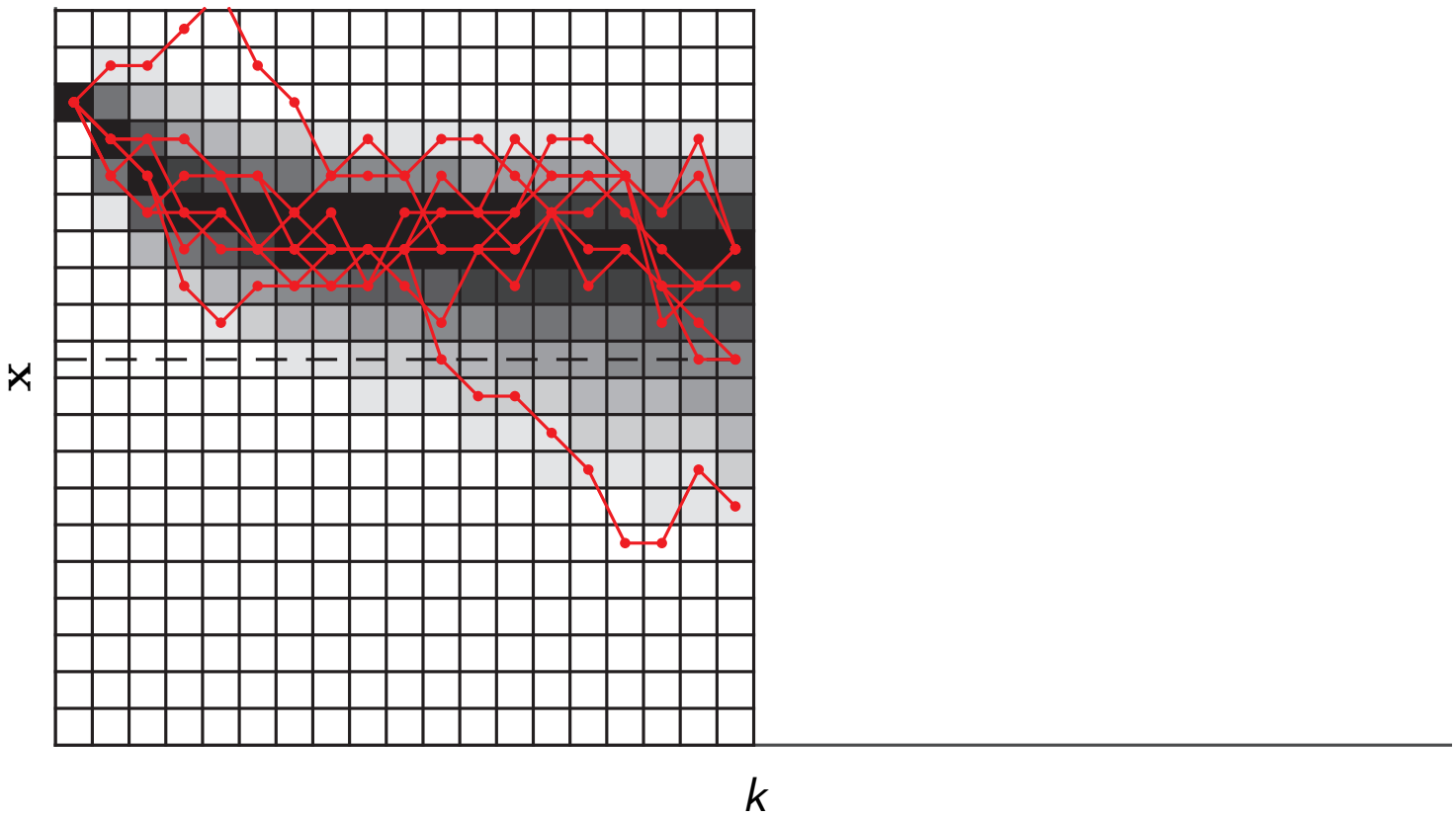


e.g. for
$\mathbf{u} = \text{round}\left(-\frac{\mathbf{x}}{10}\right)$

Let's use the following example (for illustration purposes)

- State $\mathbf{x} \in \mathbb{N}$ and input $\mathbf{u} \in \mathbb{N}$ locked on a grid
- Input $\mathbf{u}$ can only move $\mathbf{x}$ on the grid
- Stage cost: $L(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\mathbf{x}^2 + \frac{1}{2}\mathbf{u}^2$
- Discount $\gamma = 0.9$

Dynamics: $\mathbf{x}_+ = \mathbf{x} + \mathbf{u} + \text{round}(\mathbf{e})$ where $\mathbf{e} \sim \mathcal{N}(0, 1)$

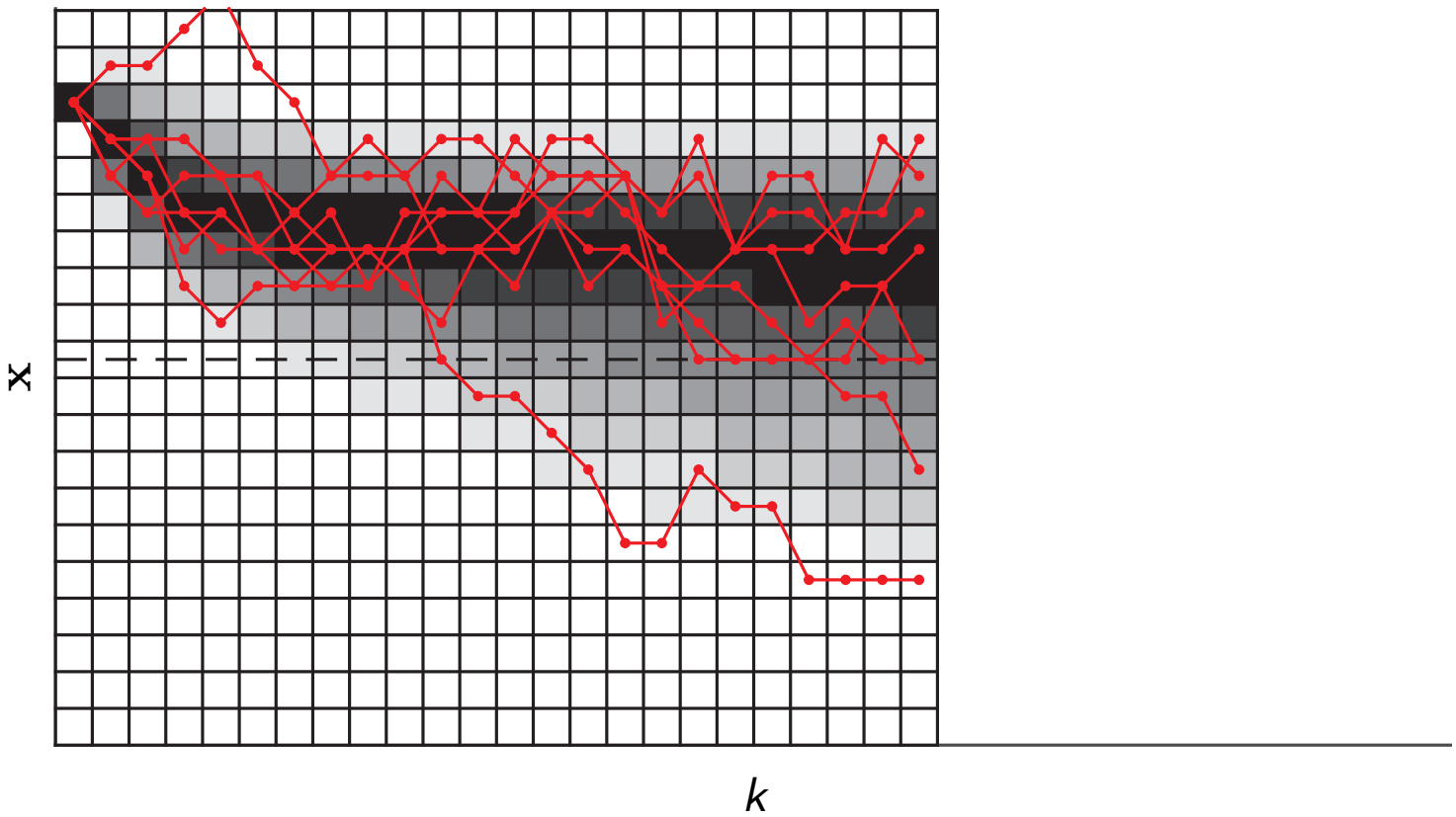


e.g. for
$\mathbf{u} = \text{round}\left(-\frac{\mathbf{x}}{10}\right)$

Let's use the following example (for illustration purposes)

- State $\mathbf{x} \in \mathbb{N}$ and input $\mathbf{u} \in \mathbb{N}$ locked on a grid
- Input $\mathbf{u}$ can only move $\mathbf{x}$ on the grid
- Stage cost: $L(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\mathbf{x}^2 + \frac{1}{2}\mathbf{u}^2$
- Discount $\gamma = 0.9$

Dynamics: $\mathbf{x}_+ = \mathbf{x} + \mathbf{u} + \mathrm{round}(\mathbf{e})$ where $\mathbf{e} \sim \mathcal{N}(0, 1)$

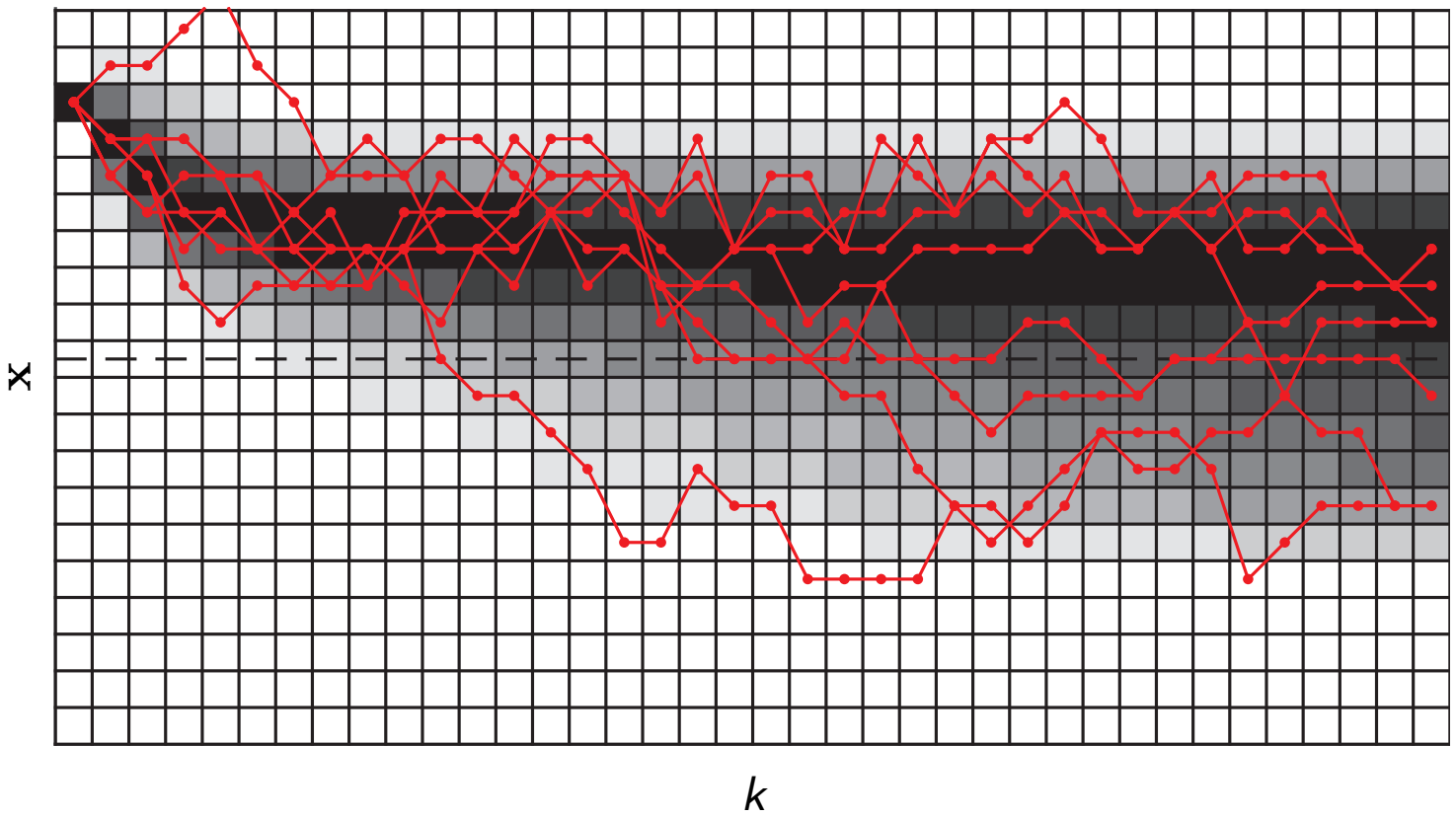


e.g. for
$\mathbf{u} = \mathrm{round}\left(-\frac{\mathbf{x}}{10}\right)$

Let's use the following example (for illustration purposes)

- State $\mathbf{x} \in \mathbb{N}$ and input $\mathbf{u} \in \mathbb{N}$ locked on a grid
- Input $\mathbf{u}$ can only move $\mathbf{x}$ on the grid
- Stage cost: $L(\mathbf{x}, \mathbf{u}) = \frac{1}{2}\mathbf{x}^2 + \frac{1}{2}\mathbf{u}^2$
- Discount $\gamma = 0.9$

Dynamics: $\mathbf{x}_+ = \mathbf{x} + \mathbf{u} + \text{round}(\mathbf{e})$ where $\mathbf{e} \sim \mathcal{N}(0, 1)$

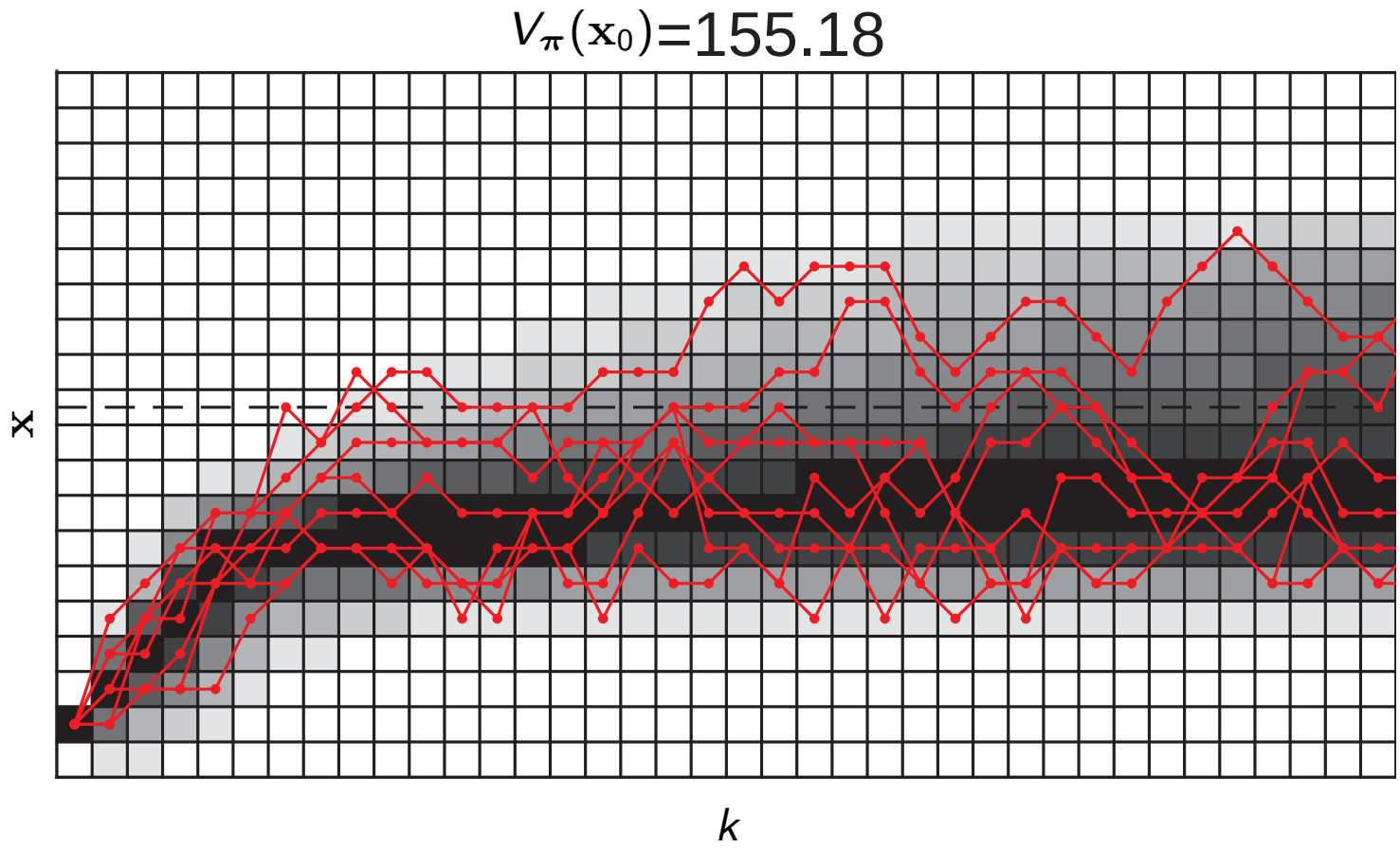


e.g. for
$\mathbf{u} = \text{round}\left(-\frac{\mathbf{x}}{10}\right)$

Value function - How good is it to be in a given state?

Value function for policy π:

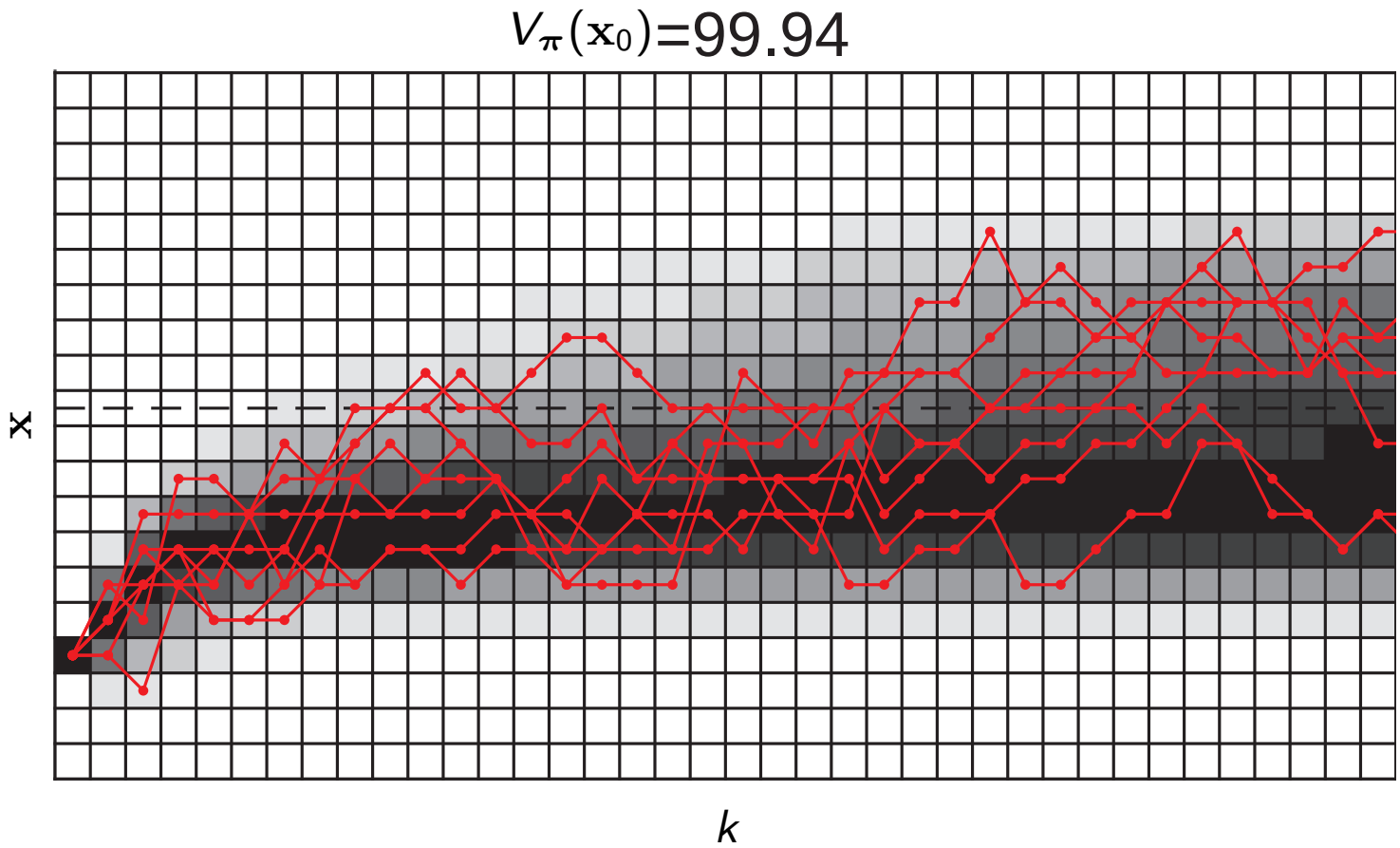
$$V_{\pi}(\mathbf{x}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \mathbf{u}_k = \boldsymbol{\pi}(\mathbf{x}_k),\ \mathbf{x}_0 = \mathbf{x}\right]$$



Value function - How good is it to be in a given state?

Value function for policy π:

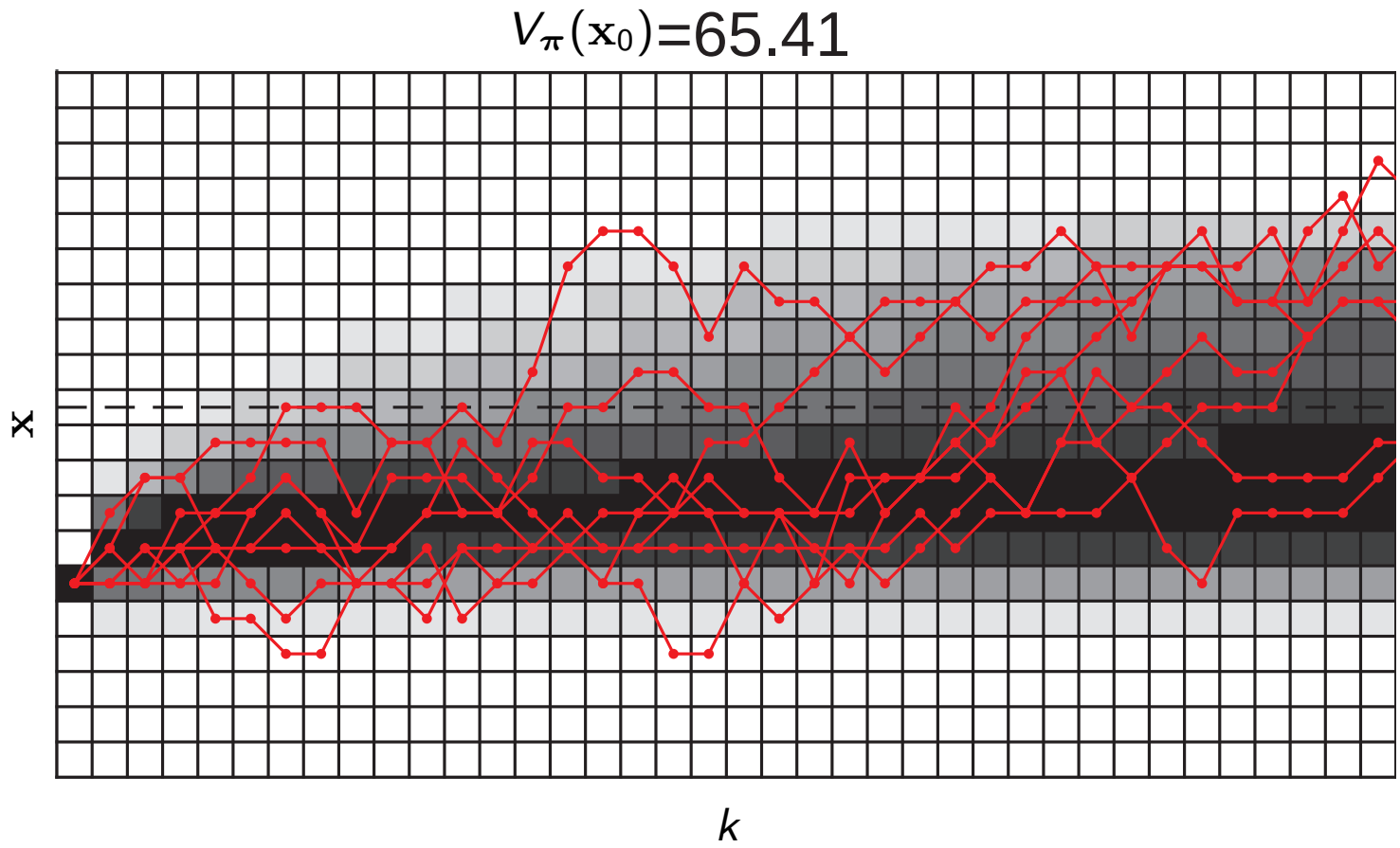
$$V_{\pi}(\mathbf{x}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \mathbf{u}_k = \boldsymbol{\pi}(\mathbf{x}_k),\ \mathbf{x}_0 = \mathbf{x}\right]$$



Value function - How good is it to be in a given state?

Value function for policy π:

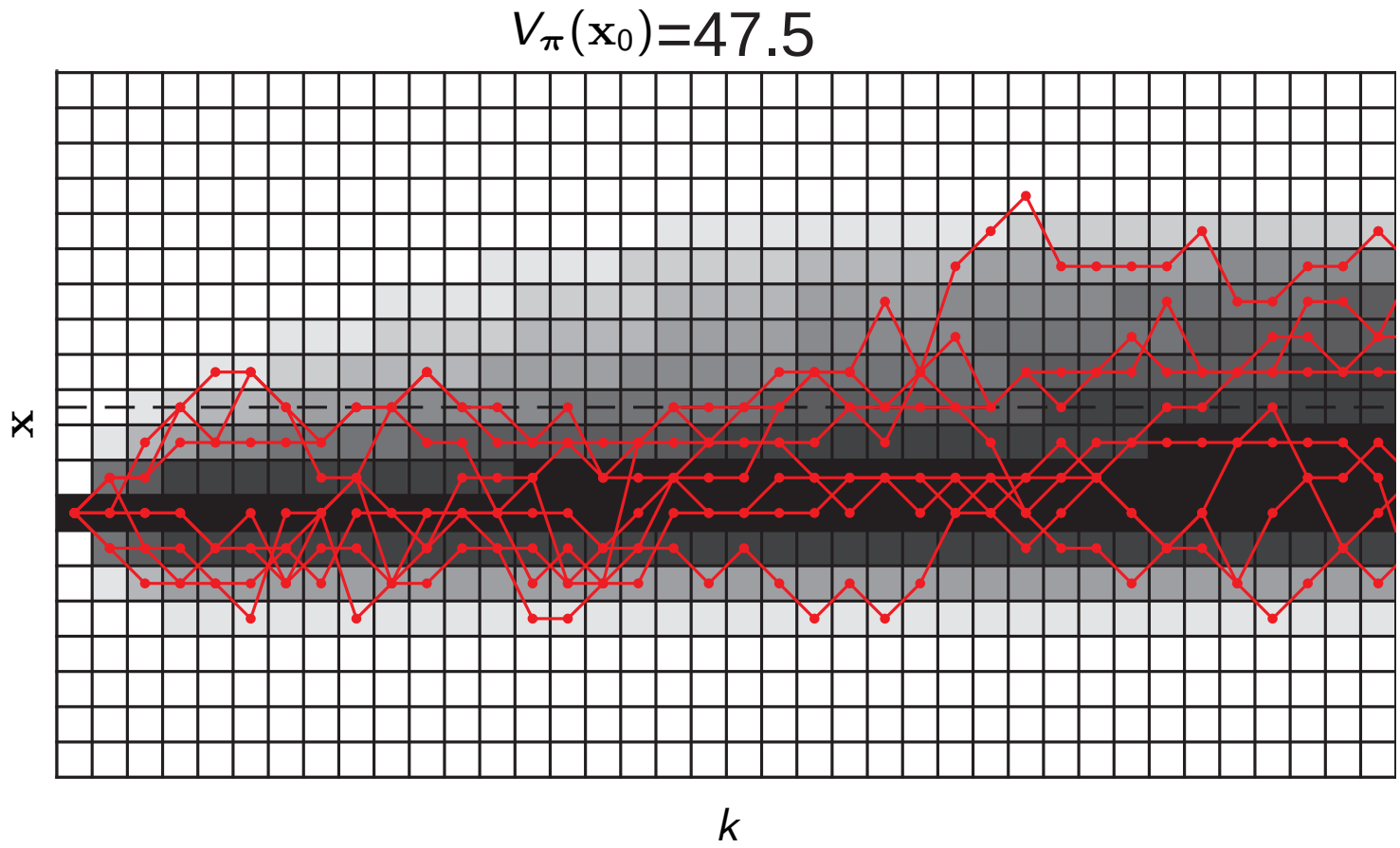
$$V_{\pi}(\mathbf{x}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \mathbf{u}_k = \boldsymbol{\pi}(\mathbf{x}_k),\ \mathbf{x}_0 = \mathbf{x}\right]$$



Value function - How good is it to be in a given state?

Value function for policy π:

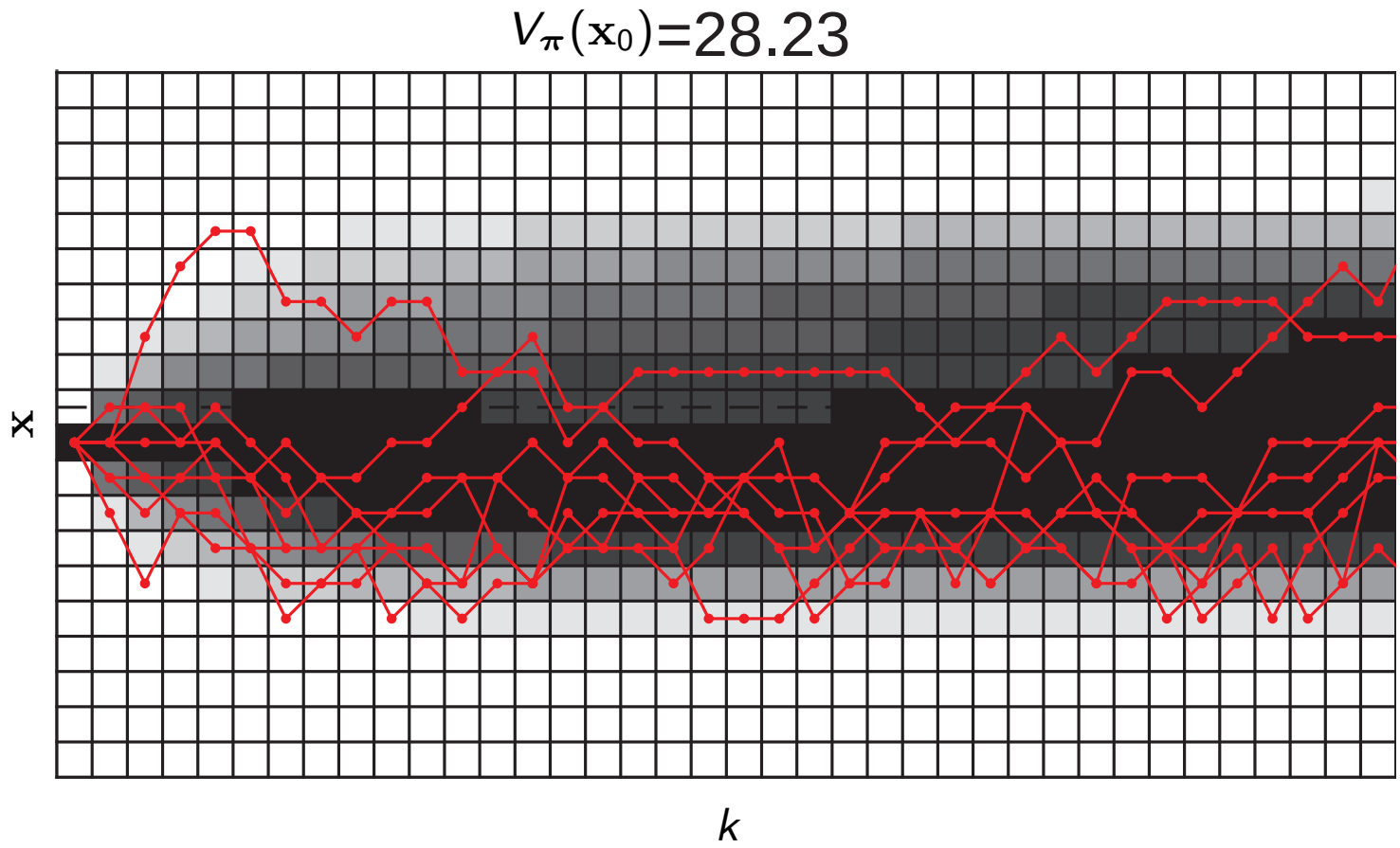
$$V_{\pi}(\mathbf{x}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \mathbf{u}_k = \boldsymbol{\pi}(\mathbf{x}_k),\ \mathbf{x}_0 = \mathbf{x}\right]$$



Value function - How good is it to be in a given state?

Value function for policy π:

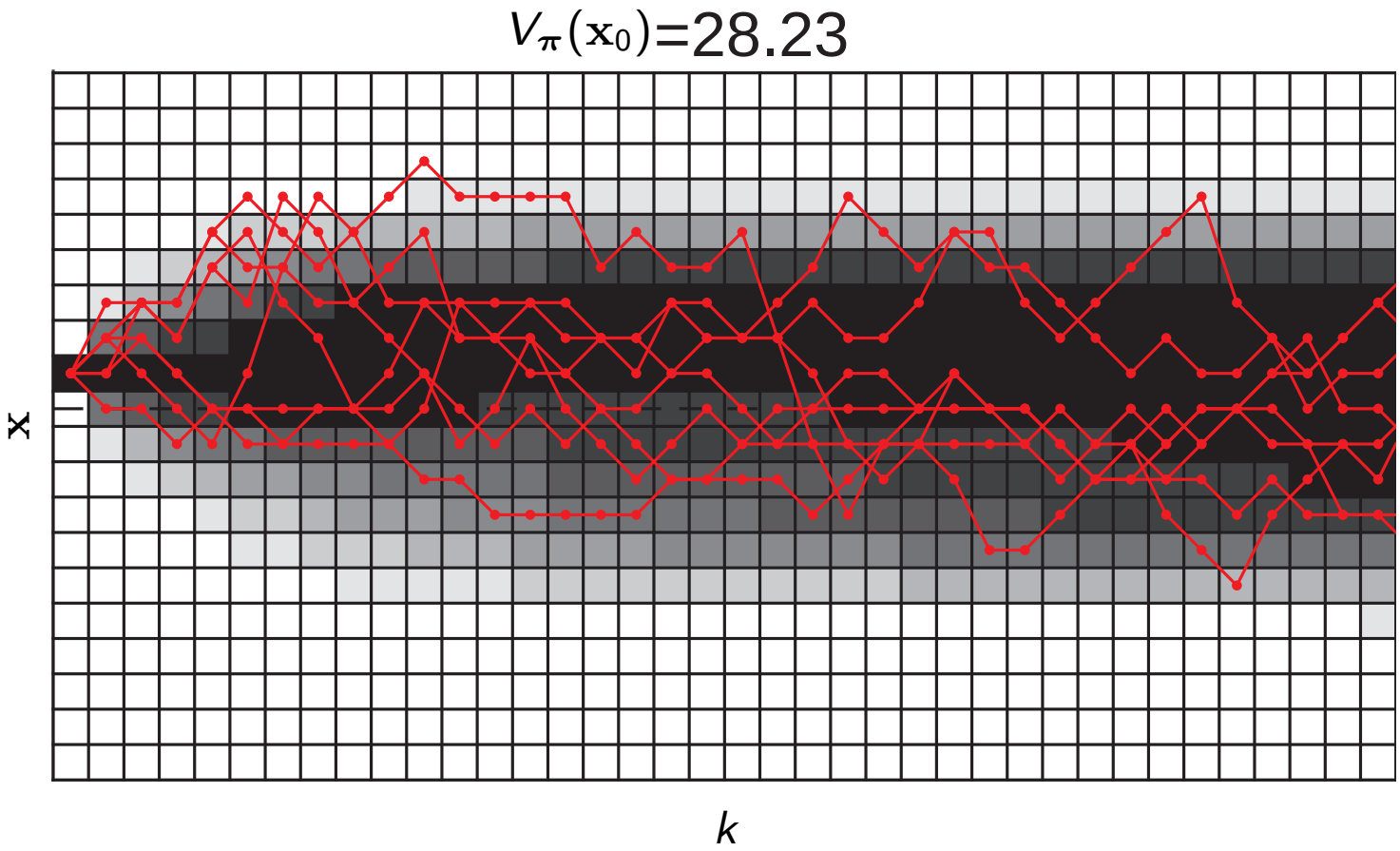
$$V_{\pi}(\mathbf{x}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \mathbf{u}_k = \boldsymbol{\pi}(\mathbf{x}_k),\ \mathbf{x}_0 = \mathbf{x}\right]$$



Value function - How good is it to be in a given state?

Value function for policy π:

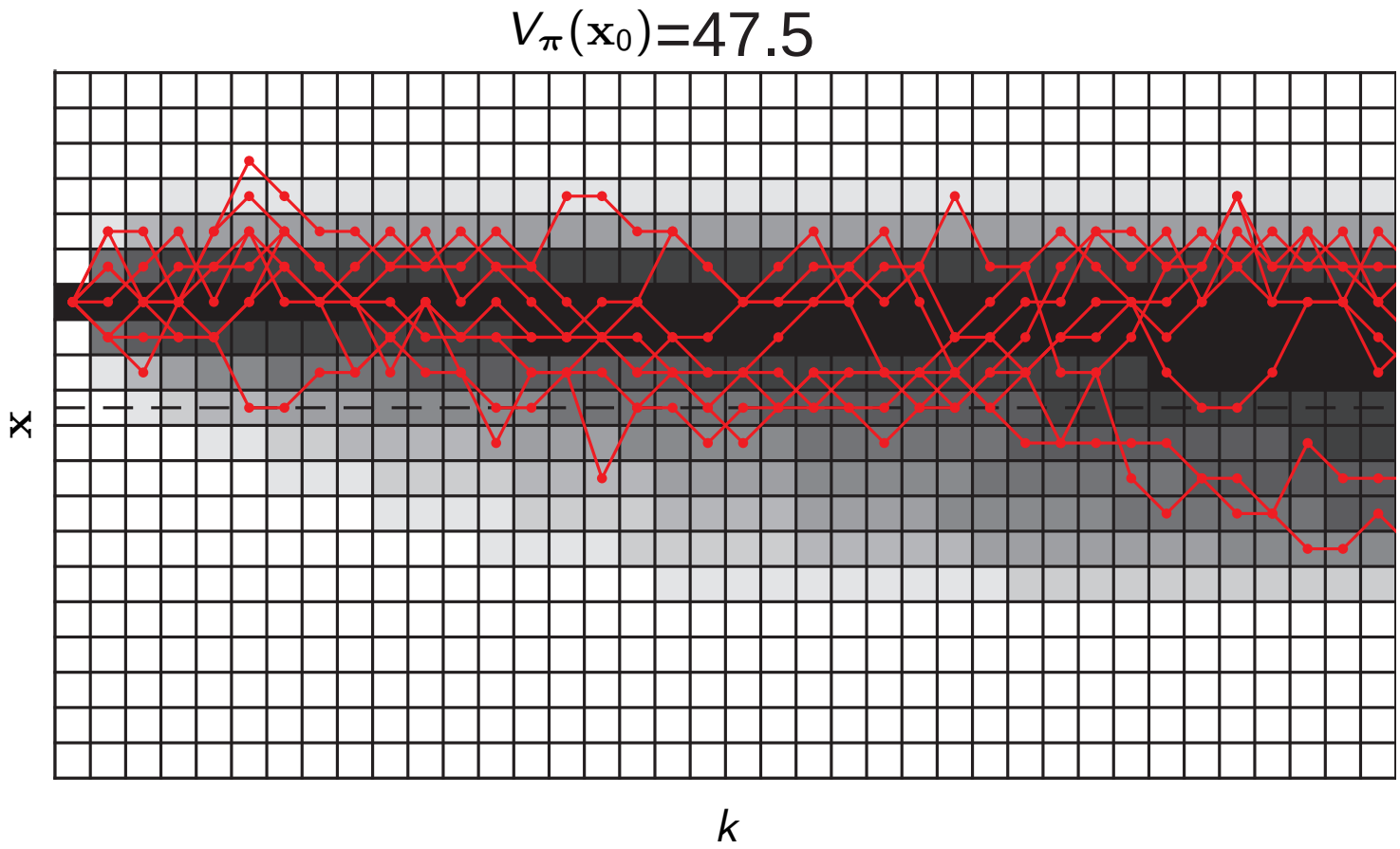
$$V_{\pi}(\mathbf{x}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \mathbf{u}_k = \boldsymbol{\pi}(\mathbf{x}_k),\ \mathbf{x}_0 = \mathbf{x}\right]$$



Value function - How good is it to be in a given state?

Value function for policy π:

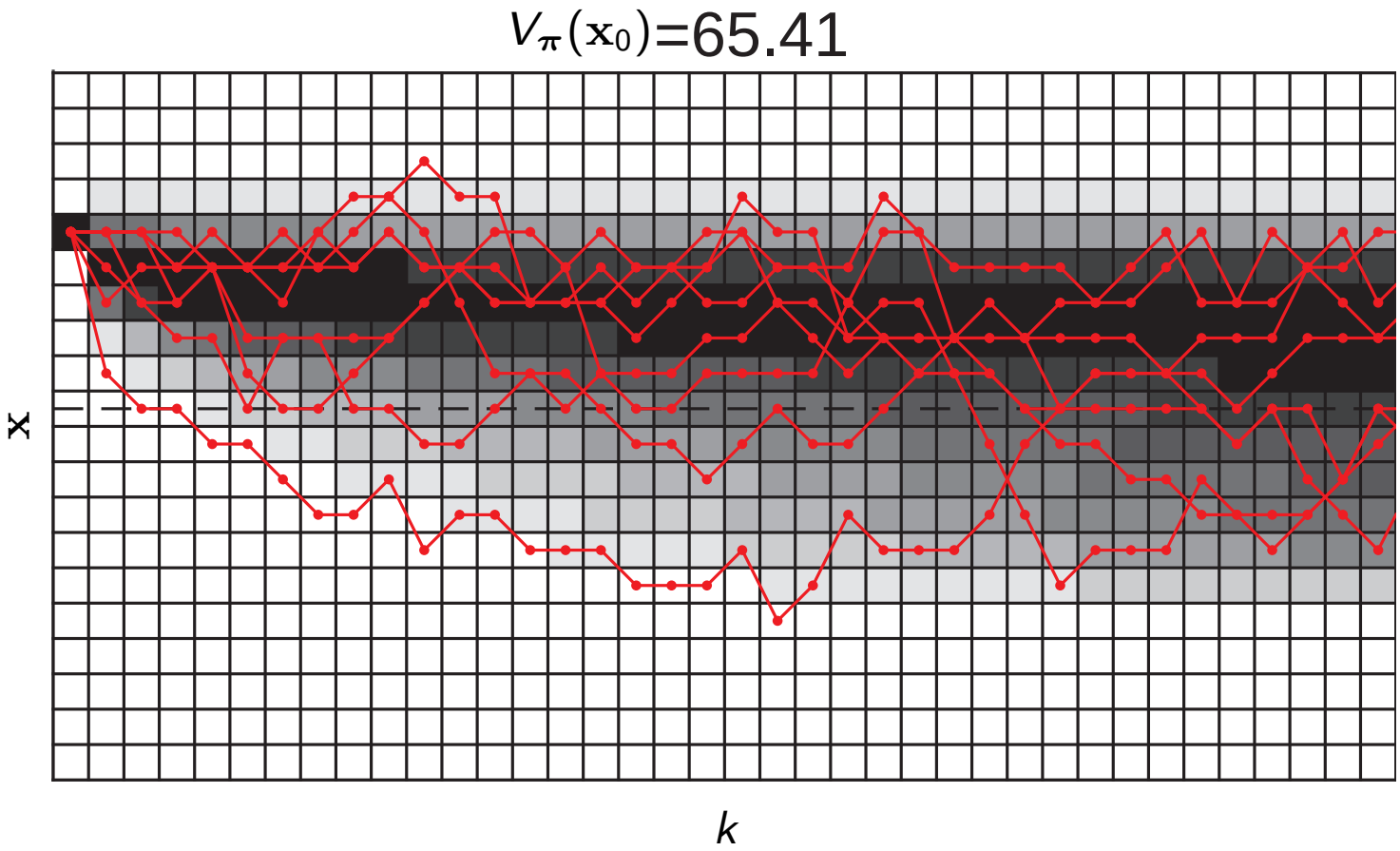
$$V_{\pi}(\mathbf{x}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \mathbf{u}_k = \boldsymbol{\pi}(\mathbf{x}_k),\ \mathbf{x}_0 = \mathbf{x}\right]$$



Value function - How good is it to be in a given state?

Value function for policy π:

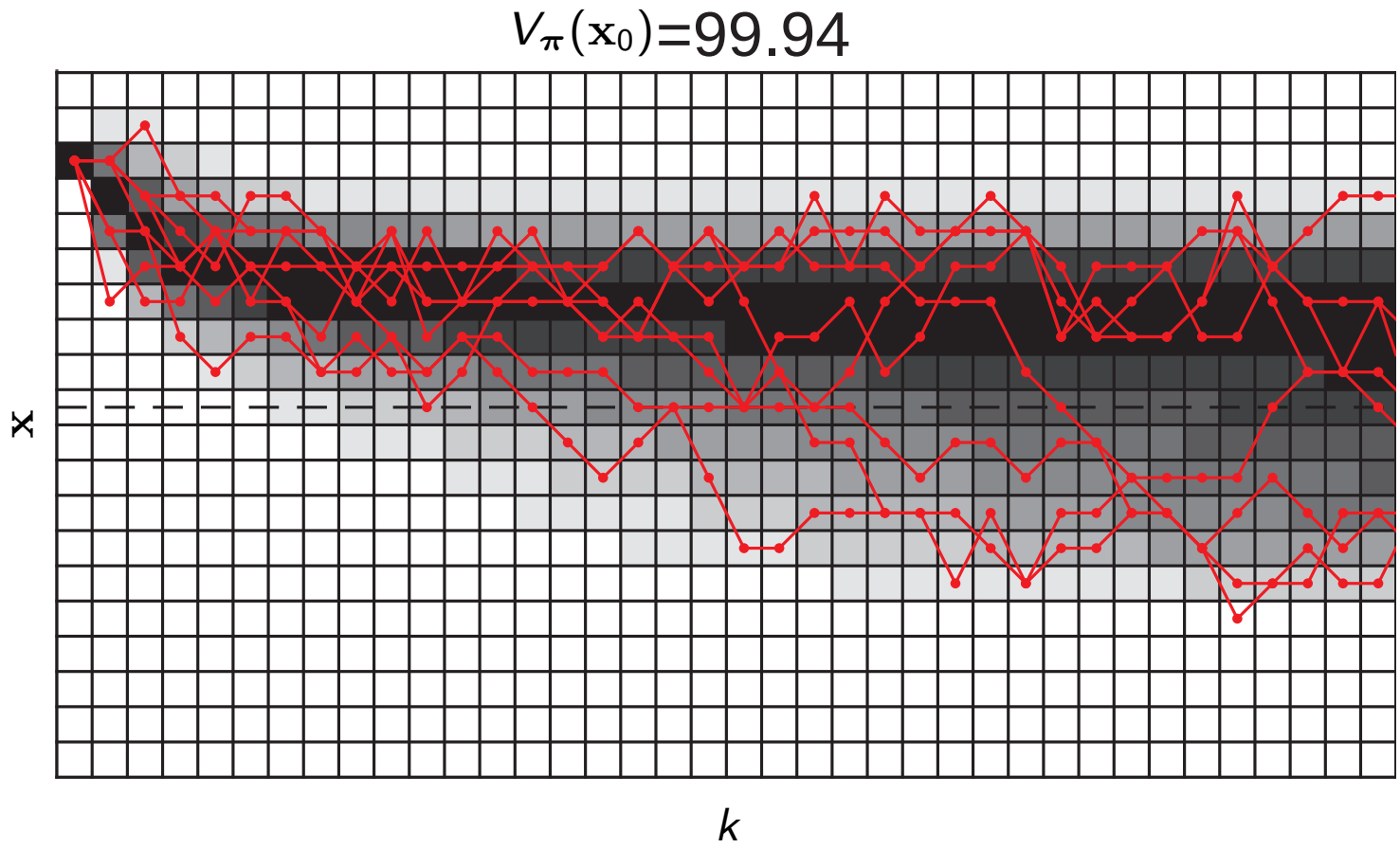
$$V_{\pi}(\mathbf{x}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \mathbf{u}_k = \boldsymbol{\pi}(\mathbf{x}_k),\ \mathbf{x}_0 = \mathbf{x}\right]$$



Value function - How good is it to be in a given state?

Value function for policy π:

$$V_{\pi}(\mathbf{x}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \mathbf{u}_k = \boldsymbol{\pi}(\mathbf{x}_k),\ \mathbf{x}_0 = \mathbf{x} \right]$$



Value function - How good is it to be in a given state?

Value function for policy π:

$$V_{\pi}(\mathbf{x}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \mathbf{u}_k = \boldsymbol{\pi}(\mathbf{x}_k),\ \mathbf{x}_0 = \mathbf{x} \right]$$

Meaning: for a given state $\mathbf{x}$

- We follow policy $\boldsymbol{\pi}$ for all k (can be stochastic)
- State trajectory $\mathbf{x}_{1,\ldots,\infty}$ is stochastic as it follows $\mathbb{P}[\mathbf{x}_+ \,|\, \mathbf{x}, \mathbf{u}]$
- Initial state $\mathbf{x}_0 = \mathbf{x}$ is fixed
- Value function at $\mathbf{x}$ is the resulting cumulated (discounted) cost
- Expected value $\mathbb{E}[.]$ taken over trajectories, using trajectories distributions...

Value function - How good is it to be in a given state?

Value function for policy π:

$$V_{\pi}(\mathbf{x}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \mathbf{u}_k = \boldsymbol{\pi}(\mathbf{x}_k),\, \mathbf{x}_0 = \mathbf{x} \right]$$

Formally:

$$V_{\pi}(\mathbf{x}) = \sum_{k=0}^{\infty} \int \gamma^k L(\mathbf{x}_k, \boldsymbol{\pi}(\mathbf{x}_k)) \mathbb{P}\left[\mathbf{x}_k \,|\, \mathbf{x}_0 = \mathbf{x}\right] \mathrm{d}\mathbf{x}_k$$

where we use the distributions $\mathbf{x}_0 \to \mathbf{x}_k$, recall

$$\mathbb{P}\left[\mathbf{x}_k \,|\, \mathbf{x}_0\right] = \int \prod_{i=1}^{k} \mathbb{P}\left[\mathbf{x}_i \,|\, \mathbf{x}_{i-1},\, \boldsymbol{\pi}(\mathbf{x}_{i-1})\right] \cdot \mathrm{d}\mathbf{x}_1 \ldots \mathrm{d}\mathbf{x}_{k-1}$$

Value function - How good is it to be in a given state?

Value function for policy π:

$$V_{\pi}(\mathbf{x}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \mathbf{u}_k = \boldsymbol{\pi}(\mathbf{x}_k),\ \mathbf{x}_0 = \mathbf{x}\right]$$

Formally:

$$V_{\pi}(\mathbf{x}) = \sum_{k=0}^{\infty} \int \gamma^k L(\mathbf{x}_k, \boldsymbol{\pi}(\mathbf{x}_k)) \mathbb{P}\left[\mathbf{x}_k \,|\, \mathbf{x}_0 = \mathbf{x}\right] \mathrm{d}\mathbf{x}_k$$

where we use the distributions $\mathbf{x}_0 \to \mathbf{x}_k$, recall

$$\mathbb{P}\left[\mathbf{x}_k \,|\, \mathbf{x}_0\right] = \int \prod_{i=1}^{k} \mathbb{P}\left[\mathbf{x}_i \,|\, \mathbf{x}_{i-1},\ \boldsymbol{\pi}(\mathbf{x}_{i-1})\right] \cdot \mathrm{d}\mathbf{x}_1 \ldots \mathrm{d}\mathbf{x}_{k-1}$$

I.e.

$$V_{\pi}(\mathbf{x}_0) = \sum_{k=0}^{\infty} \int \gamma^k L(\mathbf{x}_k, \boldsymbol{\pi}(\mathbf{x}_k)) \prod_{i=1}^{k} \mathbb{P}\left[\mathbf{x}_i \,|\, \mathbf{x}_{i-1},\ \boldsymbol{\pi}(\mathbf{x}_{i-1})\right] \cdot \mathrm{d}\mathbf{x}_1 \ldots \mathrm{d}\mathbf{x}_{k-1}$$

Note that this construction has all the properties of an expected value (e.g. linearity)

Value function - How good is it to be in a given state?

Value function for policy π:

$$V_\pi(\mathbf{x}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \mathbf{u}_k = \boldsymbol{\pi}(\mathbf{x}_k),\ \mathbf{x}_0 = \mathbf{x}\right]$$

Formally:

$$V_\pi(\mathbf{x}) = \sum_{k=0}^{\infty} \int \gamma^k L(\mathbf{x}_k, \boldsymbol{\pi}(\mathbf{x}_k)) \mathbb{P}\left[\mathbf{x}_k \,|\, \mathbf{x}_0 = \mathbf{x}\right] \mathrm{d}\mathbf{x}_k$$

where we use the distributions $\mathbf{x}_0 \to \mathbf{x}_k$, recall

$$\mathbb{P}\left[\mathbf{x}_k \,|\, \mathbf{x}_0\right] = \int \prod_{i=1}^{k} \mathbb{P}\left[\mathbf{x}_i \,|\, \mathbf{x}_{i-1},\ \boldsymbol{\pi}(\mathbf{x}_{i-1})\right] \cdot \mathrm{d}\mathbf{x}_1 \ldots \mathrm{d}\mathbf{x}_{k-1}$$

I.e.

$$V_\pi(\mathbf{x}_0) = \sum_{k=0}^{\infty} \int \gamma^k L(\mathbf{x}_k, \boldsymbol{\pi}(\mathbf{x}_k)) \prod_{i=1}^{k} \mathbb{P}\left[\mathbf{x}_i \,|\, \mathbf{x}_{i-1},\ \boldsymbol{\pi}(\mathbf{x}_{i-1})\right] \cdot \mathrm{d}\mathbf{x}_1 \ldots \mathrm{d}\mathbf{x}_{k-1}$$

Relation $J(\boldsymbol{\pi})$ to $V_\pi(\mathbf{x})$?

$$J(\boldsymbol{\pi}) = \mathbb{E}[V_\pi(\mathbf{x})] = \int V_\pi(\mathbf{x}_0) \mathbb{P}[\mathbf{x}_0] \mathrm{d}\mathbf{x}_0$$

Note that this construction has all the properties of an expected value (e.g. linearity)

Bellman equation - Computing the value function

Value function for policy π (either $\mathbf{u} = \boldsymbol{\pi}(\mathbf{x})$ or $\boldsymbol{\pi}[\mathbf{u} \,|\, \mathbf{x}]$):

$$V_{\boldsymbol{\pi}}(\mathbf{x}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \boldsymbol{\pi},\, \mathbf{x}_0 = \mathbf{x} \right]$$

Bellman equation - Computing the value function

Value function for policy π (either $\mathbf{u} = \boldsymbol{\pi}(\mathbf{x})$ or $\boldsymbol{\pi}[\mathbf{u} \,|\, \mathbf{x}]$):

$$V_{\boldsymbol{\pi}}(\mathbf{x}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \boldsymbol{\pi},\, \mathbf{x}_0 = \mathbf{x} \right]$$

Value function backup

$$V_{\boldsymbol{\pi}}(\mathbf{x}_0) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \mathbf{x}_0, \boldsymbol{\pi} \right]$$

Bellman equation - Computing the value function

Value function for policy π (either $\mathbf{u} = \pi(\mathbf{x})$ or $\pi[\mathbf{u} \mid \mathbf{x}]$):

$$V_\pi(\mathbf{x}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \pi,\, \mathbf{x}_0 = \mathbf{x}\right]$$

Value function backup

$$V_\pi(\mathbf{x}_0) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \mathbf{x}_0, \pi\right] = L(\mathbf{x}_0, \pi(\mathbf{x}_0)) + \mathbb{E}\left[\sum_{k=1}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \mathbf{x}_0, \pi\right]$$

Bellman equation - Computing the value function

Value function for policy π (either $\mathbf{u} = \pi(\mathbf{x})$ or $\pi[\mathbf{u} \,|\, \mathbf{x}]$):

$$V_\pi(\mathbf{x}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \pi,\, \mathbf{x}_0 = \mathbf{x} \right]$$

Value function backup

$$V_\pi(\mathbf{x}_0) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \mathbf{x}_0, \pi \right] = L(\mathbf{x}_0, \pi(\mathbf{x}_0)) + \mathbb{E}\left[\sum_{k=1}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \mathbf{x}_0, \pi \right]$$

$$= L(\mathbf{x}_0, \pi(\mathbf{x}_0)) + \gamma \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_{k+1}, \mathbf{u}_{k+1}) \,\middle|\, \mathbf{x}_0, \pi \right]$$

Bellman equation - Computing the value function

Value function for policy π (either $\mathbf{u} = \pi(\mathbf{x})$ or $\pi[\mathbf{u} \,|\, \mathbf{x}]$):

$$V_\pi(\mathbf{x}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \pi,\, \mathbf{x}_0 = \mathbf{x} \right]$$

Value function backup

$$\begin{aligned} V_\pi(\mathbf{x}_0) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \mathbf{x}_0, \pi \right] &= L(\mathbf{x}_0, \pi(\mathbf{x}_0)) + \mathbb{E}\left[\sum_{k=1}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \mathbf{x}_0, \pi \right] \\ &= L(\mathbf{x}_0, \pi(\mathbf{x}_0)) + \gamma \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_{k+1}, \mathbf{u}_{k+1}) \,\middle|\, \mathbf{x}_0, \pi \right] \\ &= L(\mathbf{x}_0, \pi(\mathbf{x}_0)) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_1) \,|\, \mathbf{x}_0, \pi \right] \end{aligned}$$

Bellman equation - Computing the value function

Value function for policy π (either $\mathbf{u} = \pi(\mathbf{x})$ or $\pi[\mathbf{u} \,|\, \mathbf{x}]$):

$$V_\pi(\mathbf{x}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \pi,\, \mathbf{x}_0 = \mathbf{x} \right]$$

Value function backup

$$\begin{aligned} V_\pi(\mathbf{x}_0) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \mathbf{x}_0, \pi \right] &= L(\mathbf{x}_0, \pi(\mathbf{x}_0)) + \mathbb{E}\left[\sum_{k=1}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \mathbf{x}_0, \pi \right] \\ &= L(\mathbf{x}_0, \pi(\mathbf{x}_0)) + \gamma \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_{k+1}, \mathbf{u}_{k+1}) \,\middle|\, \mathbf{x}_0, \pi \right] \\ &= L(\mathbf{x}_0, \pi(\mathbf{x}_0)) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_1) \,|\, \mathbf{x}_0, \pi \right] \end{aligned}$$

where

$$\mathbb{E}\left[V_\pi(\mathbf{x}_1) \,|\, \mathbf{x}_0 \right] = \int V_\pi(\mathbf{x}_1) \overbrace{\mathbb{P}\left[\mathbf{x}_1 \,|\, \mathbf{x}_0, \pi(\mathbf{x}_0) \right]}^{\textit{Model}} \mathrm{d}\mathbf{x}_1$$

Bellman equation - Computing the value function

Value function for policy π (either $\mathbf{u} = \boldsymbol{\pi}(\mathbf{x})$ or $\boldsymbol{\pi}[\mathbf{u} \,|\, \mathbf{x}]$):

$$V_{\boldsymbol{\pi}}(\mathbf{x}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{u}_k) \,\middle|\, \boldsymbol{\pi},\, \mathbf{x}_0 = \mathbf{x} \right]$$

We get:

Bellman equation

$$V_{\boldsymbol{\pi}}(\mathbf{x}) = L(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) + \gamma \mathbb{E}\left[V_{\boldsymbol{\pi}}(\mathbf{x}_+) \,|\, \mathbf{x}, \boldsymbol{\pi} \right]$$

sometimes called optimality backup

Action-value function

Bellman equation for the value function

$$V_{\pi}(\mathbf{x}) = L(\mathbf{x}, \pi(\mathbf{x})) + \gamma \mathbb{E}\left[V_{\pi}(\mathbf{x}_+) \mid \mathbf{x}, \pi\right]$$

Define

Action-value function:

$$Q_{\pi}(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V_{\pi}(\mathbf{x}_+) \mid \mathbf{x}, \mathbf{u}\right]$$

Action-value function

Bellman equation for the value function

$$V_\pi(\mathbf{x}) = L(\mathbf{x}, \pi(\mathbf{x})) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_+) \mid \mathbf{x}, \pi\right]$$

Define

Action-value function:

$$Q_\pi(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_+) \mid \mathbf{x}, \mathbf{u}\right]$$

Interpretation:

- From state $\mathbf{x}$, use given input $\mathbf{u}$, yields (stochastic) transition $\mathbf{x}, \mathbf{u} \to \mathbf{x}_+$
- Then from $\mathbf{x}_+$ follow policy π
- Action-value function $Q_\pi(\mathbf{x}, \mathbf{u})$ is the expected return of the resulting trajectory

Action-value function

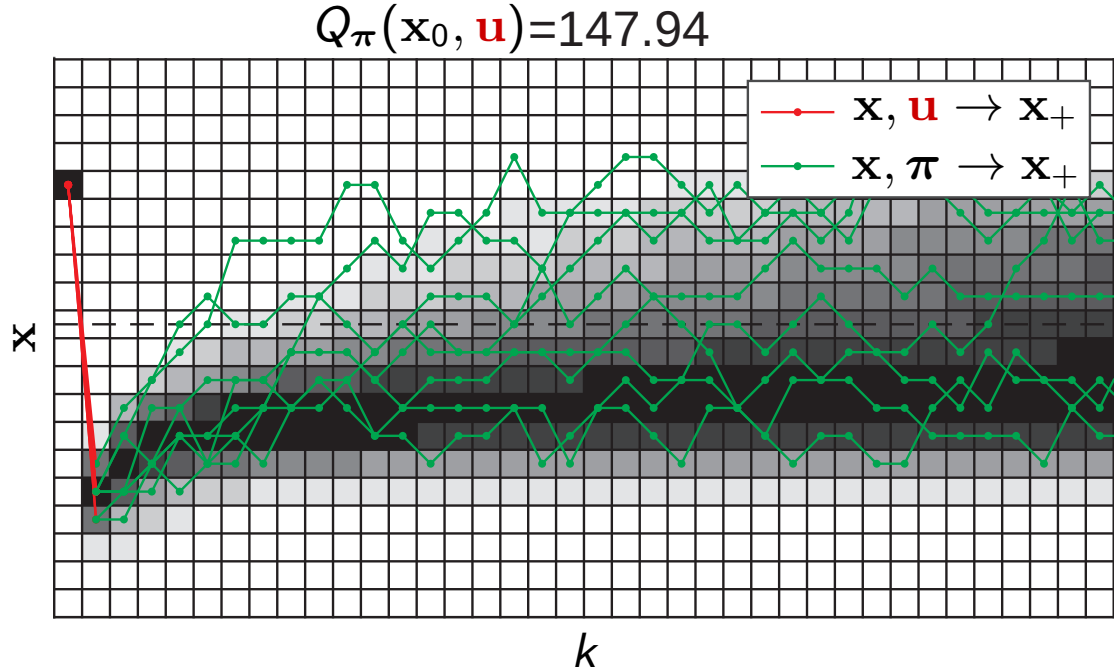
Bellman equation for the value function

$$V_\pi(\mathbf{x}) = L(\mathbf{x}, \pi(\mathbf{x})) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_+) \mid \mathbf{x}, \pi\right]$$

Define

Action-value function:

$$Q_\pi(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_+) \mid \mathbf{x}, \mathbf{u}\right]$$



Action-value function

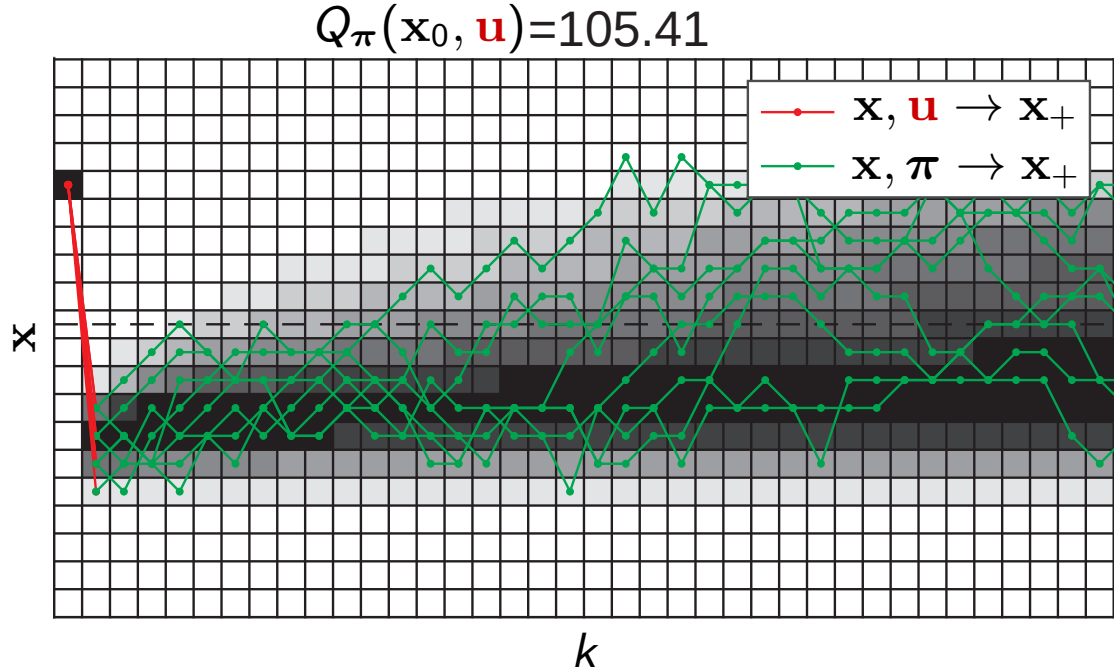
Bellman equation for the value function

$$V_\pi(\mathbf{x}) = L(\mathbf{x}, \pi(\mathbf{x})) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_+) \mid \mathbf{x}, \pi\right]$$

Define

Action-value function:

$$Q_\pi(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_+) \mid \mathbf{x}, \mathbf{u}\right]$$



Action-value function

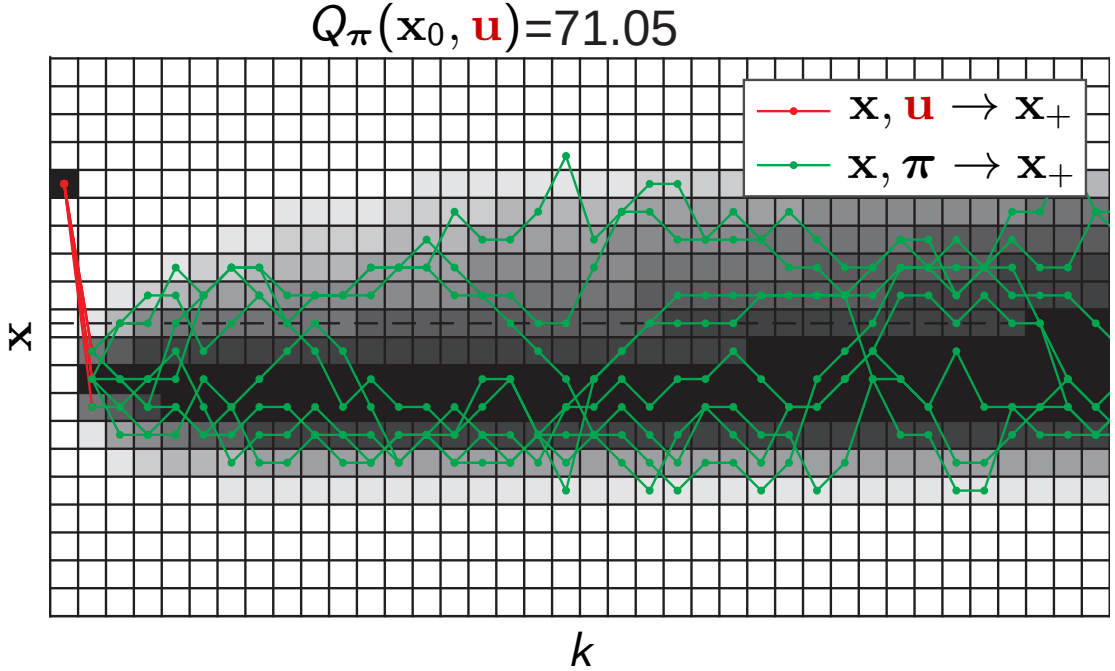
Bellman equation for the value function

$$V_\pi(\mathbf{x}) = L(\mathbf{x}, \pi(\mathbf{x})) + \gamma \mathbb{E}[V_\pi(\mathbf{x}_+) \mid \mathbf{x}, \pi]$$

Define

Action-value function:

$$Q_\pi(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}[V_\pi(\mathbf{x}_+) \mid \mathbf{x}, \mathbf{u}]$$



Action-value function

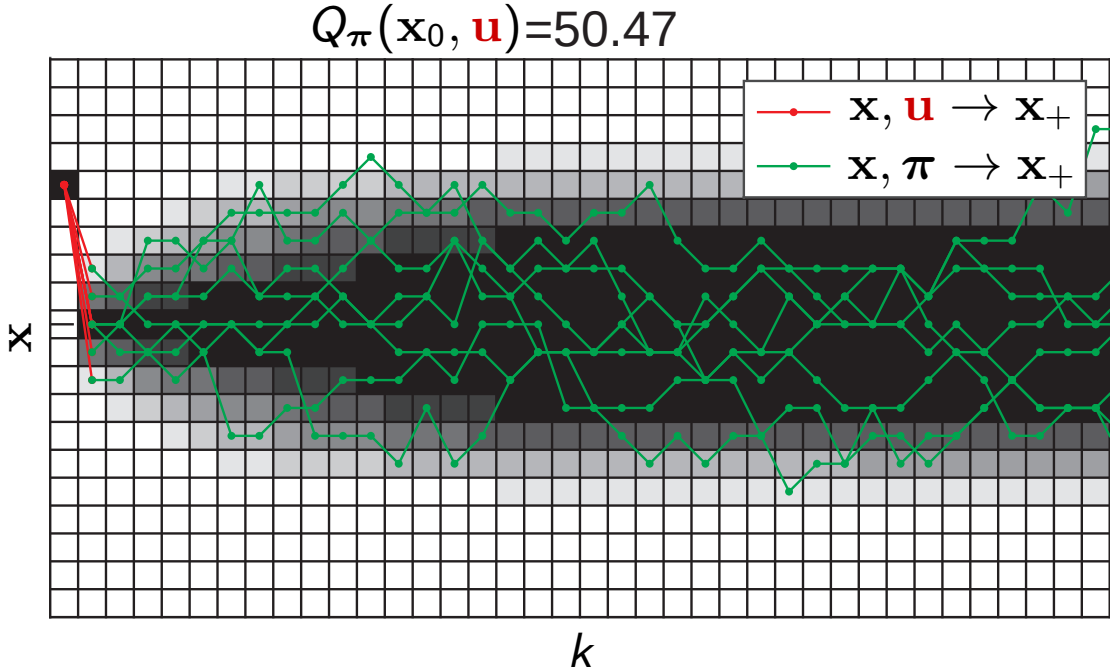
Bellman equation for the value function

$$V_\pi(\mathbf{x}) = L(\mathbf{x}, \pi(\mathbf{x})) + \gamma \mathbb{E}[V_\pi(\mathbf{x}_+) \mid \mathbf{x}, \pi]$$

Define

Action-value function:

$$Q_\pi(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}[V_\pi(\mathbf{x}_+) \mid \mathbf{x}, \mathbf{u}]$$



Action-value function

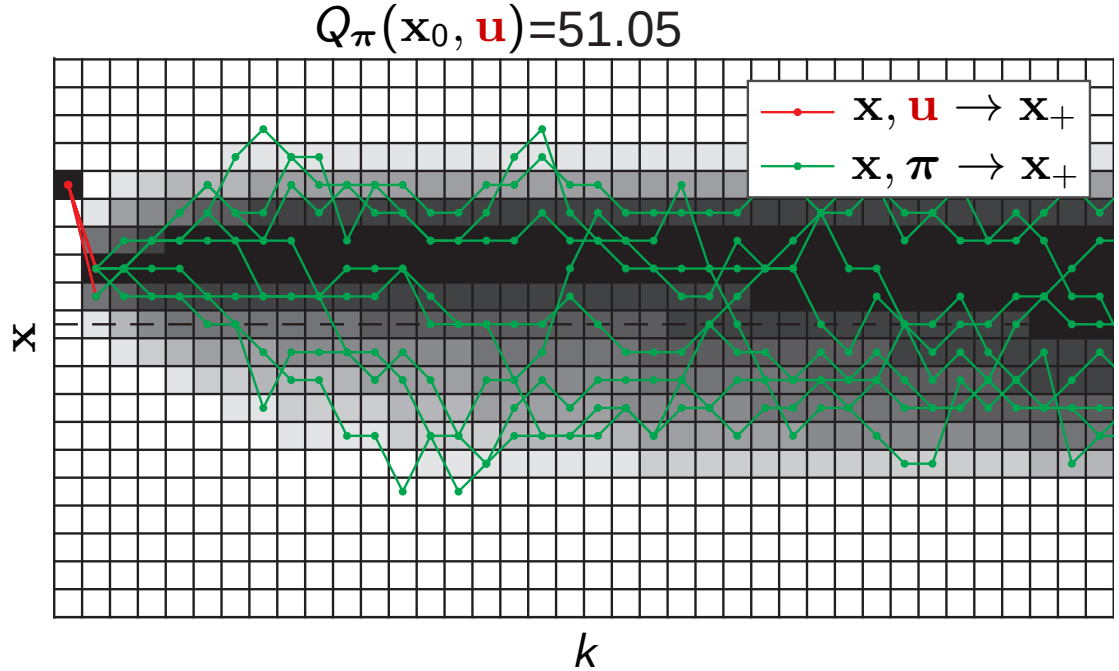
Bellman equation for the value function

$$V_\pi(\mathbf{x}) = L(\mathbf{x}, \pi(\mathbf{x})) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_+) \mid \mathbf{x}, \pi\right]$$

Define

Action-value function:

$$Q_\pi(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_+) \mid \mathbf{x}, \mathbf{u}\right]$$



Action-value function

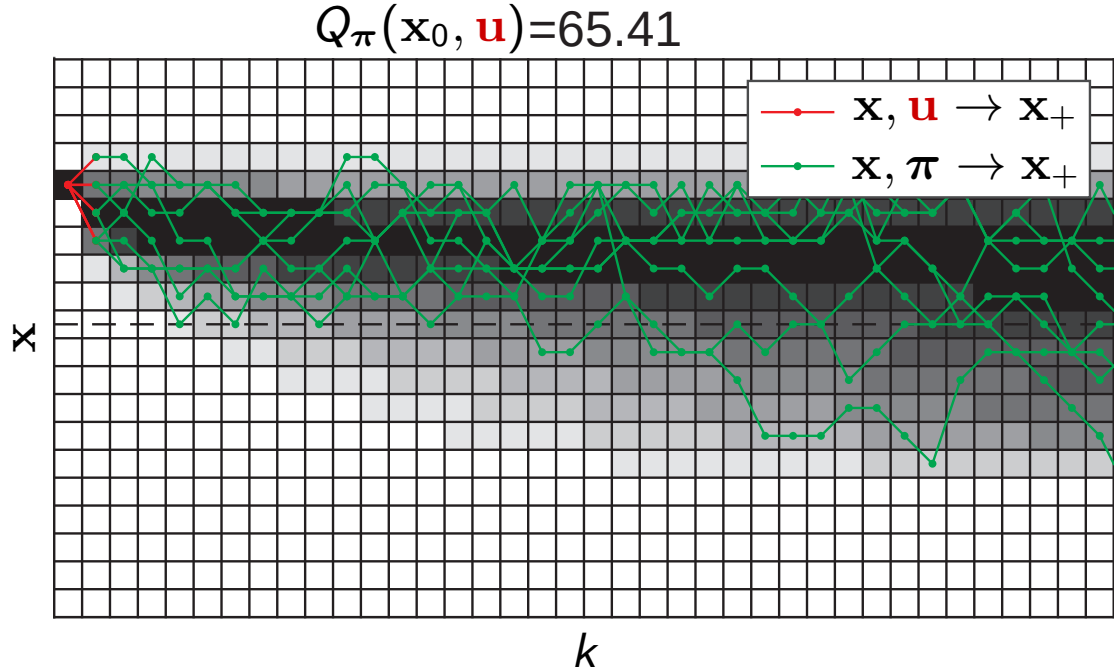
Bellman equation for the value function

$$V_\pi(\mathbf{x}) = L(\mathbf{x}, \pi(\mathbf{x})) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_+) \mid \mathbf{x}, \pi\right]$$

Define

Action-value function:

$$Q_\pi(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_+) \mid \mathbf{x}, \mathbf{u}\right]$$



Action-value function

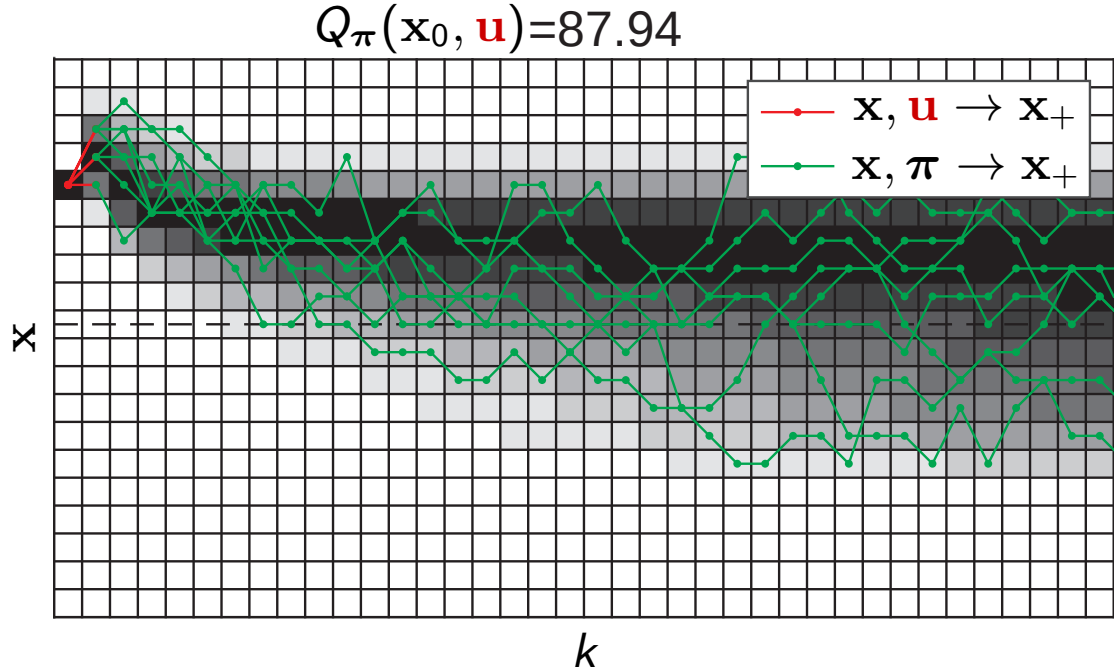
Bellman equation for the value function

$$V_\pi(\mathbf{x}) = L(\mathbf{x}, \pi(\mathbf{x})) + \gamma \mathbb{E}[V_\pi(\mathbf{x}_+) \mid \mathbf{x}, \pi]$$

Define

Action-value function:

$$Q_\pi(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}[V_\pi(\mathbf{x}_+) \mid \mathbf{x}, \mathbf{u}]$$



Action-value function

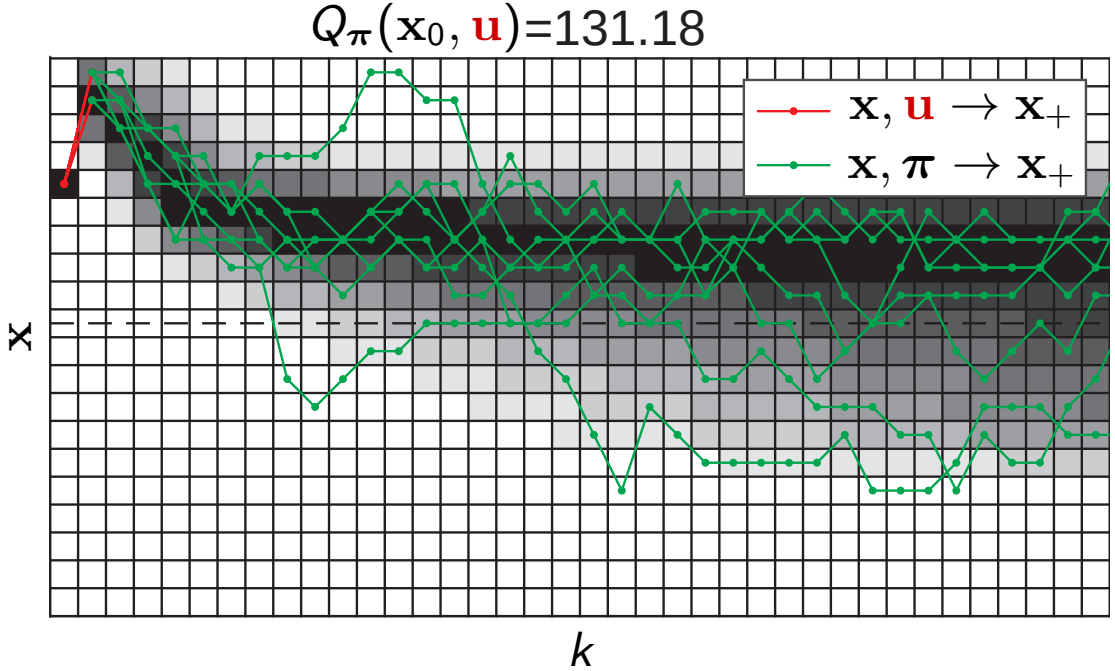
Bellman equation for the value function

$$V_\pi(\mathbf{x}) = L(\mathbf{x}, \pi(\mathbf{x})) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_+) \mid \mathbf{x}, \pi\right]$$

Define

Action-value function:

$$Q_\pi(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_+) \mid \mathbf{x}, \mathbf{u}\right]$$



Action-value function

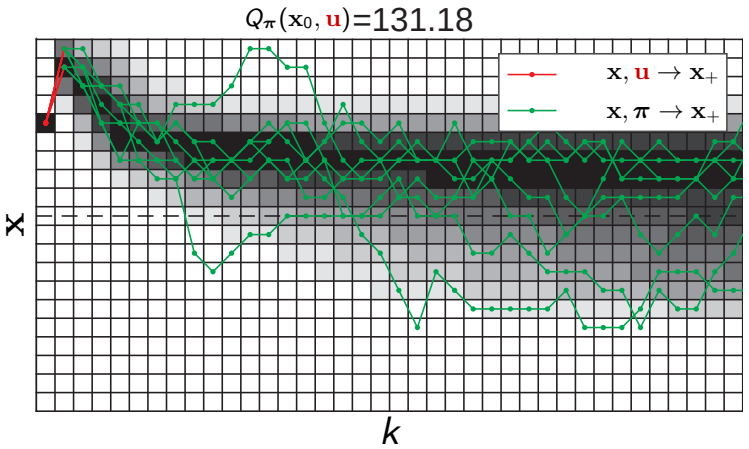
Bellman equation for the value function

$$V_\pi(\mathbf{x}) = L(\mathbf{x}, \pi(\mathbf{x})) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_+) \mid \mathbf{x}, \pi\right]$$

Define

Action-value function:

$$Q_\pi(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_+) \mid \mathbf{x}, \mathbf{u}\right]$$



Interpretation:

- From state $\mathbf{x}$, use given input $\mathbf{u}$, yields (stochastic) transition $\mathbf{x}, \mathbf{u} \to \mathbf{x}_+$
- Then from $\mathbf{x}_+$ onward follow policy π
- Action-value function $Q_\pi(\mathbf{x}, \mathbf{u})$ is the expected return of the resulting trajectory

Action-value function

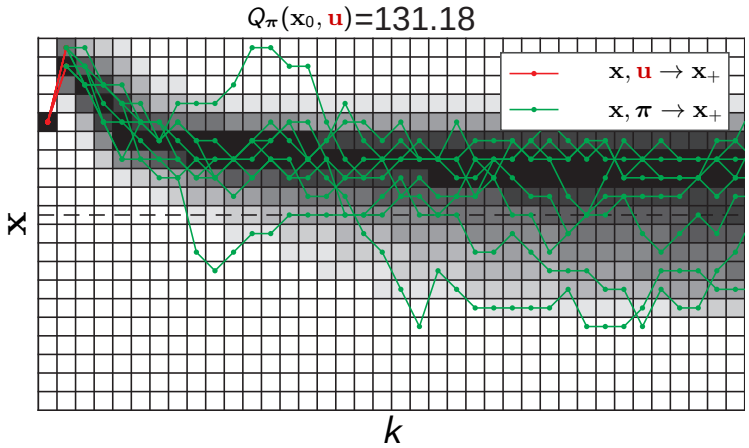
Bellman equation for the value function

$$V_\pi(\mathbf{x}) = L(\mathbf{x}, \pi(\mathbf{x})) + \gamma \mathbb{E}[V_\pi(\mathbf{x}_+) \mid \mathbf{x}, \pi]$$

Define

Action-value function:

$$Q_\pi(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}[V_\pi(\mathbf{x}_+) \mid \mathbf{x}, \mathbf{u}]$$



Interpretation:

- From state $\mathbf{x}$, use given input $\mathbf{u}$, yields (stochastic) transition $\mathbf{x}, \mathbf{u} \to \mathbf{x}_+$
- Then from $\mathbf{x}_+$ onward follow policy π
- Action-value function $Q_\pi(\mathbf{x}, \mathbf{u})$ is the expected return of the resulting trajectory

Relationship Q_π and V_π

$$Q_\pi(\mathbf{x}, \pi(\mathbf{x})) = V_\pi(\mathbf{x})$$

Optimal policy & value function

- Ordering over policies: π is better or equal to π' if $V_{\pi}(\mathbf{x}) \leq V_{\pi'}(\mathbf{x})$ for all $\mathbf{x}$

Optimal policy & value function

- Ordering over policies: π is better or equal to π' if $V_{\pi}(\mathbf{x}) \leq V_{\pi'}(\mathbf{x})$ for all $\mathbf{x}$
- Optimal policy $\pi_{\star}$ defined by $V_{\pi_{\star}}(\mathbf{x}) \leq V_{\pi}(\mathbf{x})$ for all $\mathbf{x}$ and π (maybe not unique)

Optimal policy & value function

- Ordering over policies: π is better or equal to π' if $V_{\pi}(\mathbf{x}) \leq V_{\pi'}(\mathbf{x})$ for all $\mathbf{x}$
- Optimal policy $\pi_{\star}$ defined by $V_{\pi_{\star}}(\mathbf{x}) \leq V_{\pi}(\mathbf{x})$ for all $\mathbf{x}$ and π (maybe not unique)
- Define $V_{\star}(\mathbf{x}) = V_{\pi_{\star}}(\mathbf{x})$ the optimal value function (unique)

Optimal policy & value function

- Ordering over policies: π is better or equal to π' if $V_{\pi}(\mathbf{x}) \leq V_{\pi'}(\mathbf{x})$ for all $\mathbf{x}$
- Optimal policy $\pi_{\star}$ defined by $V_{\pi_{\star}}(\mathbf{x}) \leq V_{\pi}(\mathbf{x})$ for all $\mathbf{x}$ and π (maybe not unique)
- Define $V_{\star}(\mathbf{x}) = V_{\pi_{\star}}(\mathbf{x})$ the optimal value function (unique)

Optimal value function

$$V_{\star}(\mathbf{x}) = \min_{\pi} V_{\pi}(\mathbf{x})$$

Optimal policy & value function

- Ordering over policies: π is better or equal to π' if $V_{\pi}(\mathbf{x}) \leq V_{\pi'}(\mathbf{x})$ for all $\mathbf{x}$
- Optimal policy $\pi_{\star}$ defined by $V_{\pi_{\star}}(\mathbf{x}) \leq V_{\pi}(\mathbf{x})$ for all $\mathbf{x}$ and π (maybe not unique)
- Define $V_{\star}(\mathbf{x}) = V_{\pi_{\star}}(\mathbf{x})$ the optimal value function (unique)

Optimal value function

$$V_{\star}(\mathbf{x}) = \min_{\pi} V_{\pi}(\mathbf{x})$$

Note: this is optimization over a policy, i.e. on the functions $\pi(\mathbf{x})$, rather than on inputs!

Optimal policy & value function

- Ordering over policies: π is better or equal to π' if $V_{\pi}(\mathbf{x}) \leq V_{\pi'}(\mathbf{x})$ for all $\mathbf{x}$
- Optimal policy $\pi_\star$ defined by $V_{\pi_\star}(\mathbf{x}) \leq V_{\pi}(\mathbf{x})$ for all $\mathbf{x}$ and π (maybe not unique)
- Define $V_\star(\mathbf{x}) = V_{\pi_\star}(\mathbf{x})$ the optimal value function (unique)

Optimal value function

$$V_\star(\mathbf{x}) = \min_{\pi} V_{\pi}(\mathbf{x})$$

Note: this is optimization over a policy, i.e. on the functions $\pi(\mathbf{x})$, rather than on inputs!

Note:

$$Q_{\pi_\star}(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V_\star(\mathbf{x}_+) \mid \mathbf{x}, \mathbf{u}\right]$$

Optimal policy & value function

- Ordering over policies: π is better or equal to π' if $V_{\pi}(\mathbf{x}) \leq V_{\pi'}(\mathbf{x})$ for all $\mathbf{x}$
- Optimal policy $\pi_\star$ defined by $V_{\pi_\star}(\mathbf{x}) \leq V_{\pi}(\mathbf{x})$ for all $\mathbf{x}$ and π (maybe not unique)
- Define $V_\star(\mathbf{x}) = V_{\pi_\star}(\mathbf{x})$ the optimal value function (unique)

Optimal value function

$$V_\star(\mathbf{x}) = \min_{\pi} V_{\pi}(\mathbf{x})$$

Note: this is optimization over a policy, i.e. on the functions $\pi(\mathbf{x})$, rather than on inputs!

Note:

$$Q_{\pi_\star}(\mathbf{x},\mathbf{u}) = L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[V_\star(\mathbf{x}_+) \,|\, \mathbf{x},\mathbf{u}\right] \leq L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_+) \,|\, \mathbf{x},\mathbf{u}\right] = Q_\pi(\mathbf{x},\mathbf{u})$$

for all $\mathbf{x}, \mathbf{u}$ and π

Optimal policy & value function

- Ordering over policies: π is better or equal to π' if $V_{\pi}(\mathbf{x}) \leq V_{\pi'}(\mathbf{x})$ for all $\mathbf{x}$
- Optimal policy $\pi_\star$ defined by $V_{\pi_\star}(\mathbf{x}) \leq V_{\pi}(\mathbf{x})$ for all $\mathbf{x}$ and π (maybe not unique)
- Define $V_\star(\mathbf{x}) = V_{\pi_\star}(\mathbf{x})$ the optimal value function (unique)

Optimal value function

$$V_\star(\mathbf{x}) = \min_{\pi} V_{\pi}(\mathbf{x})$$

Note: this is optimization over a policy, i.e. on the functions $\pi(\mathbf{x})$, rather than on inputs!

Note:

$$Q_{\pi_\star}(\mathbf{x},\mathbf{u}) = L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[V_\star(\mathbf{x}_+) \,|\, \mathbf{x},\mathbf{u}\right] \leq L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_+) \,|\, \mathbf{x},\mathbf{u}\right] = Q_\pi(\mathbf{x},\mathbf{u})$$

for all $\mathbf{x}, \mathbf{u}$ and π

$Q_{\pi_\star}(\mathbf{x},\mathbf{u}) \leq Q_\pi(\mathbf{x},\mathbf{u})$
Hence minimizer $\pi_\star$ of V_π is minimizer of Q_π

Optimal policy & value function

- Ordering over policies: π is better or equal to π' if $V_{\pi}(\mathbf{x}) \leq V_{\pi'}(\mathbf{x})$ for all $\mathbf{x}$
- Optimal policy $\pi_\star$ defined by $V_{\pi_\star}(\mathbf{x}) \leq V_{\pi}(\mathbf{x})$ for all $\mathbf{x}$ and π (maybe not unique)
- Define $V_\star(\mathbf{x}) = V_{\pi_\star}(\mathbf{x})$ the optimal value function (unique)

Optimal value function

$$V_\star(\mathbf{x}) = \min_{\pi} V_{\pi}(\mathbf{x})$$

Note: this is optimization over a policy, i.e. on the functions $\pi(\mathbf{x})$, rather than on inputs!

Note:

$$Q_{\pi_\star}(\mathbf{x},\mathbf{u}) = L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[V_\star(\mathbf{x}_+) \,|\, \mathbf{x},\mathbf{u}\right] \leq L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_+) \,|\, \mathbf{x},\mathbf{u}\right] = Q_\pi(\mathbf{x},\mathbf{u})$$

for all $\mathbf{x}, \mathbf{u}$ and π

Optimal action-value function

$$Q_\star(\mathbf{x},\mathbf{u}) = \min_{\pi} Q_\pi(\mathbf{x},\mathbf{u}) = Q_{\pi_\star}(\mathbf{x},\mathbf{u})$$

$Q_{\pi_\star}(\mathbf{x},\mathbf{u}) \leq Q_\pi(\mathbf{x},\mathbf{u})$
Hence minimizer $\pi_\star$ of V_π is minimizer of Q_π

Bellman optimality equation

Relationships

$$V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; Q_\star(\mathbf{x}, \mathbf{u})$$

$$\boldsymbol{\pi}_\star(\mathbf{x}) = \arg\min_{\mathbf{u}} \; Q_\star(\mathbf{x}, \mathbf{u})$$

Bellman optimality equation

Relationships

$$V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; Q_\star(\mathbf{x}, \mathbf{u})$$

$$\boldsymbol{\pi}_\star(\mathbf{x}) = \arg\min_{\mathbf{u}} \; Q_\star(\mathbf{x}, \mathbf{u})$$

Yield Bellman optimality equation for value function:

$$V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V_\star(\mathbf{x}_+) \mid \mathbf{x}, \mathbf{u}\right]$$

Bellman optimality equation

Relationships

$$V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; Q_\star(\mathbf{x}, \mathbf{u})$$

$$\boldsymbol{\pi}_\star(\mathbf{x}) = \arg\min_{\mathbf{u}} \; Q_\star(\mathbf{x}, \mathbf{u})$$

Yield Bellman optimality equation for value function:

$$V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V_\star(\mathbf{x}_+) \mid \mathbf{x}, \mathbf{u}\right]$$

Note:

$$Q_\star(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V_\star(\mathbf{x}_+) \mid \mathbf{x}, \mathbf{u}\right]$$

Bellman optimality equation

Relationships

$$V_\star(\mathbf{x}) = \min_{\mathbf{u}} Q_\star(\mathbf{x}, \mathbf{u})$$

$$\boldsymbol{\pi}_\star(\mathbf{x}) = \arg\min_{\mathbf{u}} Q_\star(\mathbf{x}, \mathbf{u})$$

Yield Bellman optimality equation for value function:

$$V_\star(\mathbf{x}) = \min_{\mathbf{u}} L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V_\star(\mathbf{x}_+) \,|\, \mathbf{x}, \mathbf{u}\right]$$

Note:

$$Q_\star(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V_\star(\mathbf{x}_+) \,|\, \mathbf{x}, \mathbf{u}\right]$$

Yields Bellman optimality equation for action-value function:

$$Q_\star(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[\min_{\mathbf{u}'} Q_\star(\mathbf{x}_+, \mathbf{u}') \,|\, \mathbf{x}, \mathbf{u}\right]$$

Bellman optimality equation

Relationships

$$V_\star(\mathbf{x}) = \min_{\mathbf{u}} Q_\star(\mathbf{x}, \mathbf{u})$$

$$\boldsymbol{\pi}_\star(\mathbf{x}) = \arg\min_{\mathbf{u}} Q_\star(\mathbf{x}, \mathbf{u})$$

Yield Bellman optimality equation for value function:

$$V_\star(\mathbf{x}) = \min_{\mathbf{u}} L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V_\star(\mathbf{x}_+) \,|\, \mathbf{x}, \mathbf{u}\right]$$

Note:

$$Q_\star(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V_\star(\mathbf{x}_+) \,|\, \mathbf{x}, \mathbf{u}\right]$$

Yields Bellman optimality equation for action-value function:

$$Q_\star(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[\min_{\mathbf{u}'} Q_\star(\mathbf{x}_+, \mathbf{u}') \,|\, \mathbf{x}, \mathbf{u}\right]$$

Observe that $V_\star \to Q_\star$ exchanges $\mathbb{E}$ and $\min_{\mathbf{u}}$

Advantage function

Associated to a policy:

$$A_\pi(\mathbf{x}, \mathbf{u}) = Q_\pi(\mathbf{x}, \mathbf{u}) - V_\pi(\mathbf{x})$$

Properties:

$$A_\pi(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) = 0$$

$$A_\pi(\mathbf{x}, \mathbf{u}) < 0 \Rightarrow \mathbf{u} \text{ better than } \boldsymbol{\pi}(\mathbf{x})$$

- Assesses the value of an input $\mathbf{u}$ compared to a base policy (for given state)
- When $A_\pi(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) \geq 0$ for all $\mathbf{x}$, we have an optimal policy
- Will be useful for policy gradient methods

Note:

- $A_\pi(\mathbf{x}, \mathbf{u}) < 0 \Rightarrow \mathbf{u}$ better than $\boldsymbol{\pi}(\mathbf{x})$ for $\mathbf{x}$
- $A_\pi(\mathbf{x}, \mathbf{u}) > 0 \Rightarrow \mathbf{u}$ worse than $\boldsymbol{\pi}(\mathbf{x})$ for $\mathbf{x}$

allows for "comparing" arbitrary input $\mathbf{u}$ to baseline policy

Advantage function

Associated to a policy:

$$A_\pi(\mathbf{x}, \mathbf{u}) = Q_\pi(\mathbf{x}, \mathbf{u}) - V_\pi(\mathbf{x})$$

Properties:

$$A_\pi(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) = 0$$

$$A_\pi(\mathbf{x}, \mathbf{u}) < 0 \Rightarrow \mathbf{u} \text{ better than } \boldsymbol{\pi}(\mathbf{x})$$

Optimal:

$$A_\star(\mathbf{x}, \mathbf{u}) = Q_\star(\mathbf{x}, \mathbf{u}) - V_\star(\mathbf{x})$$

Properties:

$$A_\star(\mathbf{x}, \boldsymbol{\pi}_\star(\mathbf{x})) = 0$$

$$A_\star(\mathbf{x}, \mathbf{u}) \geq 0$$

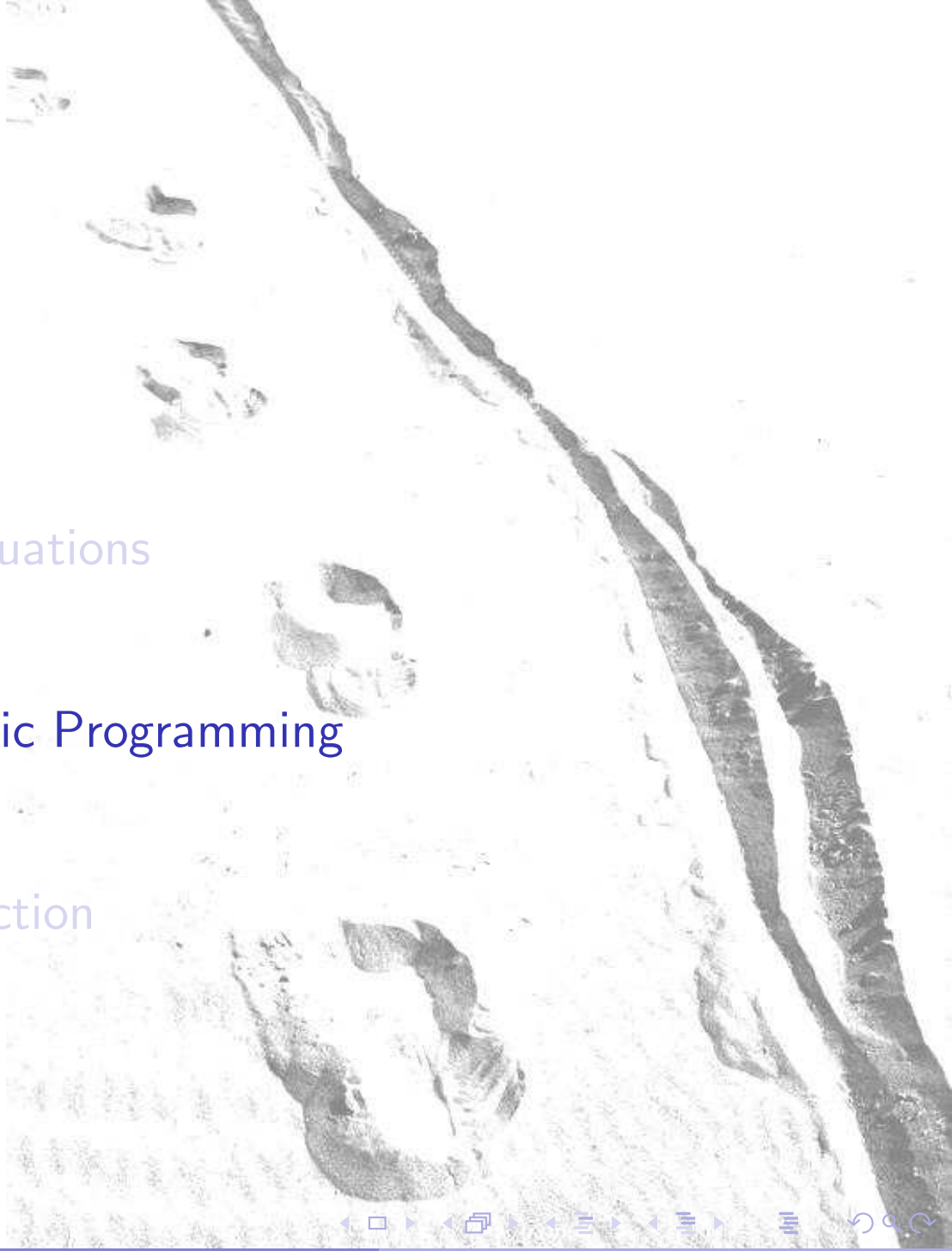
- Assesses the value of an input $\mathbf{u}$ compared to a base policy (for given state)
- When $A_\pi(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) \geq 0$ for all $\mathbf{x}$, we have an optimal policy
- Will be useful for policy gradient methods

Note:

- $A_\pi(\mathbf{x}, \mathbf{u}) < 0 \Rightarrow \mathbf{u}$ better than $\boldsymbol{\pi}(\mathbf{x})$ for $\mathbf{x}$
- $A_\pi(\mathbf{x}, \mathbf{u}) > 0 \Rightarrow \mathbf{u}$ worse than $\boldsymbol{\pi}(\mathbf{x})$ for $\mathbf{x}$

allows for "comparing" arbitrary input $\mathbf{u}$ to baseline policy

Outline



How to compute value functions?

How to solve the Bellman equations?

$$V_{\pi}(\mathbf{x}) = L(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) + \gamma \mathbb{E}\left[V_{\pi}(\mathbf{x}_+) \,|\, \mathbf{x}, \boldsymbol{\pi}\right] \qquad V_{\pi}(\mathbf{x}) = Q_{\pi}(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x}))$$

$$Q_{\pi}(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V_{\pi}(\mathbf{x}_+) \,|\, \mathbf{x}, \mathbf{u}\right]$$

How to compute value functions?

How to solve the Bellman equations?

$$V_\pi(\mathbf{x}) = L(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_+) \,|\, \mathbf{x}, \boldsymbol{\pi}\right] \qquad V_\pi(\mathbf{x}) = Q_\pi(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x}))$$
$$Q_\pi(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_+) \,|\, \mathbf{x}, \mathbf{u}\right]$$

Policy evaluation: for a given policy $\boldsymbol{\pi}$ start with arbitrary value function $V_0(\mathbf{x})$, iterate:

$$V_{k+1}(\mathbf{x}) \leftarrow L(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) + \gamma \mathbb{E}\left[V_k(\mathbf{x}_+) \,|\, \mathbf{x}, \boldsymbol{\pi}\right] \quad \text{(sweep over } \mathbf{x}\text{)}$$

How to compute value functions?

How to solve the Bellman equations?

$$V_\pi(\mathbf{x}) = L(\mathbf{x}, \pi(\mathbf{x})) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_+) \,|\, \mathbf{x}, \pi\right] \qquad V_\pi(\mathbf{x}) = Q_\pi(\mathbf{x}, \pi(\mathbf{x}))$$
$$Q_\pi(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_+) \,|\, \mathbf{x}, \mathbf{u}\right]$$

Policy evaluation: for a given policy π start with arbitrary value function $V_0(\mathbf{x})$, iterate:

$$V_{k+1}(\mathbf{x}) \leftarrow L(\mathbf{x}, \pi(\mathbf{x})) + \gamma \mathbb{E}\left[V_k(\mathbf{x}_+) \,|\, \mathbf{x}, \pi\right] \quad \text{(sweep over } \mathbf{x}\text{)}$$

Theorem

For $\gamma < 1$

$$\lim_{k \to \infty} V_k(\mathbf{x}) = V_\pi(\mathbf{x})$$

holds for any $V_0(\mathbf{x})$ (finite)

How to compute value functions?

How to solve the Bellman equations?

$$V_\pi(\mathbf{x}) = L(\mathbf{x}, \pi(\mathbf{x})) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_+) \,|\, \mathbf{x}, \pi\right] \qquad V_\pi(\mathbf{x}) = Q_\pi(\mathbf{x}, \pi(\mathbf{x}))$$
$$Q_\pi(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V_\pi(\mathbf{x}_+) \,|\, \mathbf{x}, \mathbf{u}\right]$$

Policy evaluation: for a given policy π start with arbitrary value function $V_0(\mathbf{x})$, iterate:

$$V_{k+1}(\mathbf{x}) \leftarrow L(\mathbf{x}, \pi(\mathbf{x})) + \gamma \mathbb{E}\left[V_k(\mathbf{x}_+) \,|\, \mathbf{x}, \pi\right] \quad \text{(sweep over } \mathbf{x}\text{)}$$

Theorem

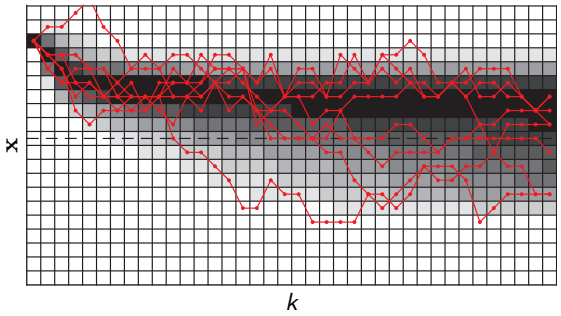
For $\gamma < 1$

$$\lim_{k \to \infty} V_k(\mathbf{x}) = V_\pi(\mathbf{x})$$

holds for any $V_0(\mathbf{x})$ (finite)

Note: convergence for $\gamma = 1$ needs extra assumptions!

Computing V_π - Illustration

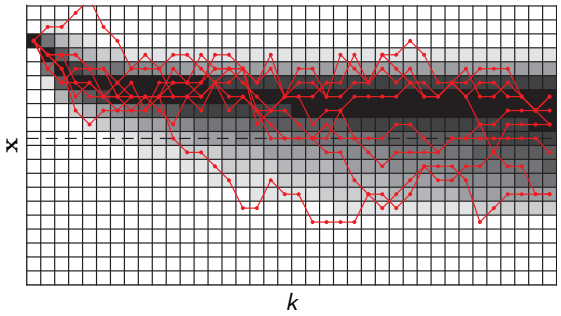


Start with e.g. $V(\mathbf{x}) = 0$ and iterate

$$V(\mathbf{x}) \leftarrow L(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \boldsymbol{\pi}\right]$$

as a sweep over all $\mathbf{x}$

Computing V_π - Illustration



Start with e.g. $V(\mathbf{x}) = 0$ and iterate

$$V(\mathbf{x}) \leftarrow L(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \boldsymbol{\pi}\right]$$

as a sweep over all $\mathbf{x}$

Algorithm: π-evaluation

Input: V (e.g. $= 0$) and $\mathrm{tol} > 0$

$\mathrm{Iter} = \mathrm{True}$

while Iter **do**

  for *All* $\mathbf{x}$ **do**

    $V_+(\mathbf{x}) \leftarrow L(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \boldsymbol{\pi}\right]$

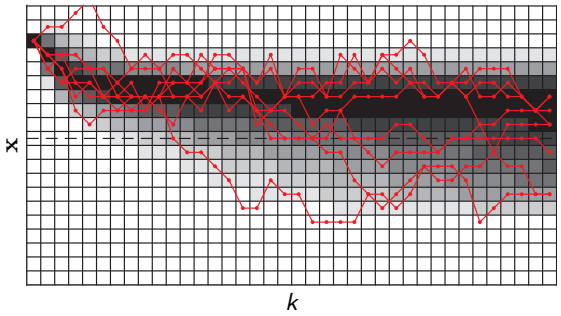
  if $\|V_+ - V\|_\infty \leq \mathrm{tol}$ **then**

    $\mathrm{Iter} = \mathrm{False}$

  $V \leftarrow V_+$

return $V_\pi \leftarrow V_+$

Computing V_π - Illustration



Start with e.g. $V(\mathbf{x}) = 0$ and iterate

$$V(\mathbf{x}) \leftarrow L(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \boldsymbol{\pi}\right]$$

as a sweep over all $\mathbf{x}$

Algorithm: π-evaluation

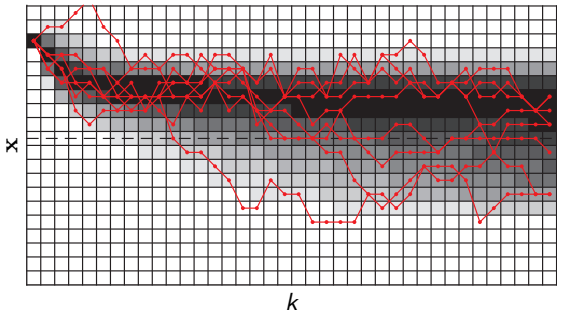
Input: V (e.g. $= 0$) and $\mathrm{tol} > 0$
$\mathrm{Iter} = \mathrm{True}$
while Iter **do**
  for *All* $\mathbf{x}$ **do**
    $V_+(\mathbf{x}) \leftarrow L(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \boldsymbol{\pi}\right]$
  if $\|V_+ - V\|_\infty \leq \mathrm{tol}$ **then**
    $\mathrm{Iter} = \mathrm{False}$
  $V \leftarrow V_+$
return $V_\pi \leftarrow V_+$

Remarks:

- Coding π-evaluation requires a discretized state-space $\mathbf{x} \in \mathbb{X}$
- V, V_+ are stored as tables (say of size n)
- Each iteration requires n^2 accesses
- Expected value for discrete state

$$\mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \boldsymbol{\pi}\right] = \sum_{\mathbf{x}_+ \in \mathbb{X}} V\left(\mathbf{x}_+\right) \underbrace{\mathbb{P}[\mathbf{x}_+ \mid \mathbf{x}, \boldsymbol{\pi}(\mathbf{x})]}_{\textit{Model}}$$

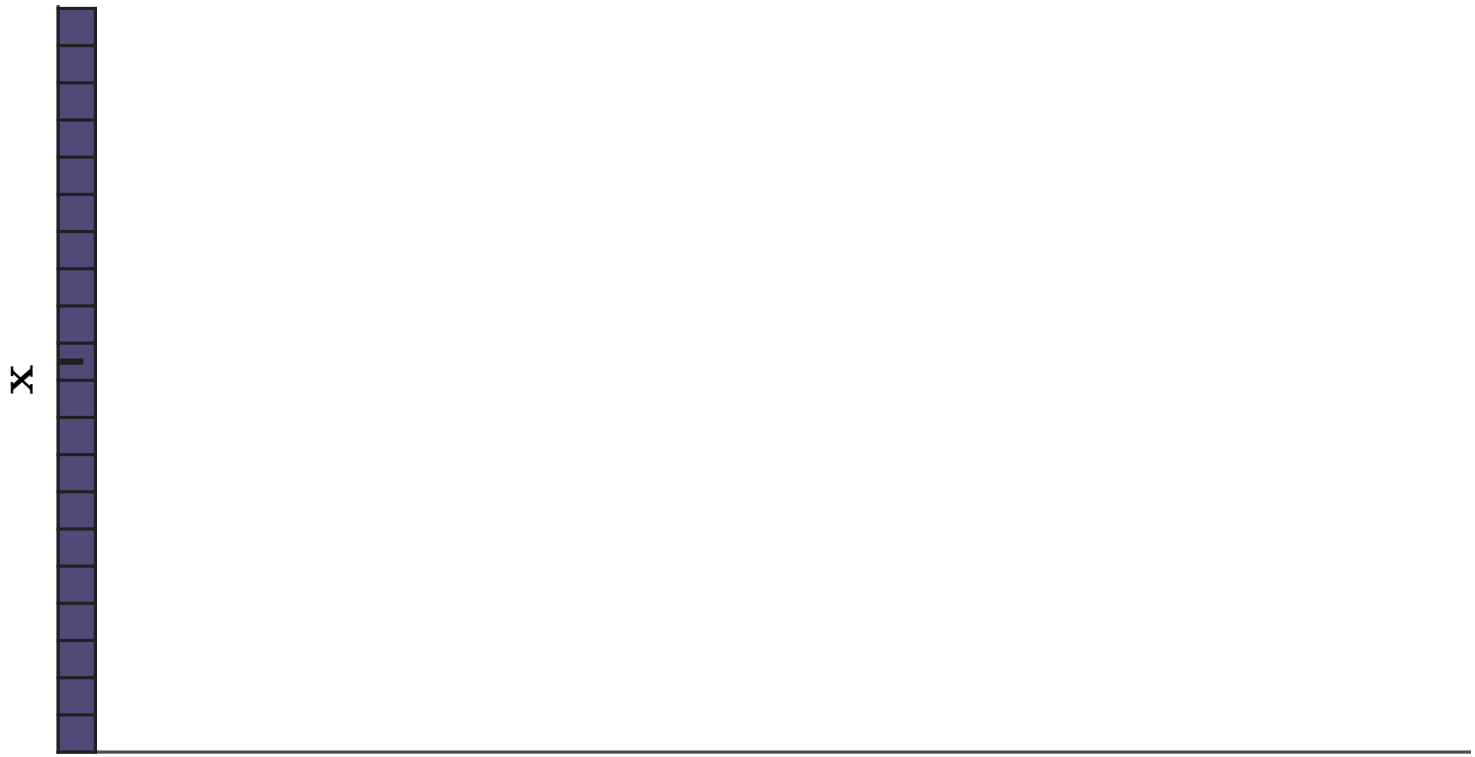
Computing V_π - Illustration



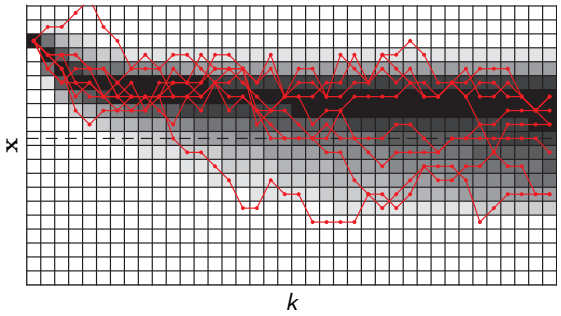
Start with e.g. $V(\mathbf{x}) = 0$ and iterate

$$V(\mathbf{x}) \leftarrow L(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \boldsymbol{\pi}\right]$$

as a sweep over all $\mathbf{x}$



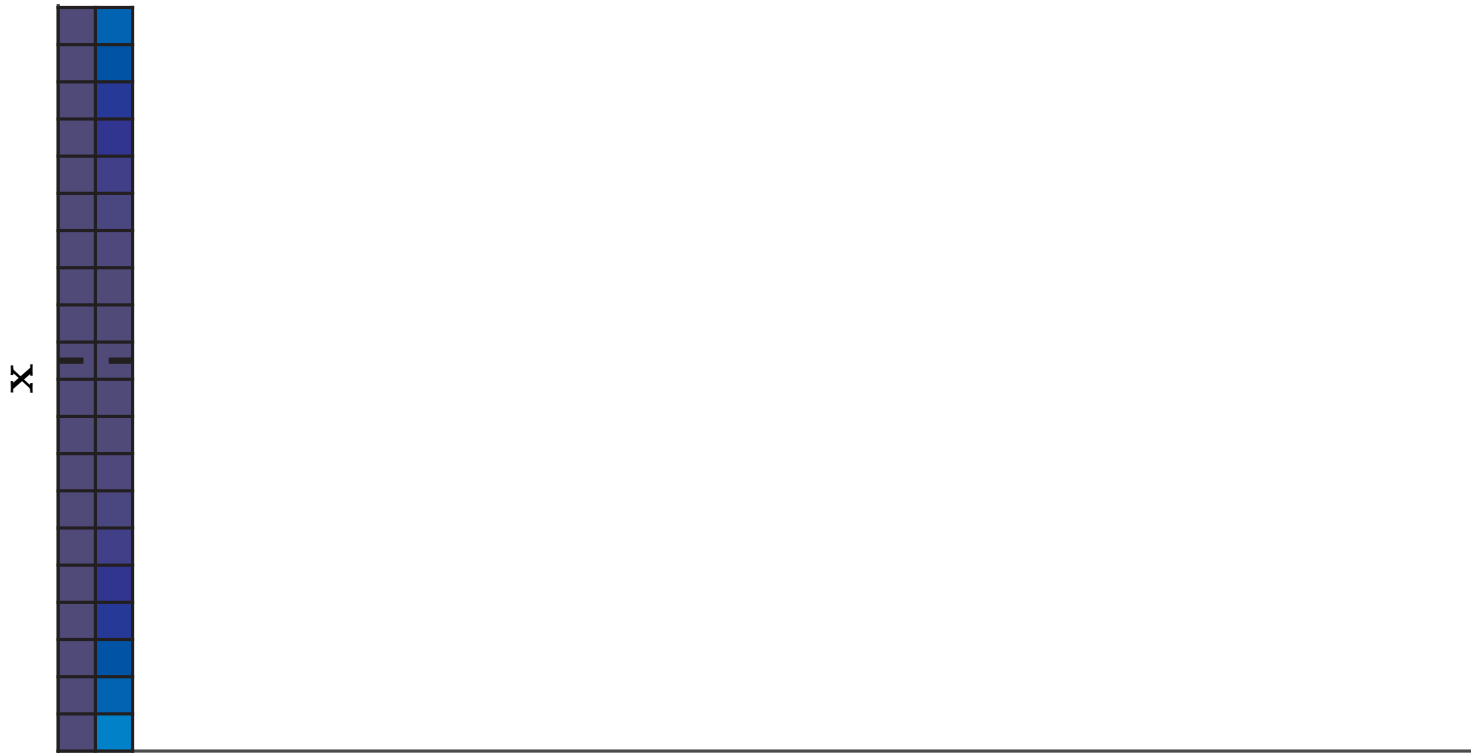
Computing V_π - Illustration



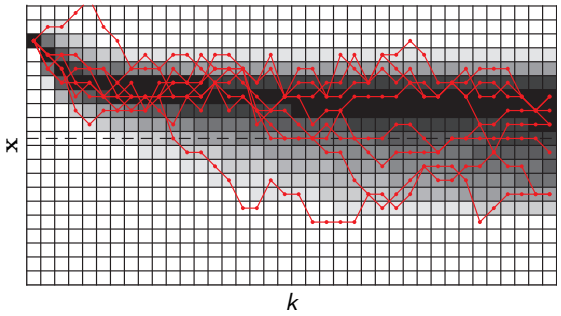
Start with e.g. $V(\mathbf{x}) = 0$ and iterate

$$V(\mathbf{x}) \leftarrow L(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \boldsymbol{\pi}\right]$$

as a sweep over all $\mathbf{x}$



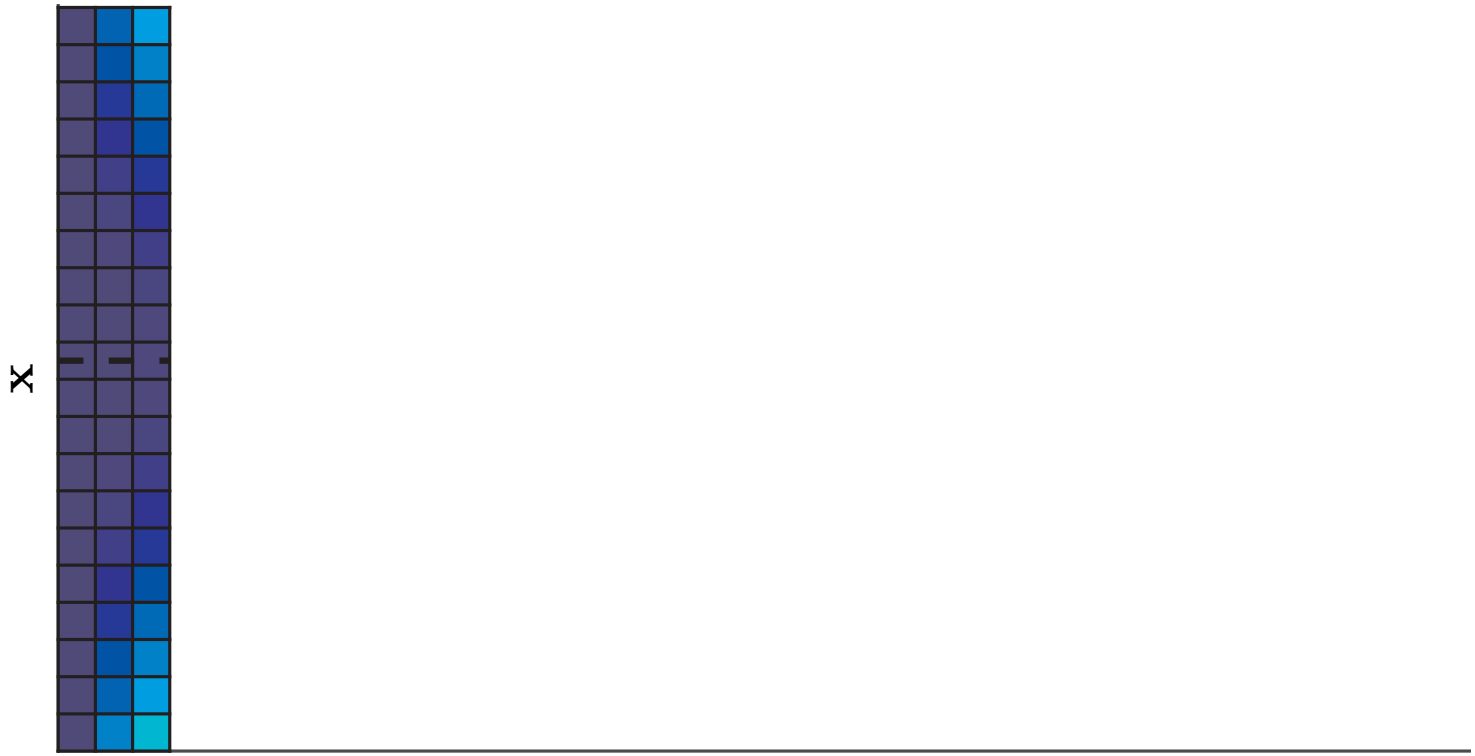
Computing V_π - Illustration



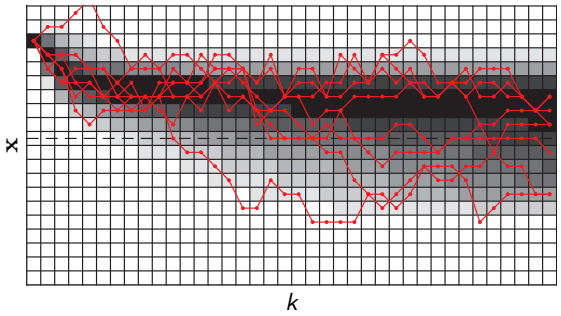
Start with e.g. $V(\mathbf{x}) = 0$ and iterate

$$V(\mathbf{x}) \leftarrow L(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \boldsymbol{\pi}\right]$$

as a sweep over all $\mathbf{x}$



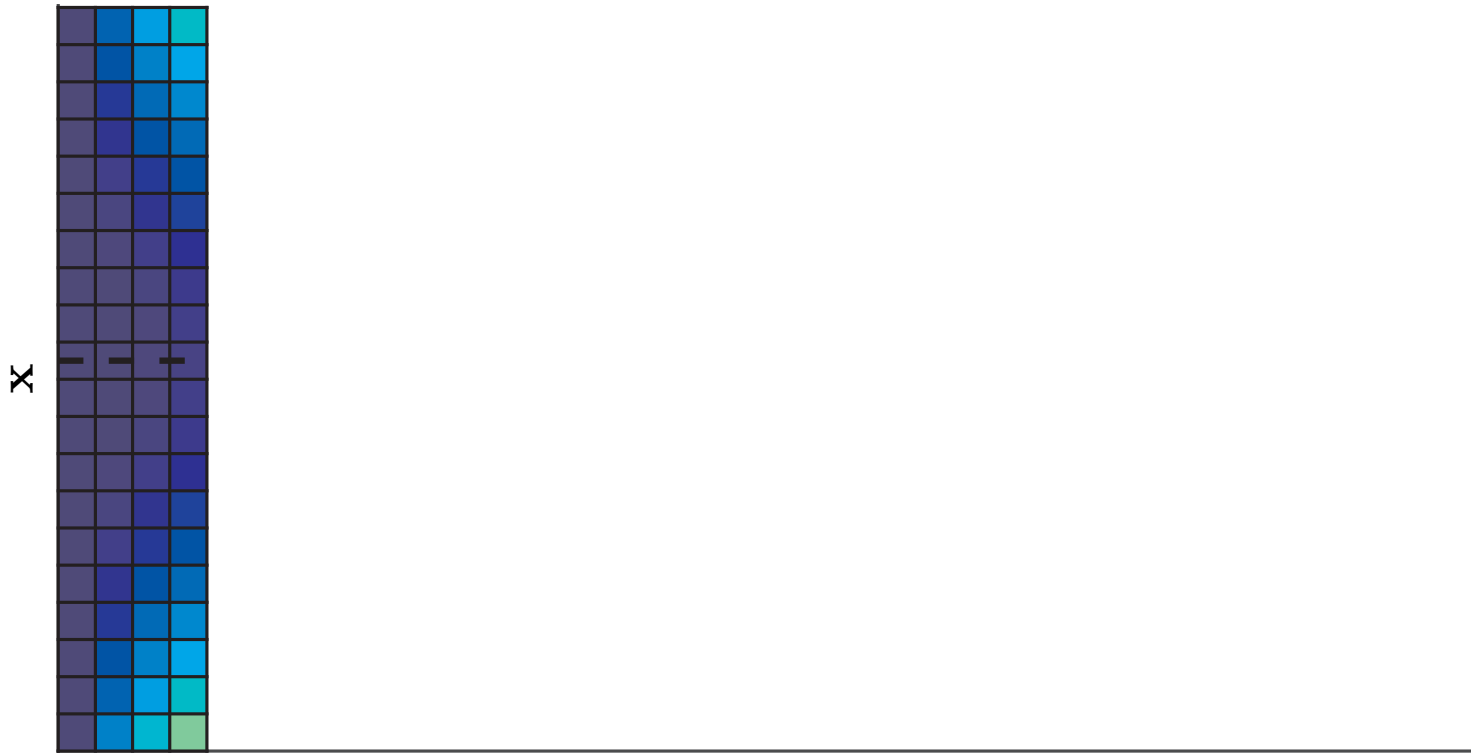
Computing V_π - Illustration



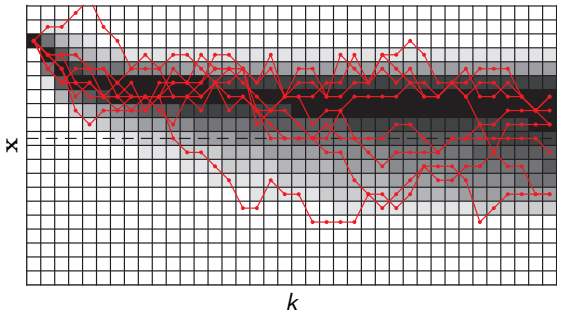
Start with e.g. $V(\mathbf{x}) = 0$ and iterate

$$V(\mathbf{x}) \leftarrow L(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \boldsymbol{\pi}\right]$$

as a sweep over all $\mathbf{x}$



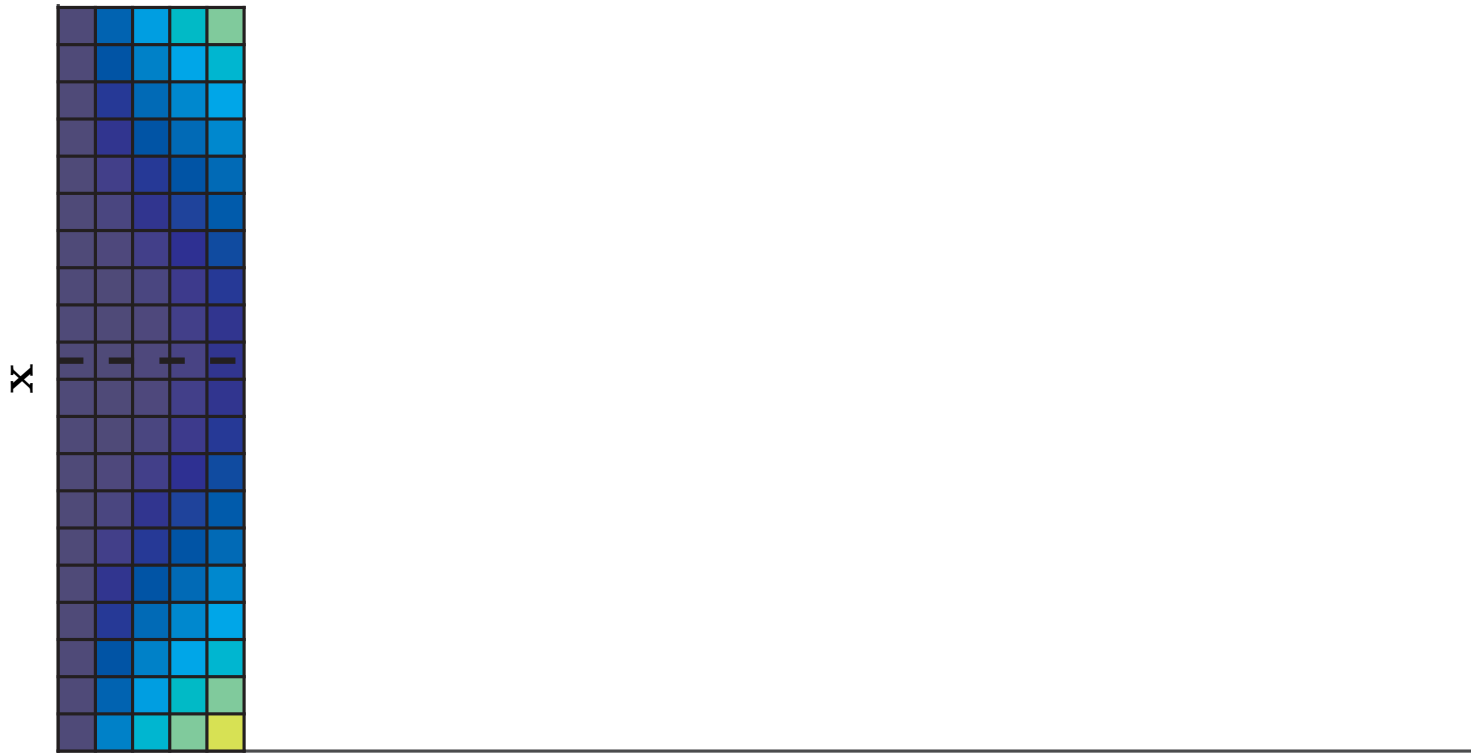
Computing V_π - Illustration



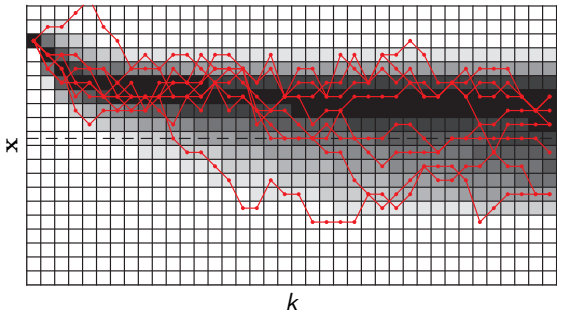
Start with e.g. $V(\mathbf{x}) = 0$ and iterate

$$V(\mathbf{x}) \leftarrow L(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \boldsymbol{\pi}\right]$$

as a sweep over all $\mathbf{x}$



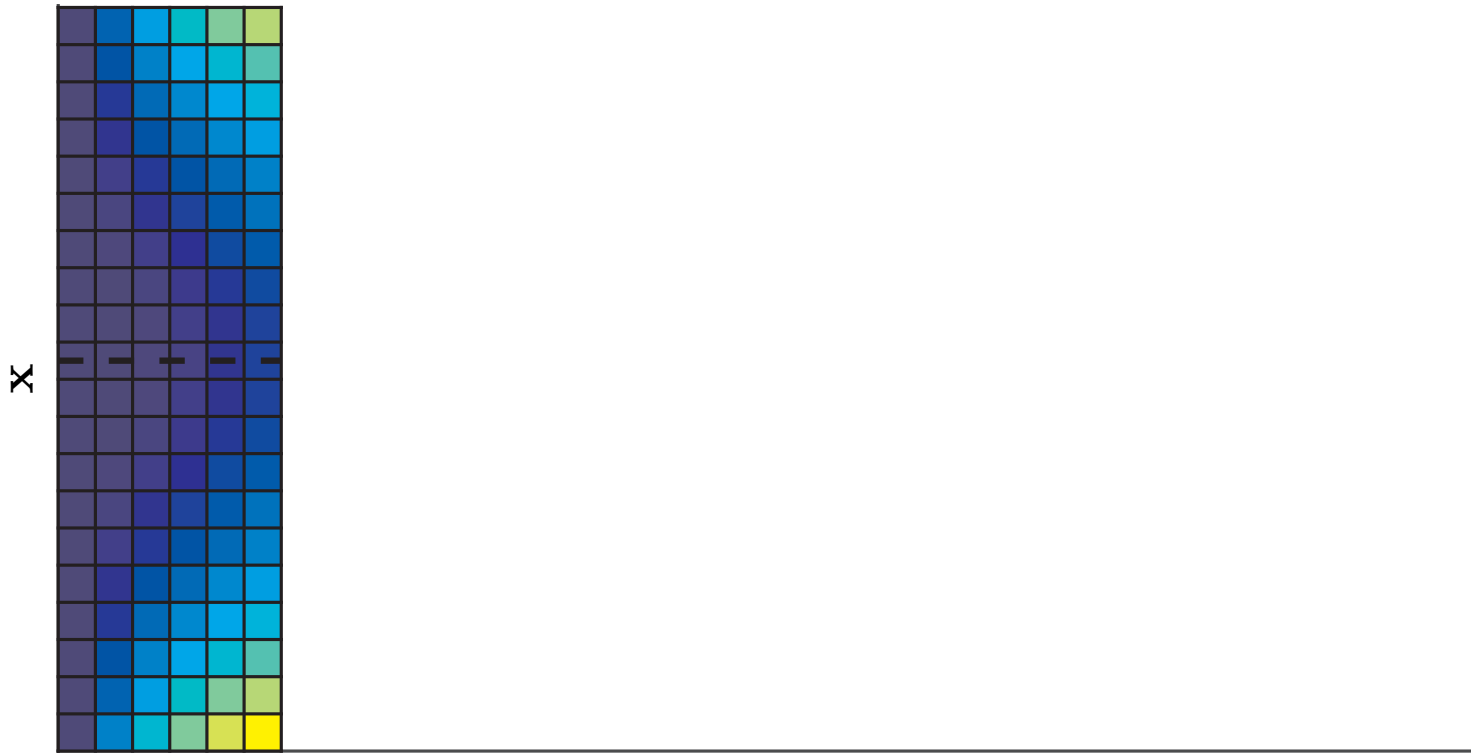
Computing V_π - Illustration



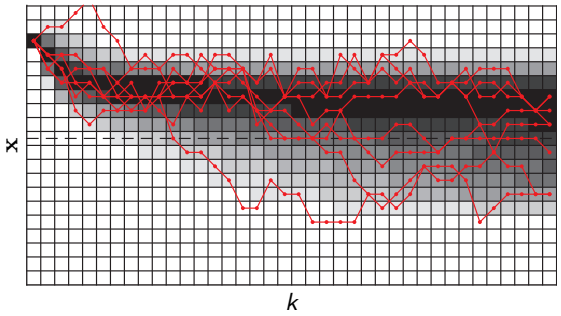
Start with e.g. $V(\mathbf{x}) = 0$ and iterate

$$V(\mathbf{x}) \leftarrow L(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \boldsymbol{\pi}\right]$$

as a sweep over all $\mathbf{x}$



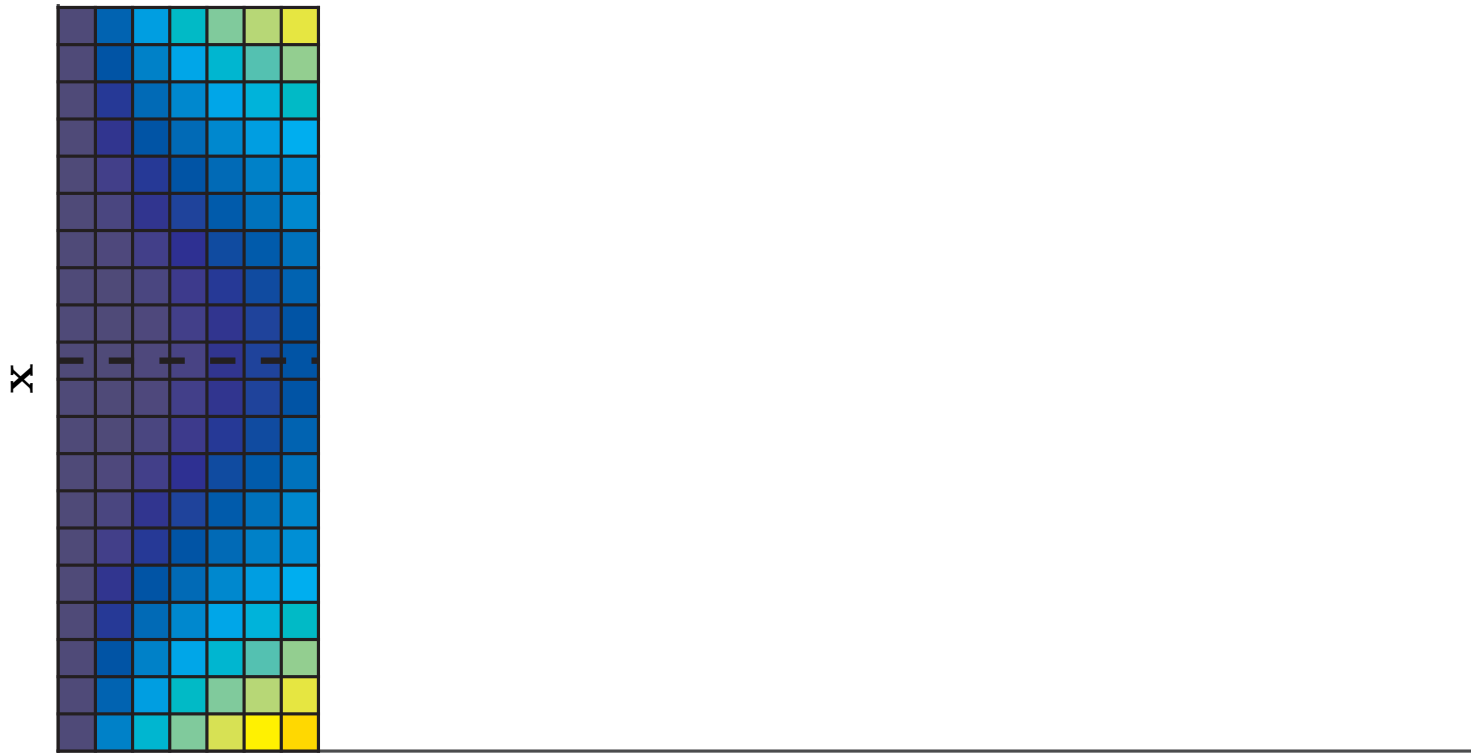
Computing V_π - Illustration



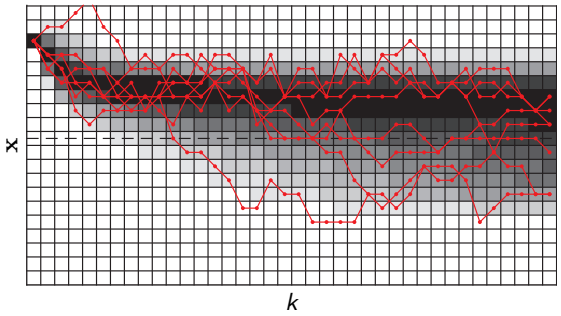
Start with e.g. $V(\mathbf{x}) = 0$ and iterate

$$V(\mathbf{x}) \leftarrow L(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \boldsymbol{\pi}\right]$$

as a sweep over all $\mathbf{x}$



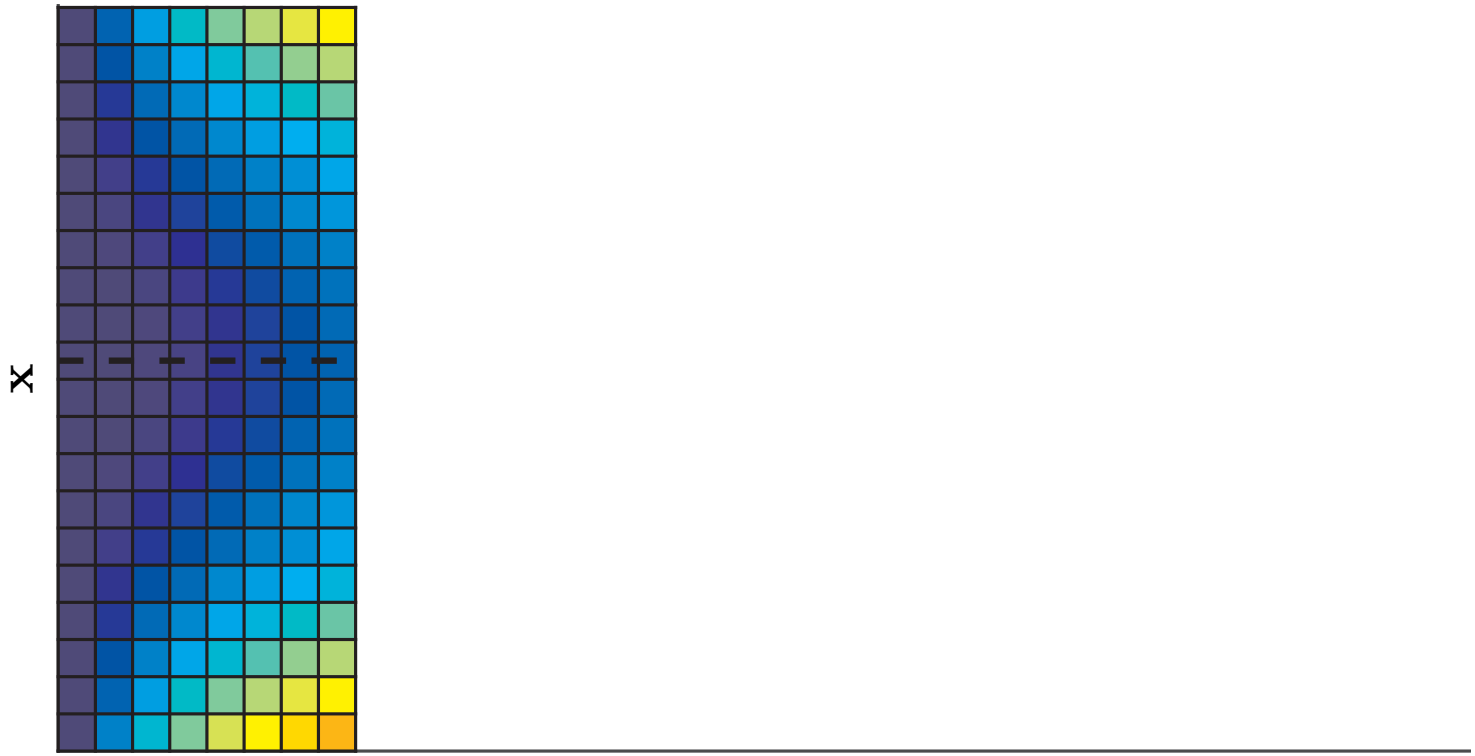
Computing V_π - Illustration



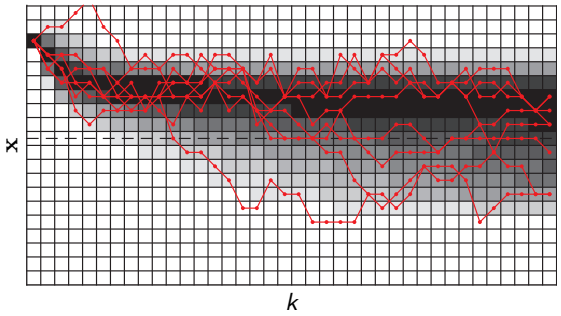
Start with e.g. $V(\mathbf{x}) = 0$ and iterate

$$V(\mathbf{x}) \leftarrow L(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \boldsymbol{\pi}\right]$$

as a sweep over all $\mathbf{x}$



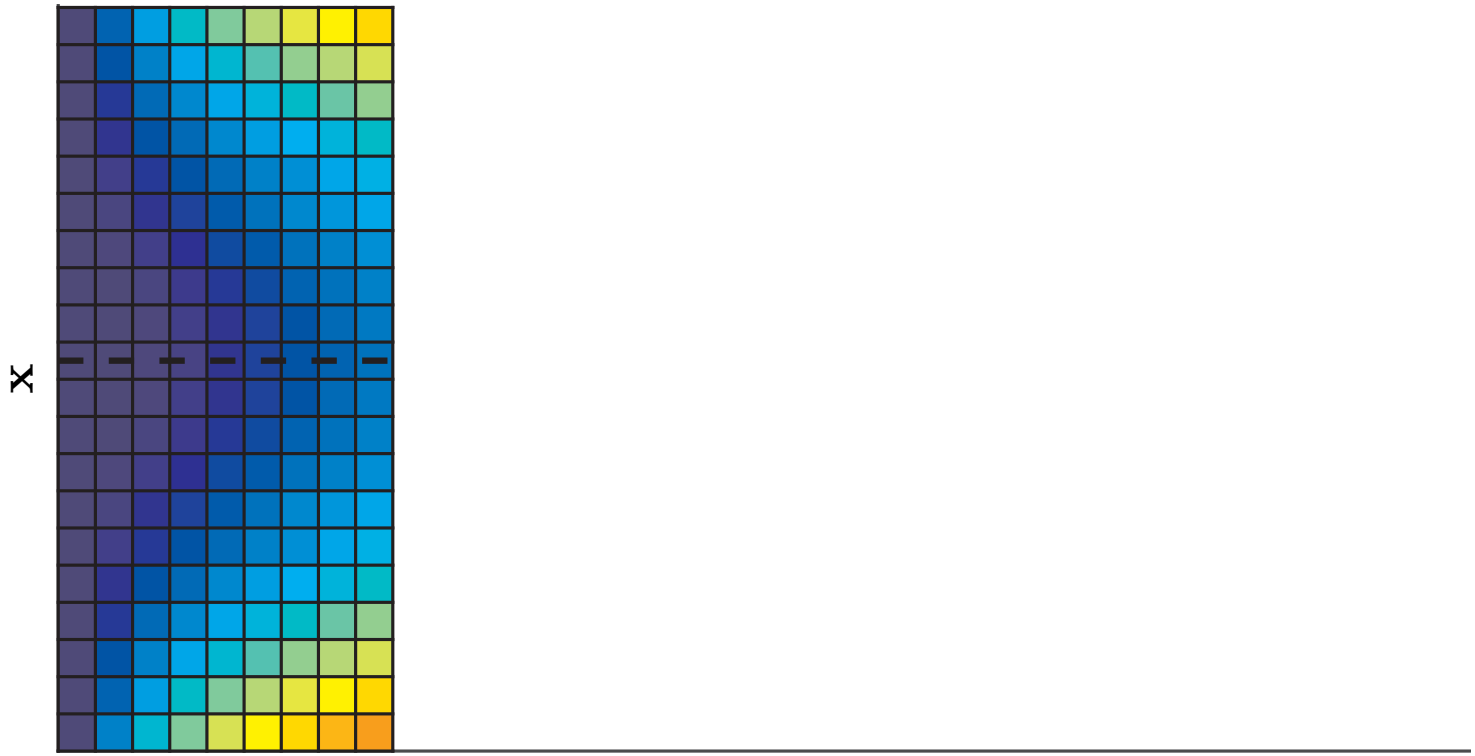
Computing V_π - Illustration



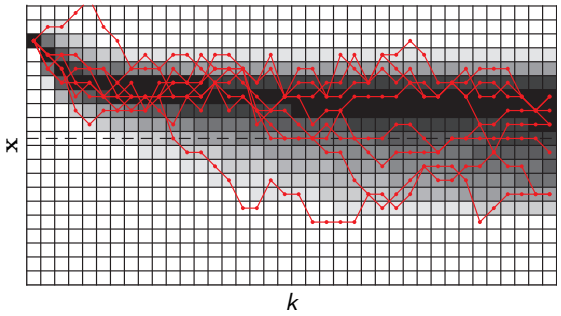
Start with e.g. $V(\mathbf{x}) = 0$ and iterate

$$V(\mathbf{x}) \leftarrow L(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \boldsymbol{\pi}\right]$$

as a sweep over all $\mathbf{x}$



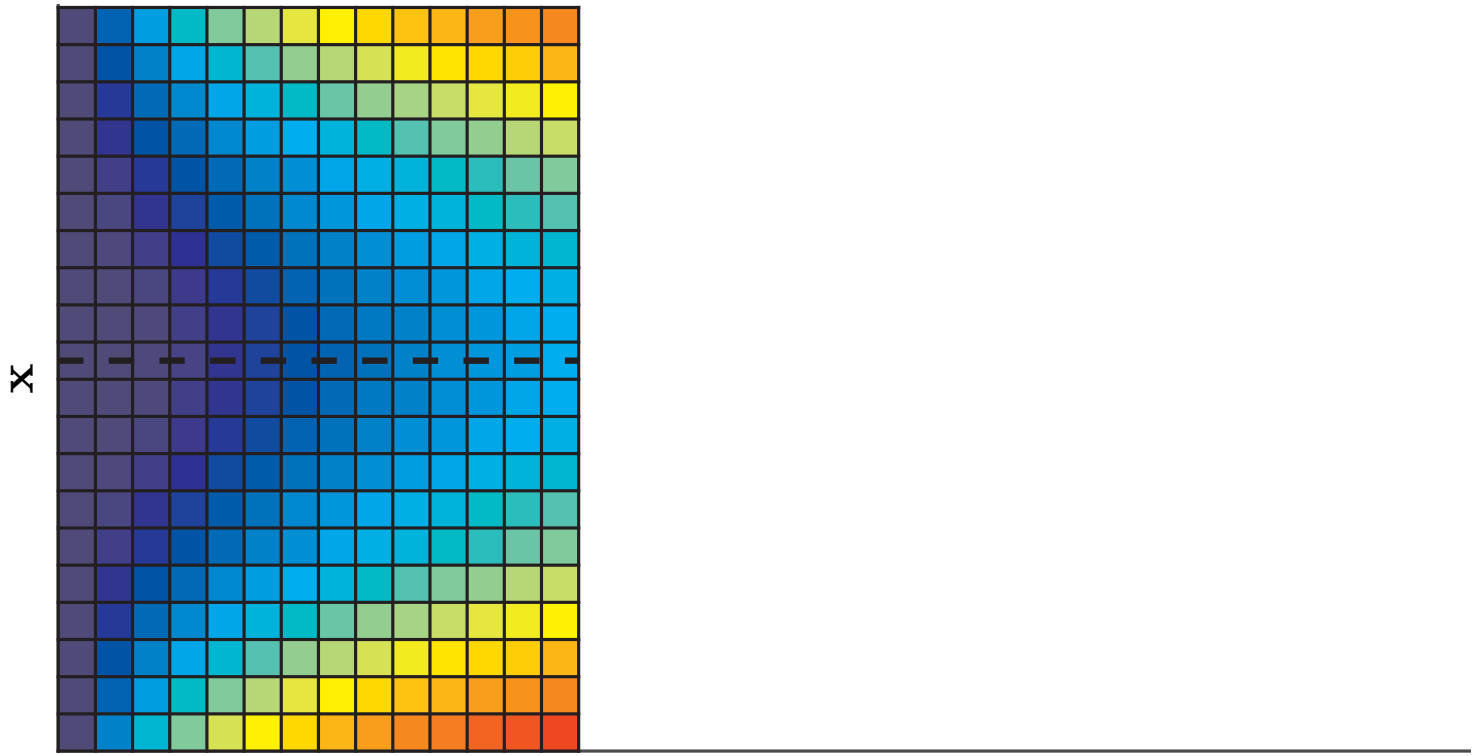
Computing V_π - Illustration



Start with e.g. $V(\mathbf{x}) = 0$ and iterate

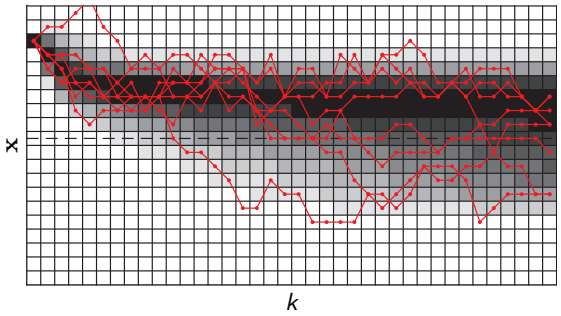
$$V(\mathbf{x}) \leftarrow L(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \boldsymbol{\pi}\right]$$

as a sweep over all $\mathbf{x}$



Iteration of $\boldsymbol{\pi}$ evaluation

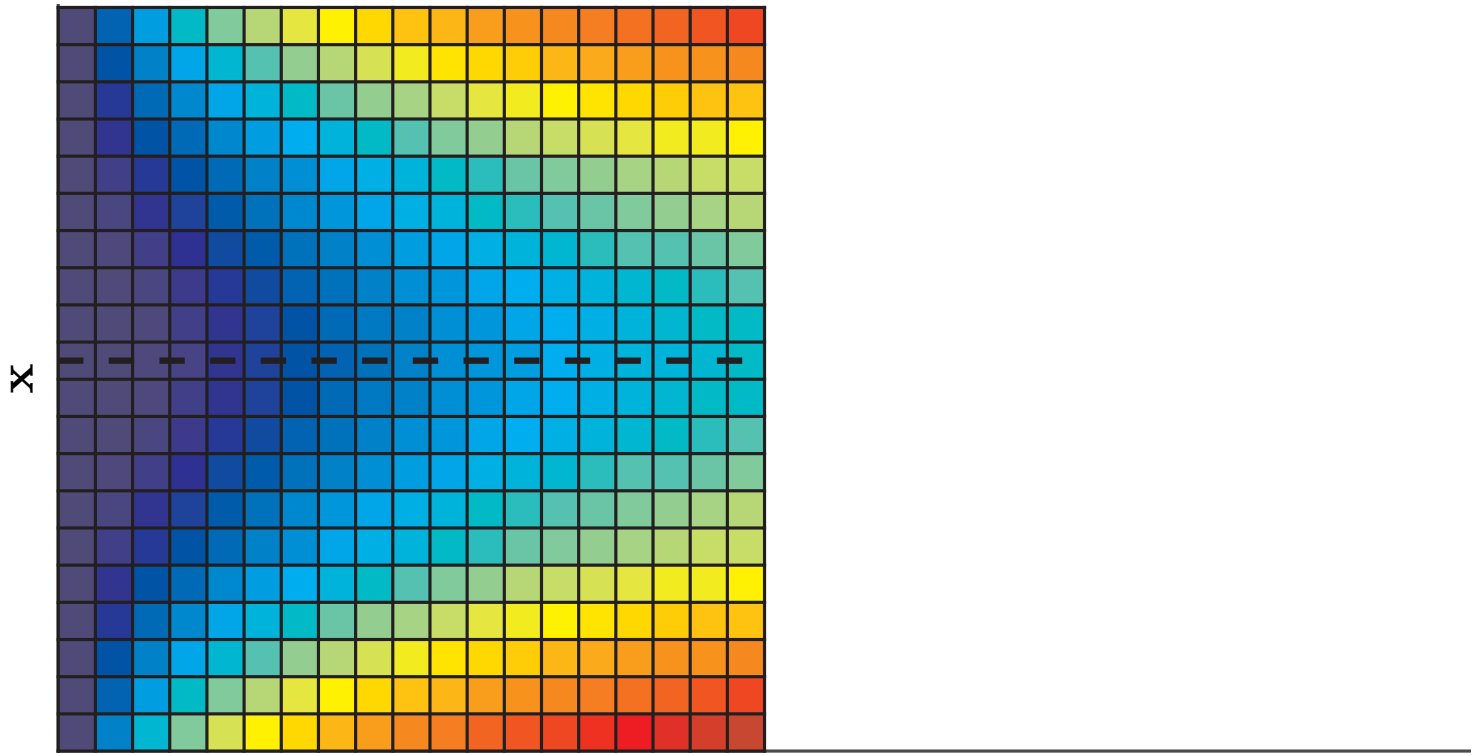
Computing V_π - Illustration



Start with e.g. $V(\mathbf{x}) = 0$ and iterate

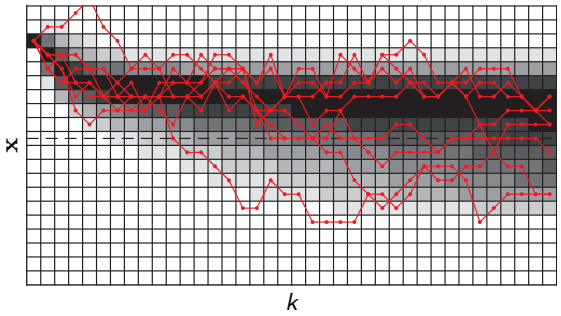
$$V(\mathbf{x}) \leftarrow L(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \boldsymbol{\pi}\right]$$

as a sweep over all $\mathbf{x}$



Iteration of $\boldsymbol{\pi}$ evaluation

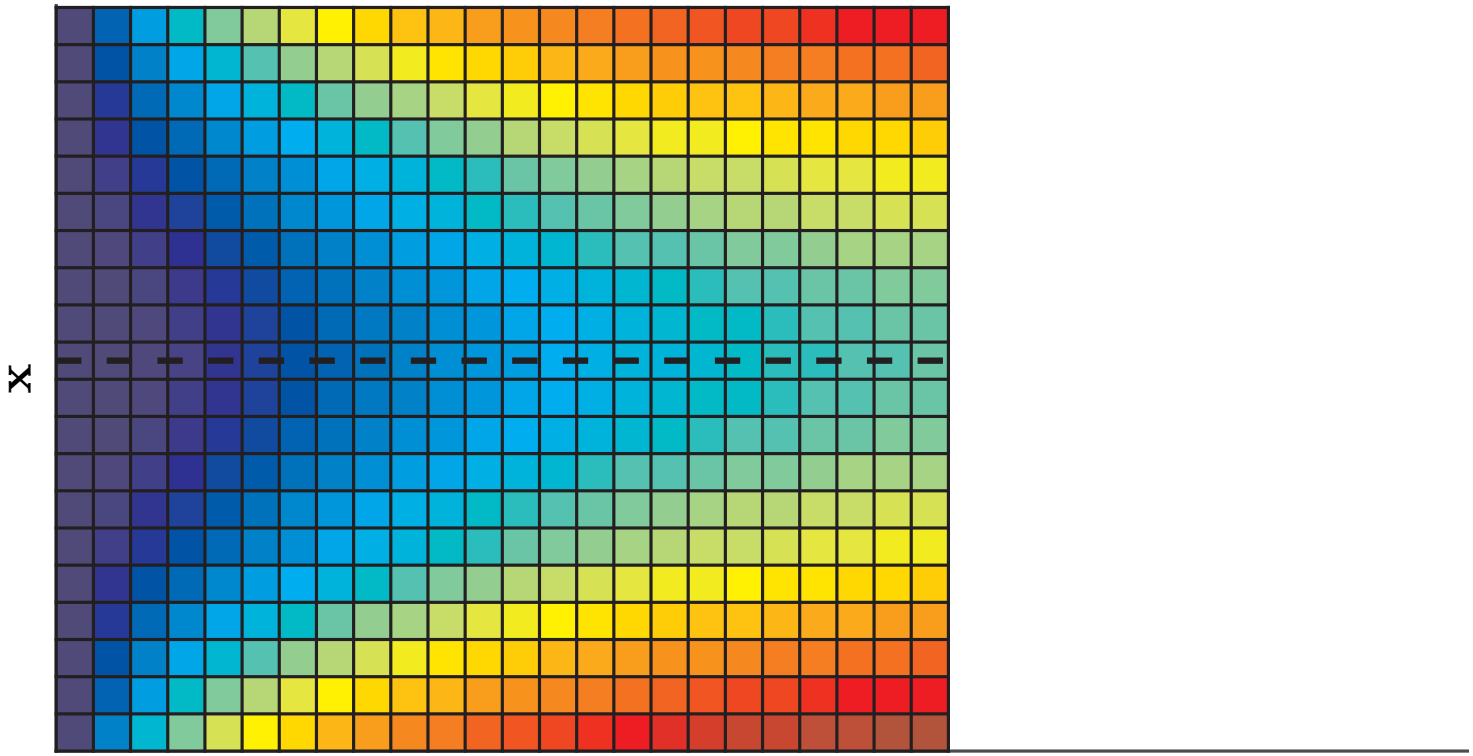
Computing V_π - Illustration



Start with e.g. $V(\mathbf{x}) = 0$ and iterate

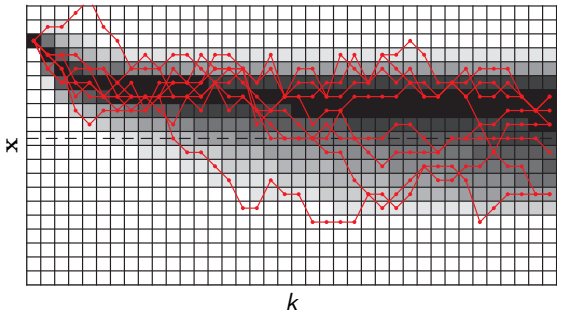
$$V(\mathbf{x}) \leftarrow L(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \boldsymbol{\pi}\right]$$

as a sweep over all $\mathbf{x}$



Iteration of $\boldsymbol{\pi}$ evaluation

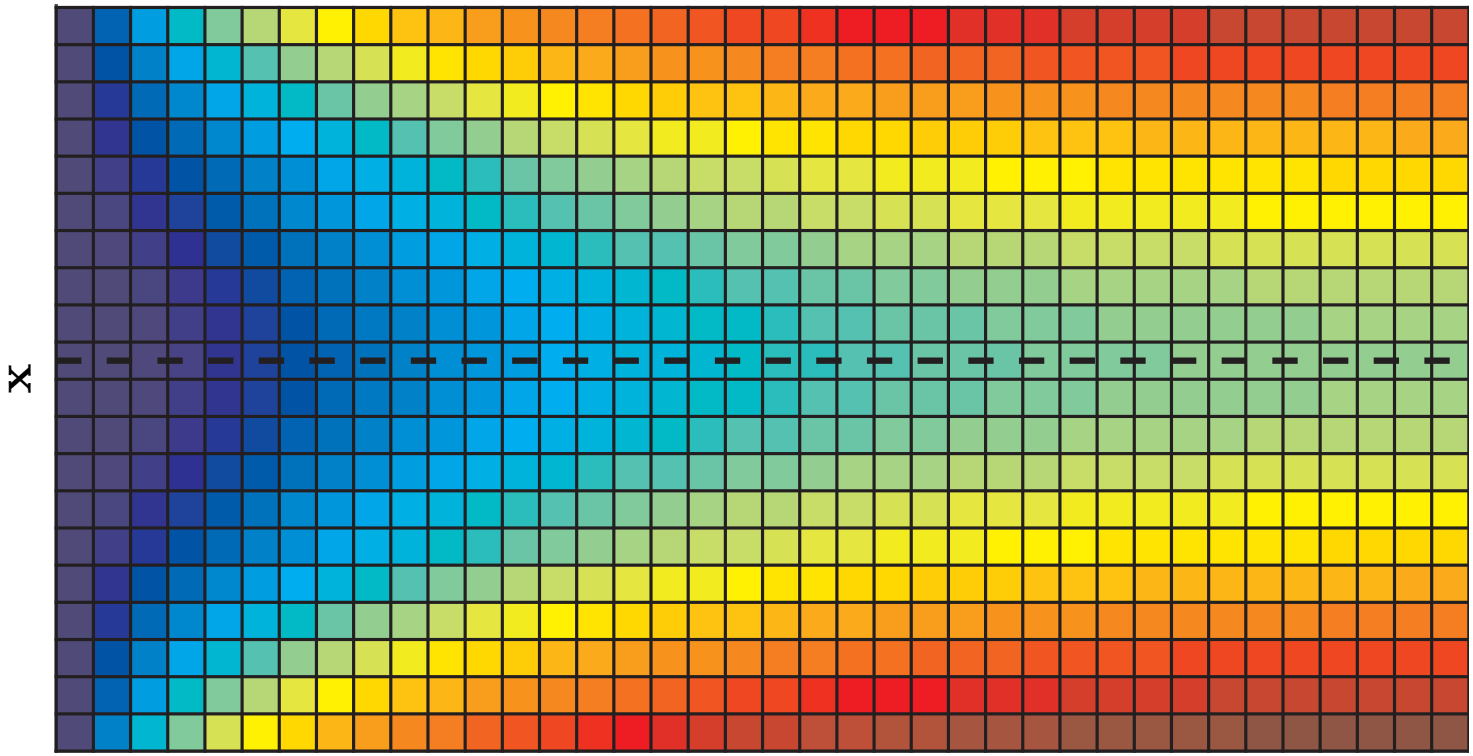
Computing V_π - Illustration



Start with e.g. $V(\mathbf{x}) = 0$ and iterate

$$V(\mathbf{x}) \leftarrow L(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x})) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \boldsymbol{\pi}\right]$$

as a sweep over all $\mathbf{x}$



Iteration of $\boldsymbol{\pi}$ evaluation

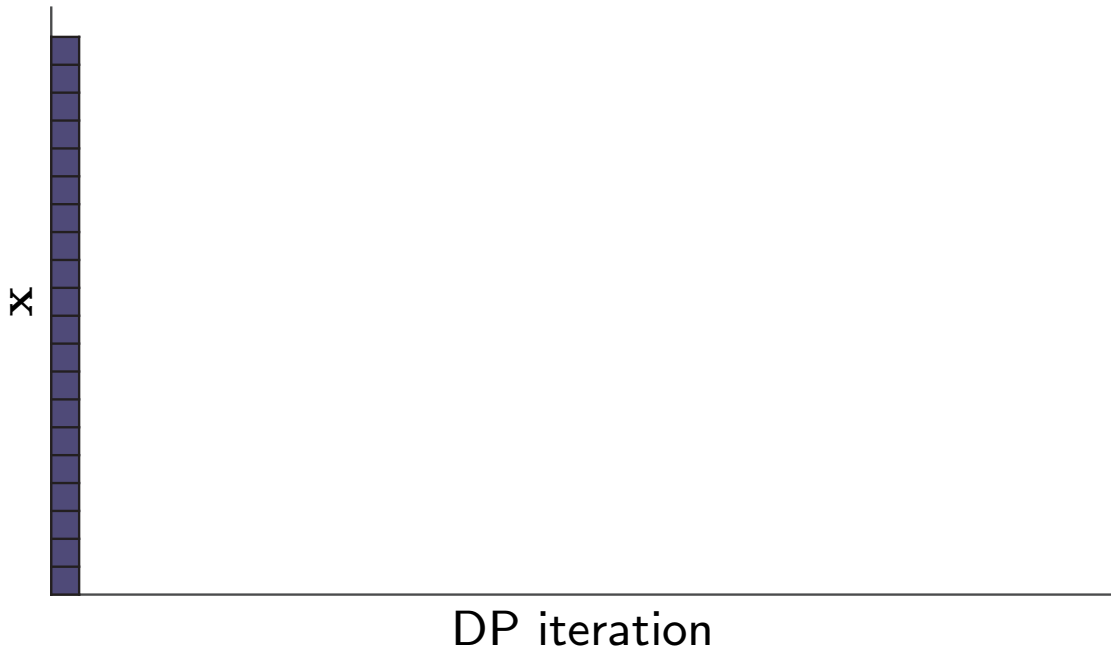
Computing Q_π - Illustration

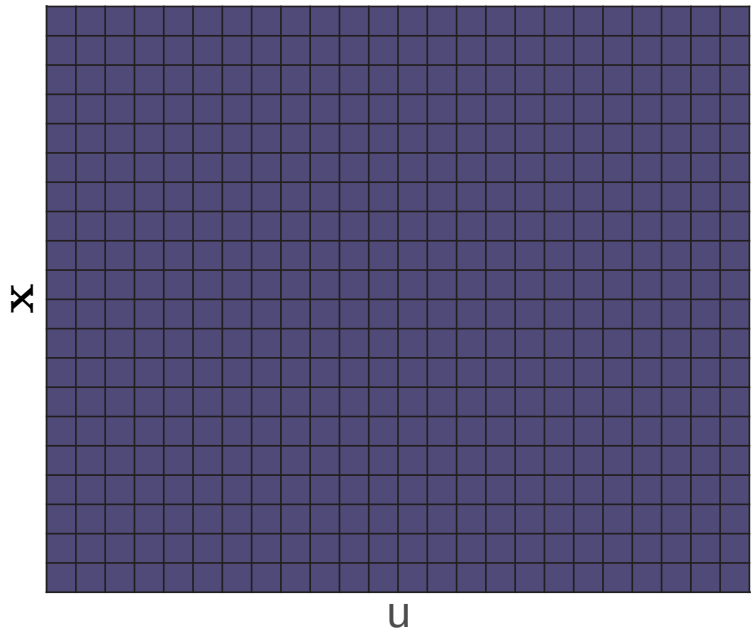
Can build V_π, Q_π jointly using:

$$Q(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V(\mathbf{x}_+) \,|\, \mathbf{x}, \mathbf{u}\right]$$
$$V(\mathbf{x}) \leftarrow Q(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x}))$$

Need to sweep over $\mathbf{x}$ and $\mathbf{u}$ to build Q

V, Q converge to V_π, Q_π





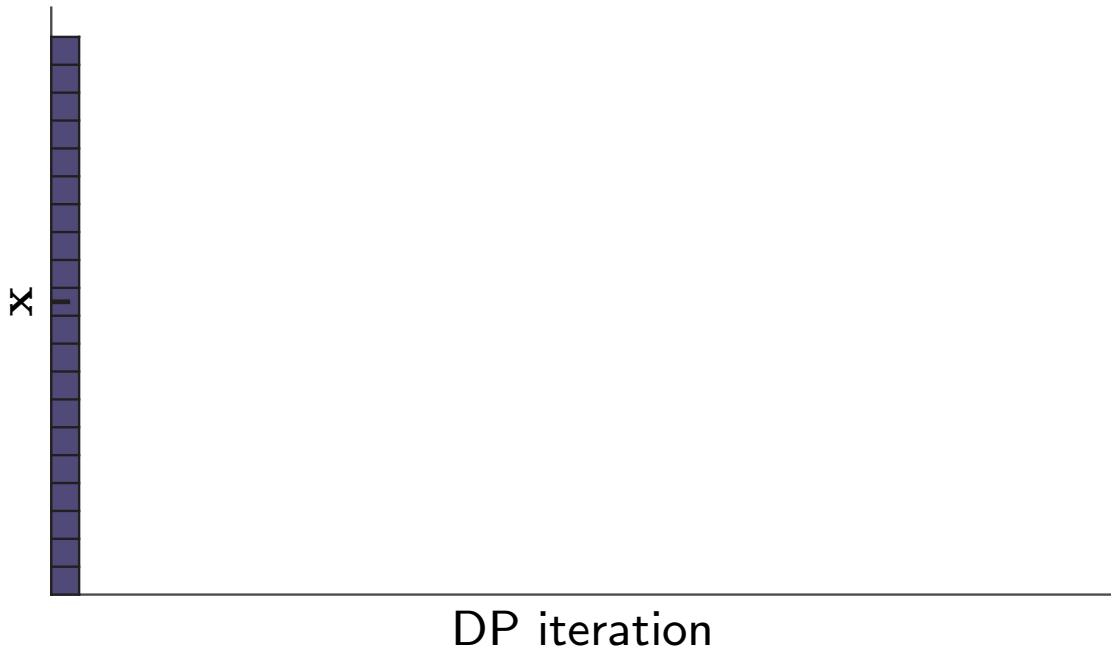
Computing Q_π - Illustration

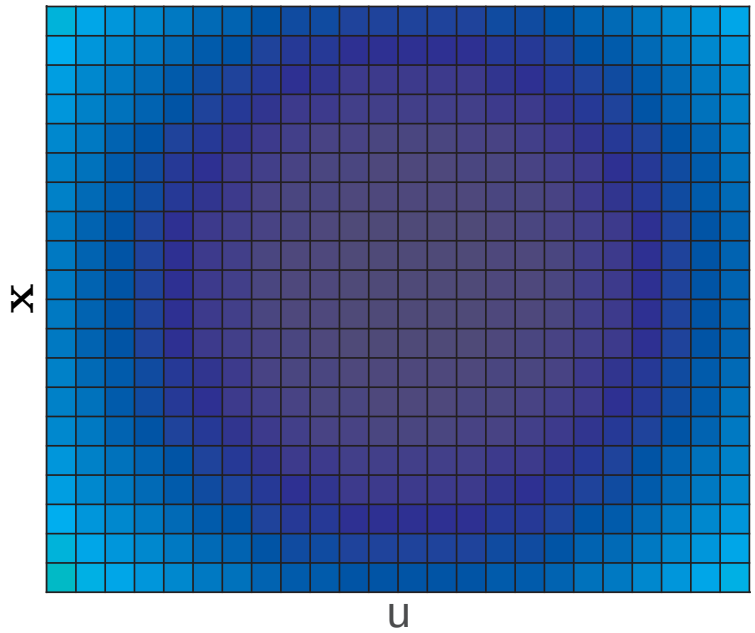
Can build V_π, Q_π jointly using:

$$Q(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V(\mathbf{x}_+) \,|\, \mathbf{x}, \mathbf{u}\right]$$
$$V(\mathbf{x}) \leftarrow Q(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x}))$$

Need to sweep over $\mathbf{x}$ and $\mathbf{u}$ to build Q

V, Q converge to V_π, Q_π





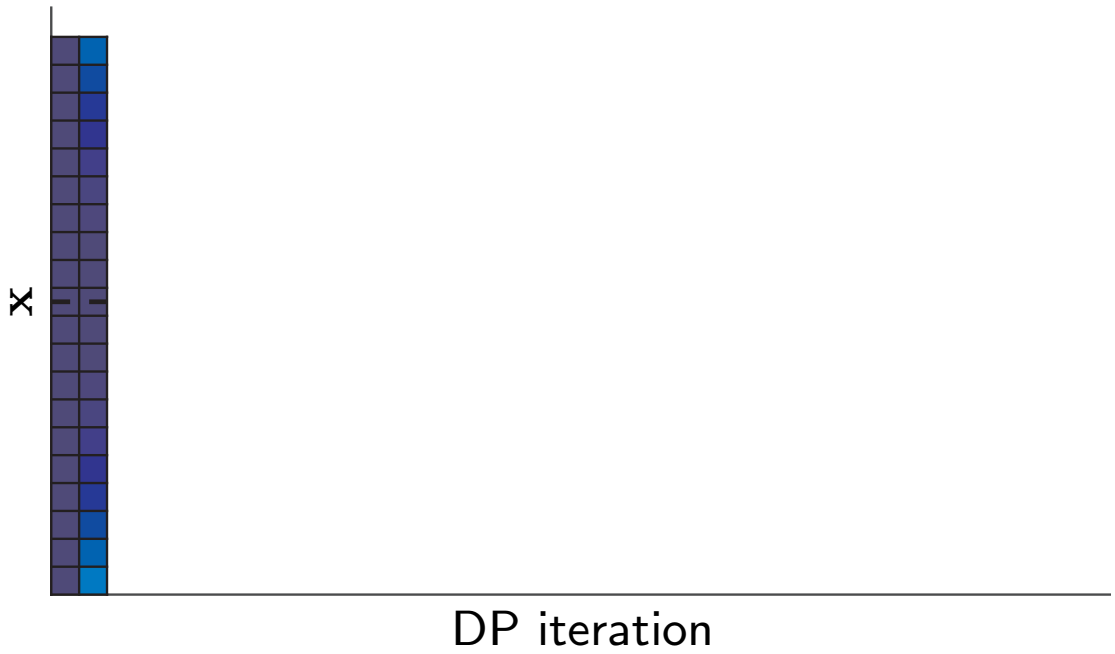
Computing Q_π - Illustration

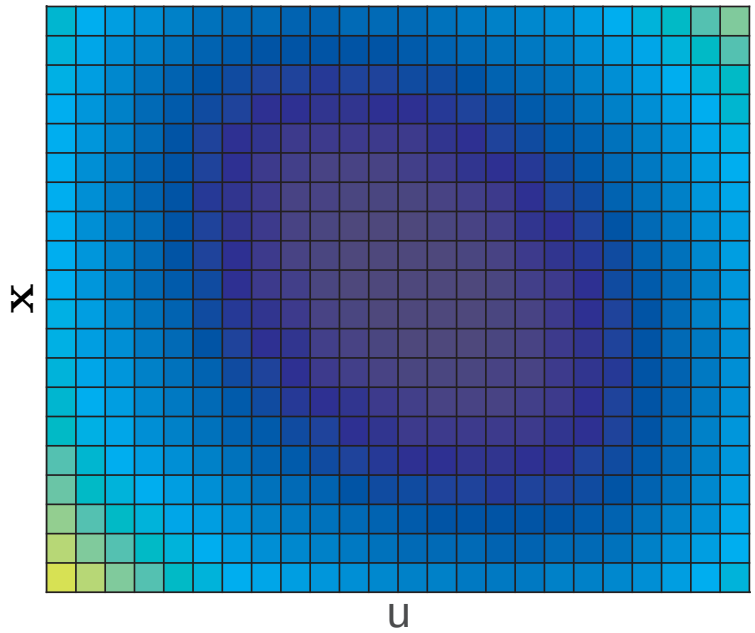
Can build V_π, Q_π jointly using:

$$Q(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V(\mathbf{x}_+) \,|\, \mathbf{x}, \mathbf{u}\right]$$
$$V(\mathbf{x}) \leftarrow Q(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x}))$$

Need to sweep over $\mathbf{x}$ and $\mathbf{u}$ to build Q

V, Q converge to V_π, Q_π





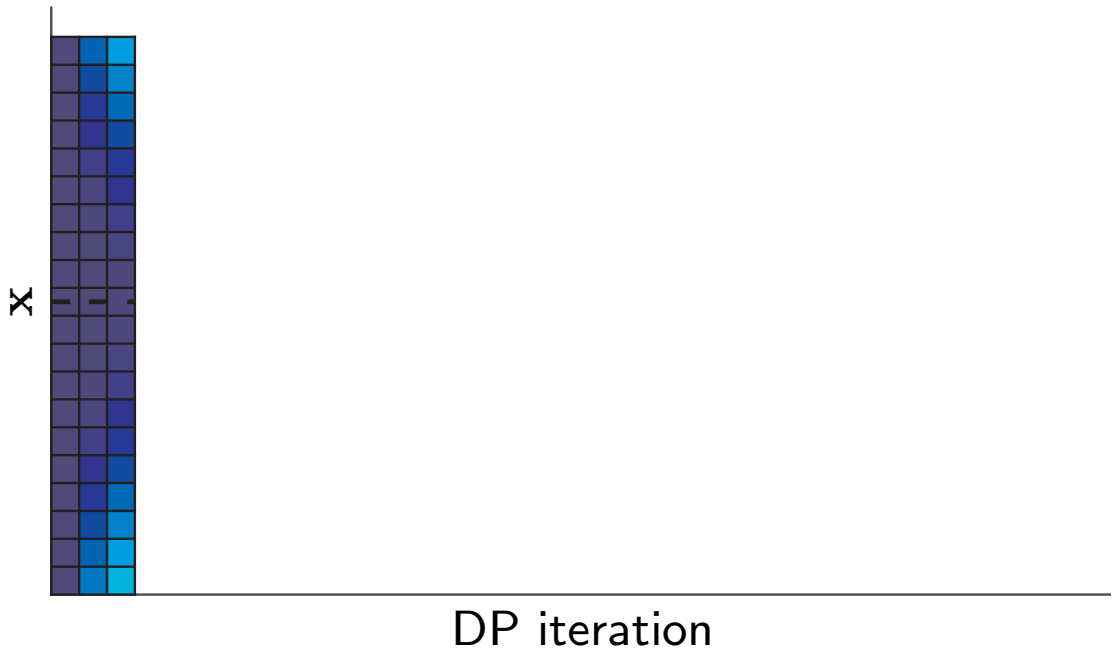
Computing Q_π - Illustration

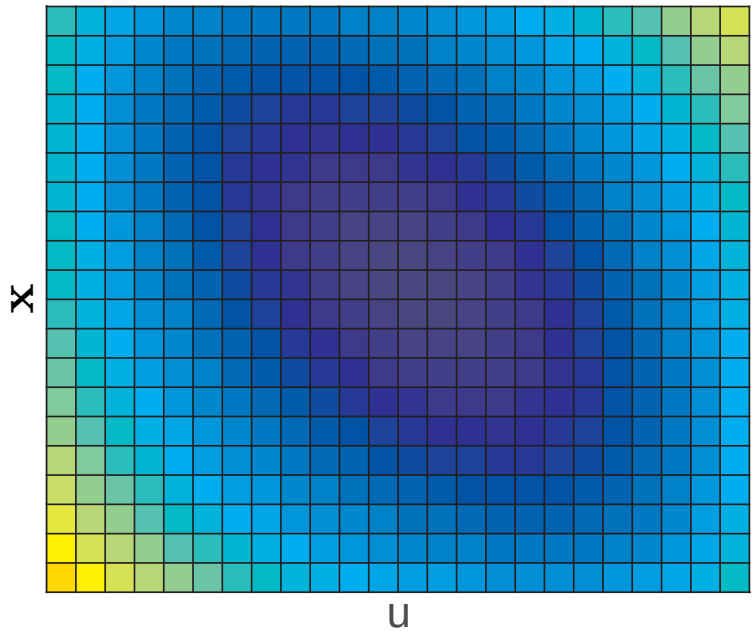
Can build V_π, Q_π jointly using:

$$Q(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V(\mathbf{x}_+) \,|\, \mathbf{x}, \mathbf{u}\right]$$
$$V(\mathbf{x}) \leftarrow Q(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x}))$$

Need to sweep over $\mathbf{x}$ and $\mathbf{u}$ to build Q

V, Q converge to V_π, Q_π





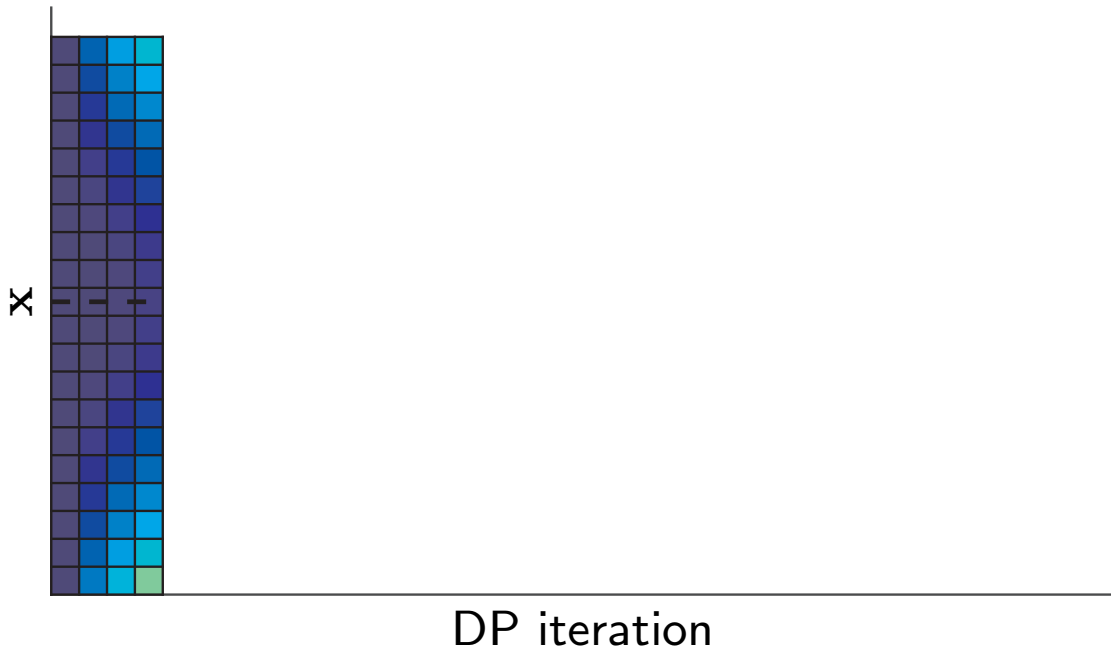
Computing Q_π - Illustration

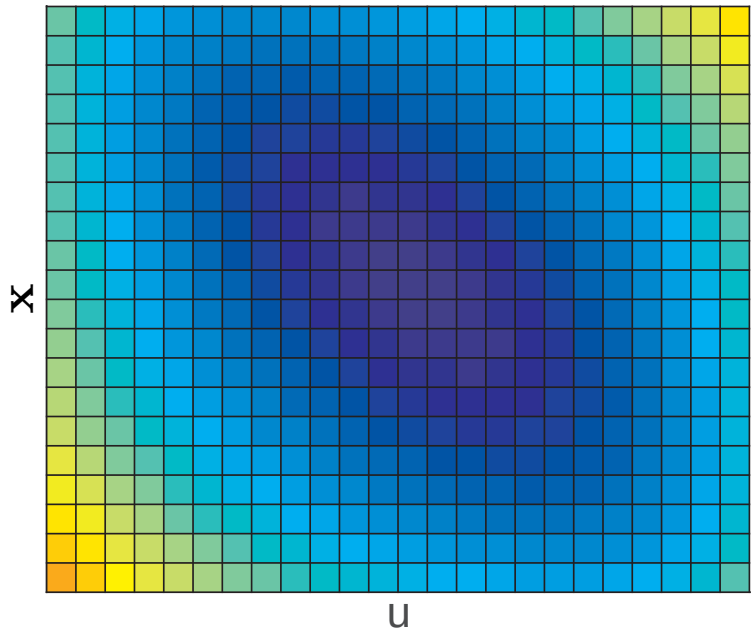
Can build V_π, Q_π jointly using:

$$Q(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V(\mathbf{x}_+) \,|\, \mathbf{x}, \mathbf{u}\right]$$
$$V(\mathbf{x}) \leftarrow Q(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x}))$$

Need to sweep over $\mathbf{x}$ and $\mathbf{u}$ to build Q

V, Q converge to V_π, Q_π





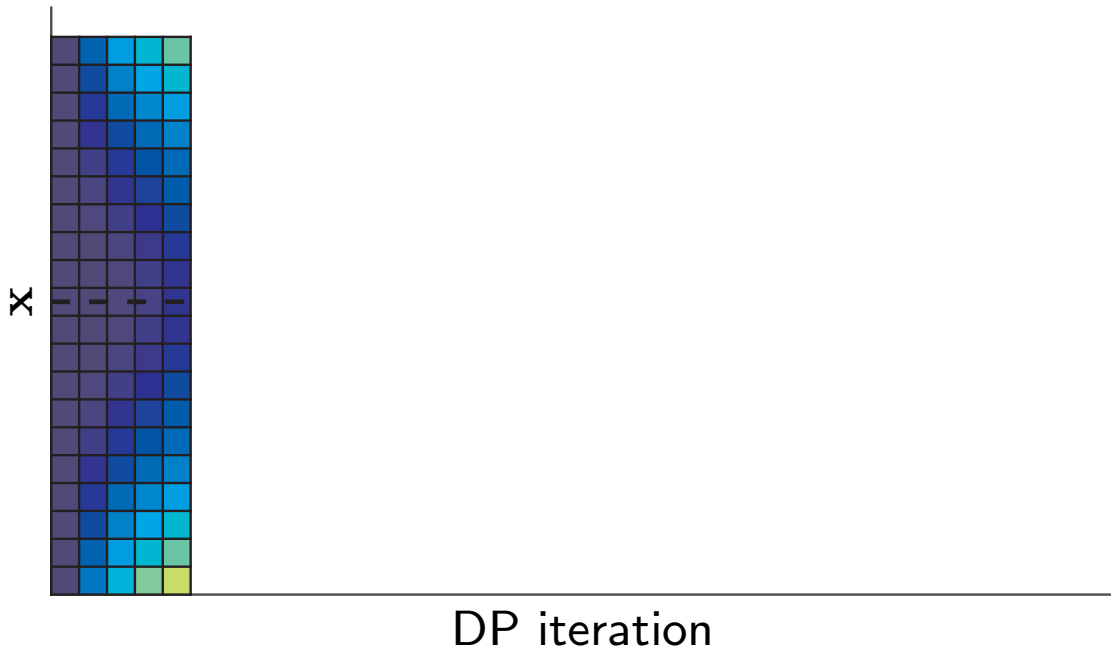
Computing Q_π - Illustration

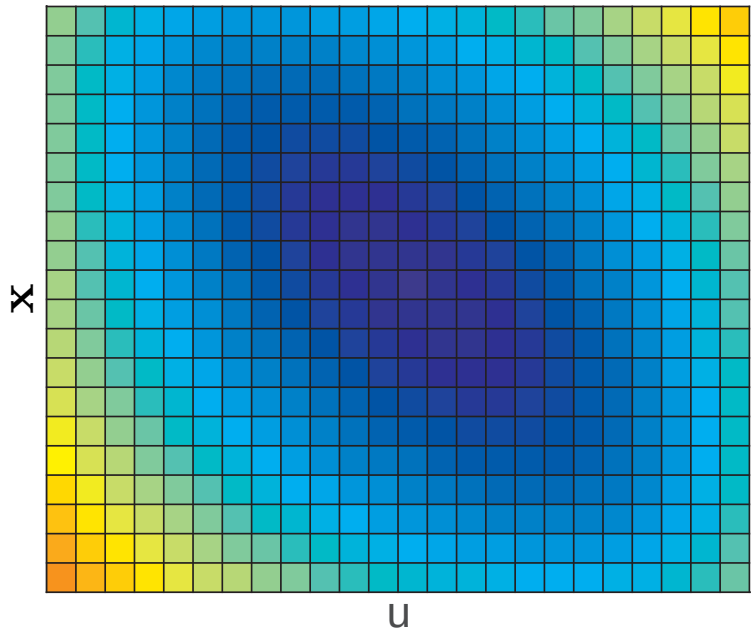
Can build V_π, Q_π jointly using:

$$Q(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V(\mathbf{x}_+) \,|\, \mathbf{x}, \mathbf{u}\right]$$
$$V(\mathbf{x}) \leftarrow Q(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x}))$$

Need to sweep over $\mathbf{x}$ and $\mathbf{u}$ to build Q

V, Q converge to V_π, Q_π





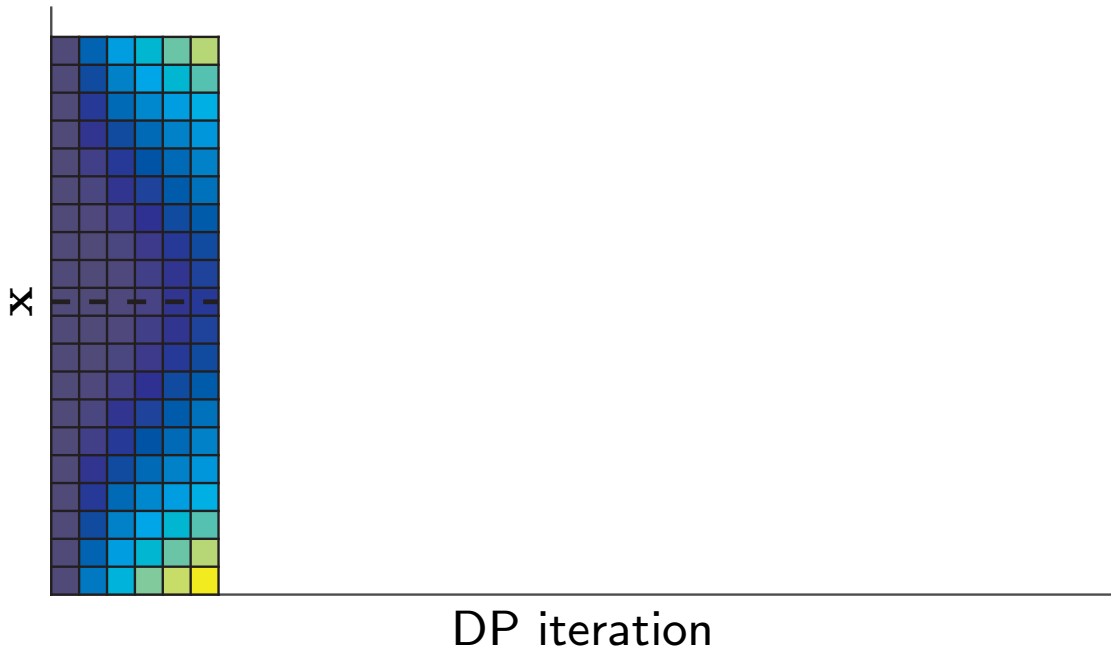
Computing Q_π - Illustration

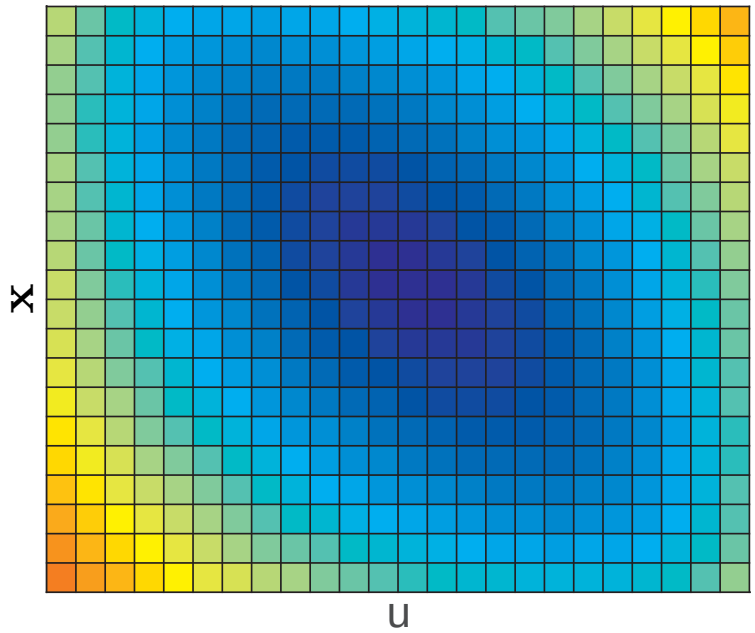
Can build V_π, Q_π jointly using:

$$Q(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V(\mathbf{x}_+) \,|\, \mathbf{x}, \mathbf{u}\right]$$
$$V(\mathbf{x}) \leftarrow Q(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x}))$$

Need to sweep over $\mathbf{x}$ and $\mathbf{u}$ to build Q

V, Q converge to V_π, Q_π





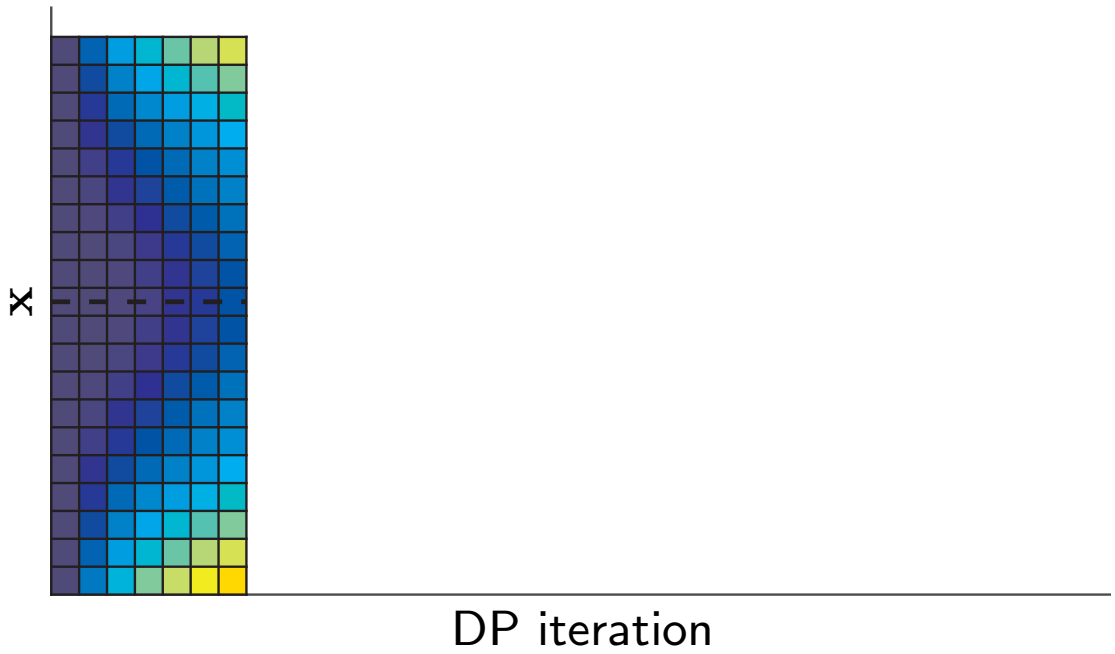
Computing Q_π - Illustration

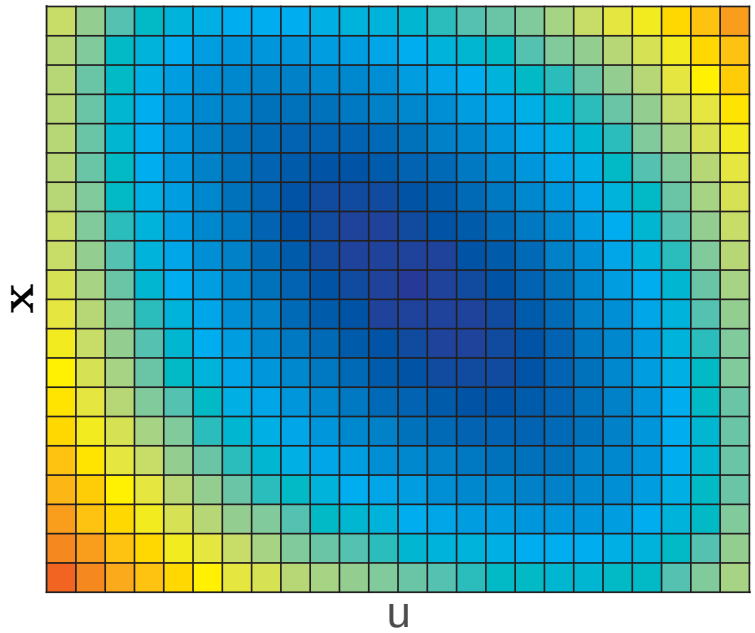
Can build V_π, Q_π jointly using:

$$Q(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V(\mathbf{x}_+) \,|\, \mathbf{x}, \mathbf{u}\right]$$
$$V(\mathbf{x}) \leftarrow Q(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x}))$$

Need to sweep over $\mathbf{x}$ and $\mathbf{u}$ to build Q

V, Q converge to V_π, Q_π





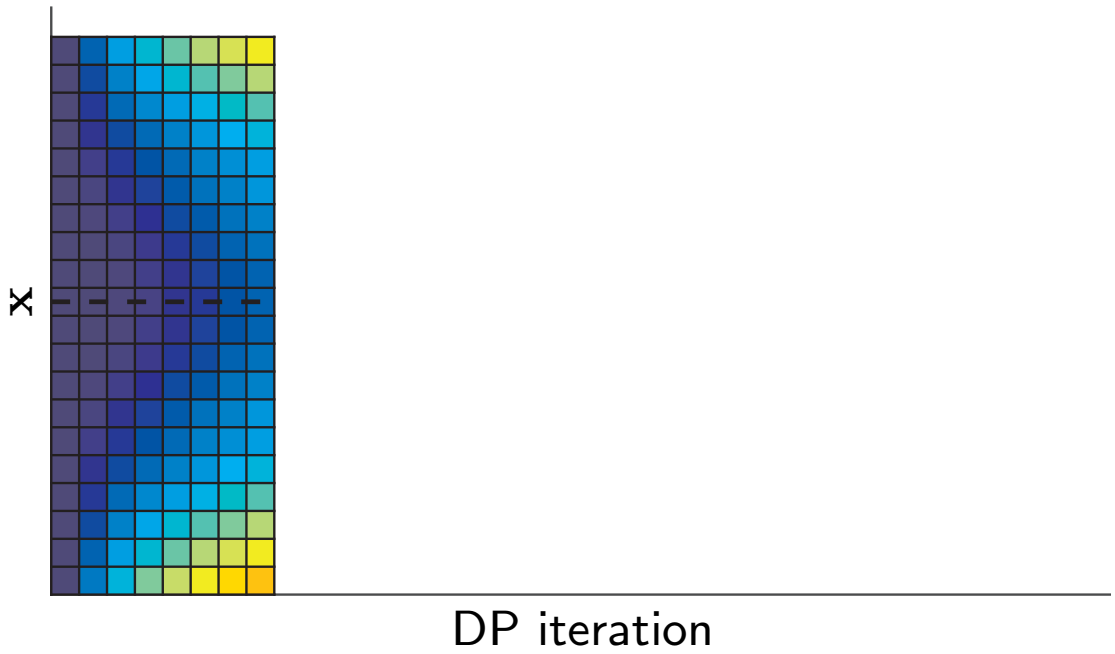
Computing Q_π - Illustration

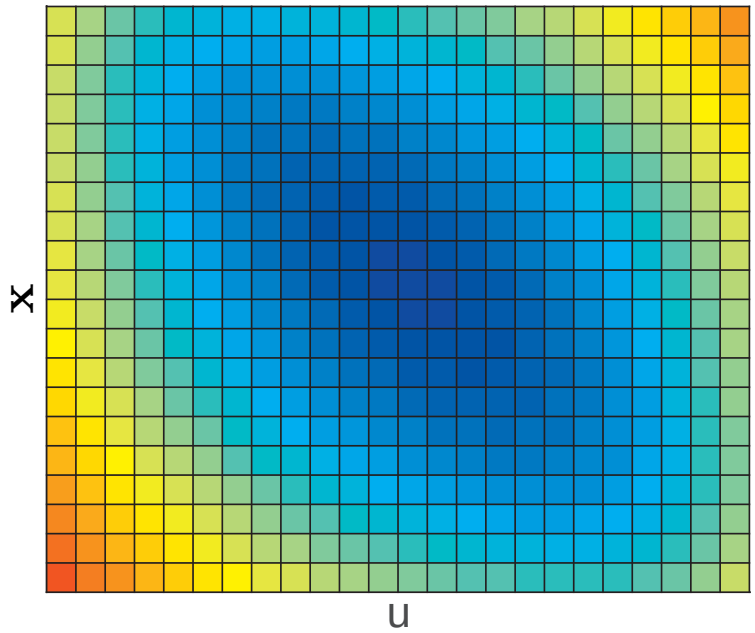
Can build V_π, Q_π jointly using:

$$Q(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V(\mathbf{x}_+) \,|\, \mathbf{x}, \mathbf{u}\right]$$
$$V(\mathbf{x}) \leftarrow Q(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x}))$$

Need to sweep over $\mathbf{x}$ and $\mathbf{u}$ to build Q

V, Q converge to V_π, Q_π





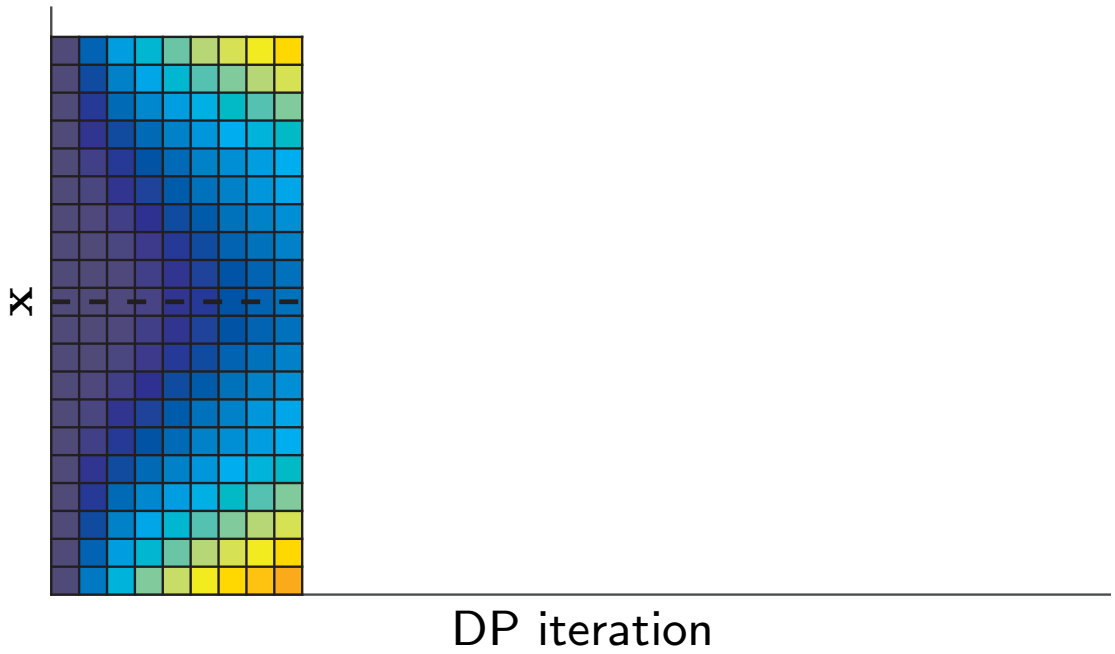
Computing Q_π - Illustration

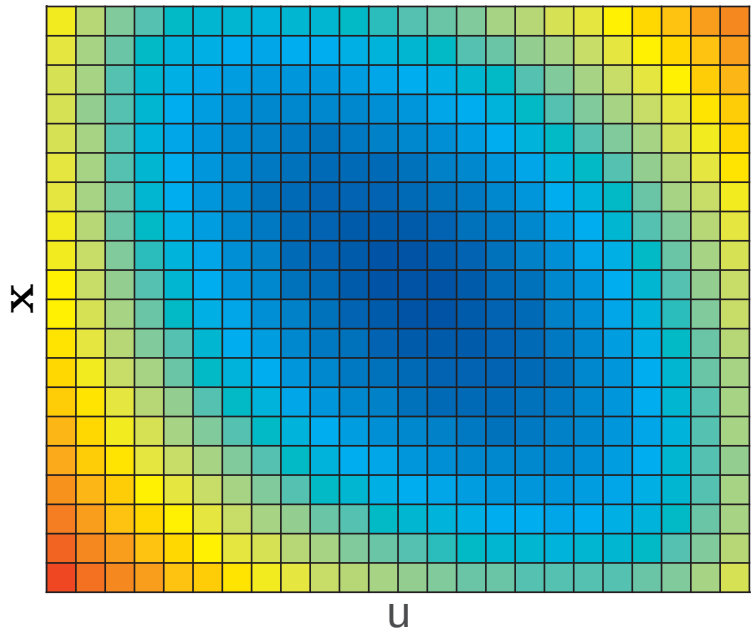
Can build V_π, Q_π jointly using:

$$Q(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V(\mathbf{x}_+) \,|\, \mathbf{x}, \mathbf{u}\right]$$
$$V(\mathbf{x}) \leftarrow Q(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x}))$$

Need to sweep over $\mathbf{x}$ and $\mathbf{u}$ to build Q

V, Q converge to V_π, Q_π





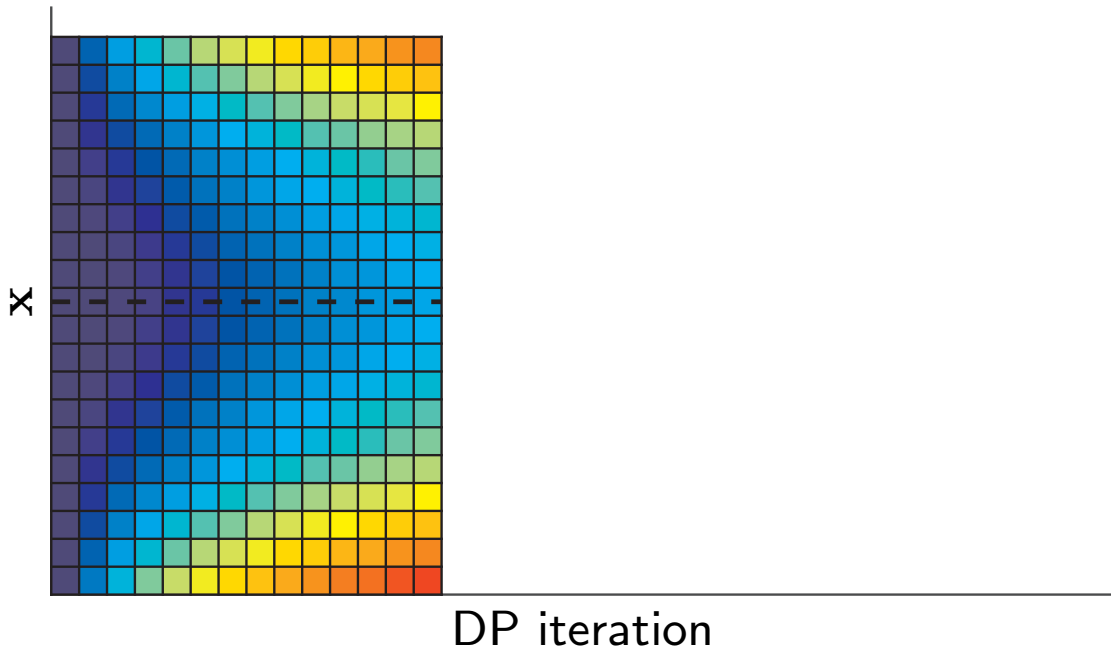
Computing Q_π - Illustration

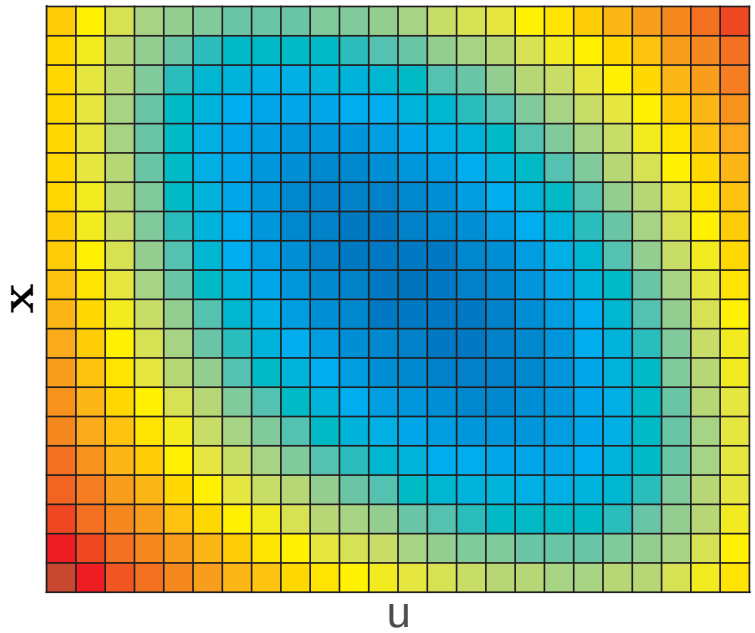
Can build V_π, Q_π jointly using:

$$Q(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V(\mathbf{x}_+) \,|\, \mathbf{x}, \mathbf{u}\right]$$
$$V(\mathbf{x}) \leftarrow Q(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x}))$$

Need to sweep over $\mathbf{x}$ and $\mathbf{u}$ to build Q

V, Q converge to V_π, Q_π





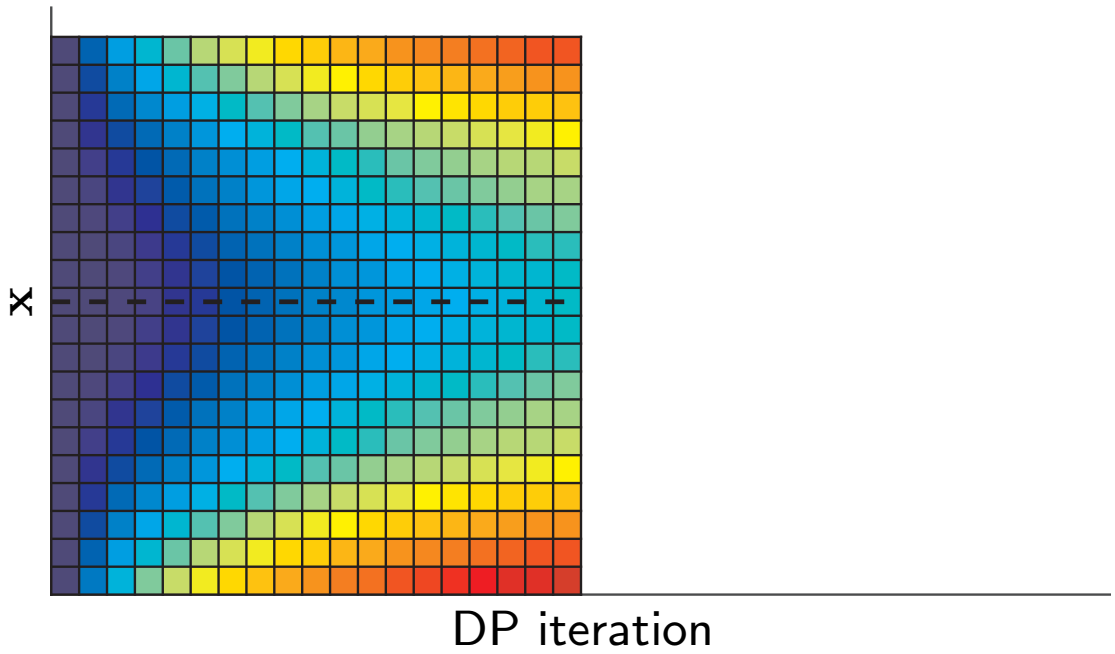
Computing Q_π - Illustration

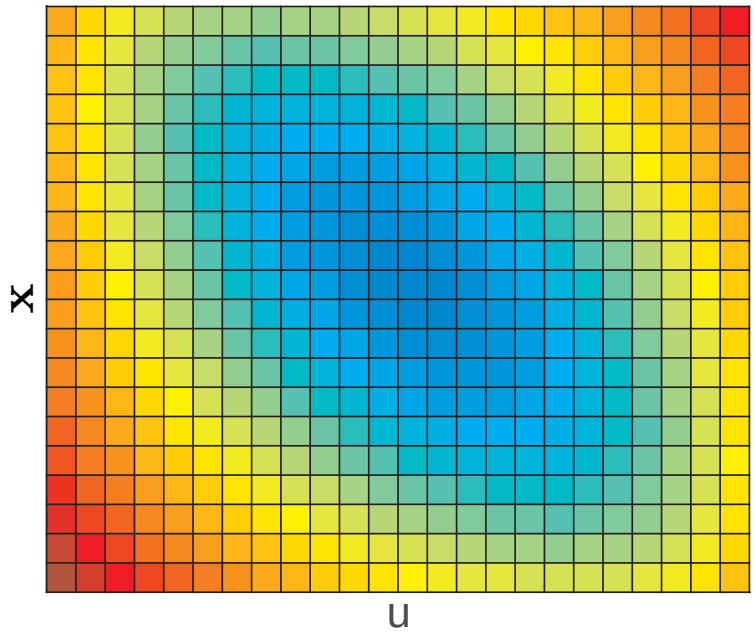
Can build V_π, Q_π jointly using:

$$Q(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V(\mathbf{x}_+) \,|\, \mathbf{x}, \mathbf{u}\right]$$
$$V(\mathbf{x}) \leftarrow Q(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x}))$$

Need to sweep over $\mathbf{x}$ and $\mathbf{u}$ to build Q

V, Q converge to V_π, Q_π





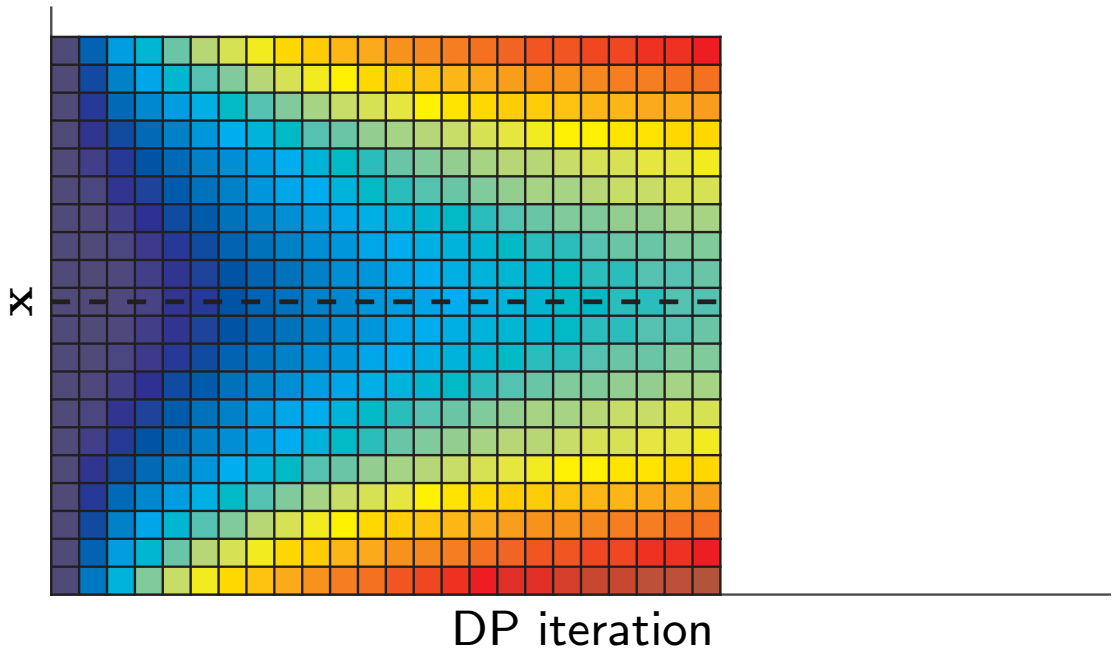
Computing Q_π - Illustration

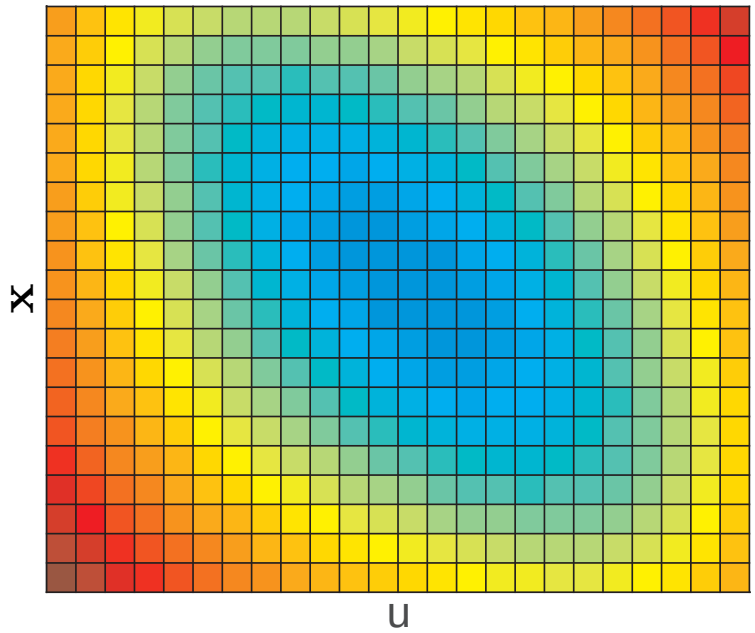
Can build V_π, Q_π jointly using:

$$Q(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V(\mathbf{x}_+) \,|\, \mathbf{x}, \mathbf{u}\right]$$
$$V(\mathbf{x}) \leftarrow Q(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x}))$$

Need to sweep over $\mathbf{x}$ and $\mathbf{u}$ to build Q

V, Q converge to V_π, Q_π





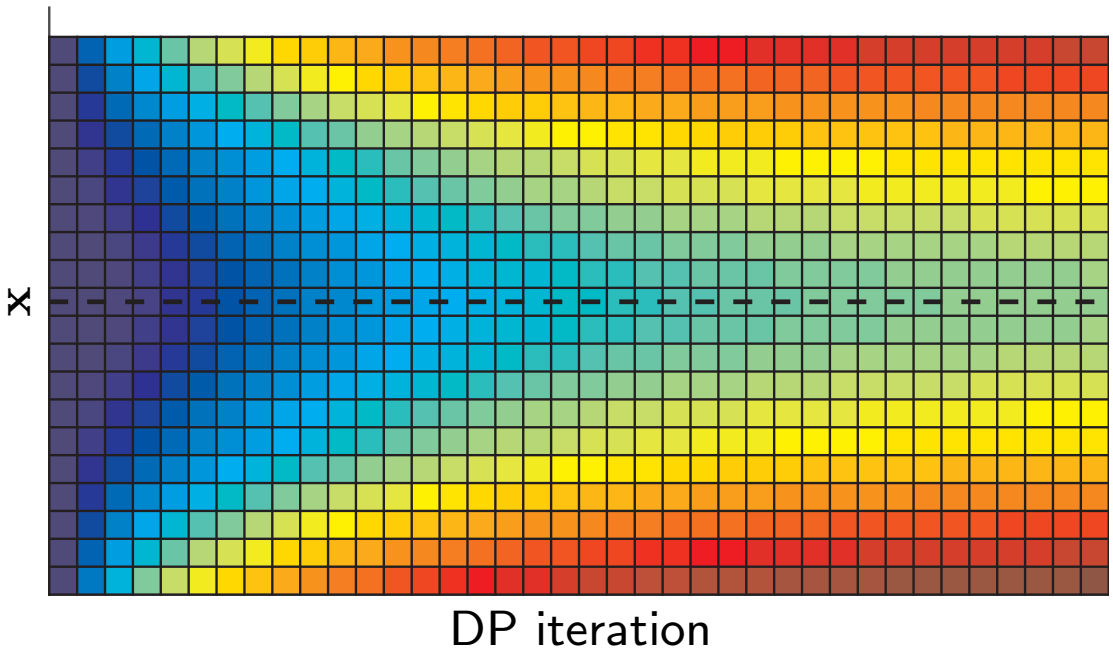
Computing Q_π - Illustration

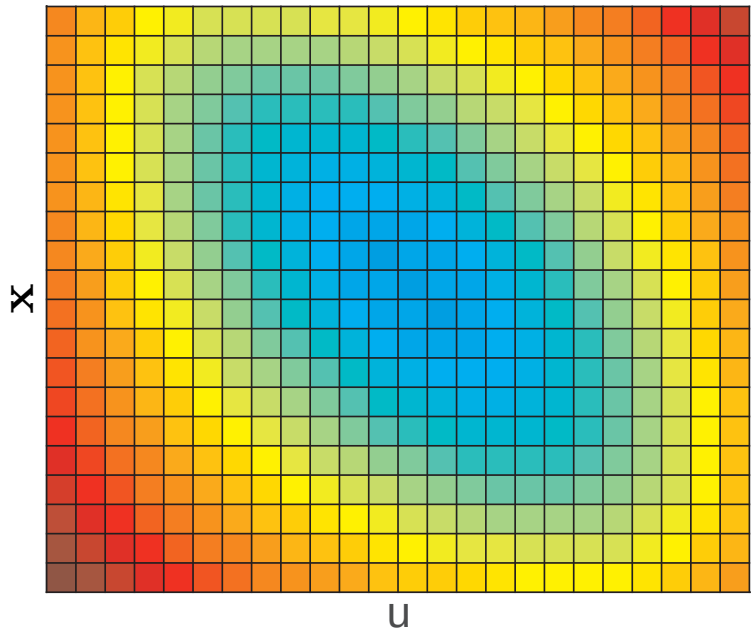
Can build V_π, Q_π jointly using:

$$Q(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V(\mathbf{x}_+) \,|\, \mathbf{x}, \mathbf{u}\right]$$
$$V(\mathbf{x}) \leftarrow Q(\mathbf{x}, \boldsymbol{\pi}(\mathbf{x}))$$

Need to sweep over $\mathbf{x}$ and $\mathbf{u}$ to build Q

V, Q converge to V_π, Q_π





How to compute optimal policies?

How to solve the Bellman optimality equations?

$$V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[V_\star(\mathbf{x}_+) \,|\, \mathbf{x},\mathbf{u}\right] \qquad V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; Q_\star(\mathbf{x},\mathbf{u}),$$

$$Q_\star(\mathbf{x},\mathbf{u}) = L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[\min_{\mathbf{u}'} \; Q_\star(\mathbf{x}_+,\mathbf{u}') \,|\, \mathbf{x},\mathbf{u}\right] \qquad \boldsymbol{\pi}_\star(\mathbf{x}) = \arg\min_{\mathbf{u}} \; Q_\star(\mathbf{x},\mathbf{u})$$

How to compute optimal policies?

How to solve the Bellman optimality equations?

$$V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[V_\star(\mathbf{x}_+) \,|\, \mathbf{x},\mathbf{u}\right] \qquad V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; Q_\star(\mathbf{x},\mathbf{u}),$$

$$Q_\star(\mathbf{x},\mathbf{u}) = L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[\min_{\mathbf{u}'} \; Q_\star(\mathbf{x}_+,\mathbf{u}') \,|\, \mathbf{x},\mathbf{u}\right] \qquad \boldsymbol{\pi}_\star(\mathbf{x}) = \arg\min_{\mathbf{u}} \; Q_\star(\mathbf{x},\mathbf{u})$$

Policy improvement: take any given policy $\boldsymbol{\pi}$ with associated $Q_{\boldsymbol{\pi}}$ and define:

$$\boldsymbol{\pi}'(\mathbf{x}) = \arg\min_{\mathbf{u}} \; Q_{\boldsymbol{\pi}}(\mathbf{x},\mathbf{u})$$

How to compute optimal policies?

How to solve the Bellman optimality equations?

$$V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[V_\star(\mathbf{x}_+) \,|\, \mathbf{x},\mathbf{u}\right] \qquad V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; Q_\star(\mathbf{x},\mathbf{u}),$$

$$Q_\star(\mathbf{x},\mathbf{u}) = L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[\min_{\mathbf{u}'} \; Q_\star(\mathbf{x}_+,\mathbf{u}') \,|\, \mathbf{x},\mathbf{u}\right] \qquad \boldsymbol{\pi}_\star(\mathbf{x}) = \arg\min_{\mathbf{u}} \; Q_\star(\mathbf{x},\mathbf{u})$$

Policy improvement: take any given policy $\boldsymbol{\pi}$ with associated $Q_{\boldsymbol{\pi}}$ and define:

$$\boldsymbol{\pi}'(\mathbf{x}) = \arg\min_{\mathbf{u}} \; Q_{\boldsymbol{\pi}}(\mathbf{x},\mathbf{u})$$

Then $V_{\boldsymbol{\pi}'}(\mathbf{x}) \underbrace{\leq}_{\text{not ovbious}} Q_{\boldsymbol{\pi}}(\mathbf{x},\boldsymbol{\pi}'(\mathbf{x})) \leq Q_{\boldsymbol{\pi}}(\mathbf{x},\boldsymbol{\pi}(\mathbf{x})) = V_{\boldsymbol{\pi}}(\mathbf{x}), \quad \forall \mathbf{x}$

How to compute optimal policies?

How to solve the Bellman optimality equations?

$$V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[V_\star(\mathbf{x}_+) \,|\, \mathbf{x},\mathbf{u}\right] \qquad V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; Q_\star(\mathbf{x},\mathbf{u}),$$

$$Q_\star(\mathbf{x},\mathbf{u}) = L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[\min_{\mathbf{u}'} \; Q_\star(\mathbf{x}_+,\mathbf{u}') \,|\, \mathbf{x},\mathbf{u}\right] \qquad \boldsymbol{\pi}_\star(\mathbf{x}) = \arg\min_{\mathbf{u}} \; Q_\star(\mathbf{x},\mathbf{u})$$

Policy improvement: take any given policy $\boldsymbol{\pi}$ with associated Q_π and define:

$$\boldsymbol{\pi}'(\mathbf{x}) = \arg\min_{\mathbf{u}} \; Q_\pi(\mathbf{x},\mathbf{u})$$

Then $V_{\pi'}(\mathbf{x}) \underbrace{\leq}_{\text{not ovbious}} Q_\pi(\mathbf{x},\boldsymbol{\pi}'(\mathbf{x})) \leq Q_\pi(\mathbf{x},\boldsymbol{\pi}(\mathbf{x})) = V_\pi(\mathbf{x}), \quad \forall \mathbf{x}$

π-improvement: select a $\boldsymbol{\pi}(\mathbf{x})$, alternate

1. Compute $Q_\pi(\mathbf{x})$
2. Update policy

$$\boldsymbol{\pi}(\mathbf{x}) \leftarrow \arg\min_{\mathbf{u}} \; Q_\pi(\mathbf{x},\mathbf{u})$$

How to compute optimal policies?

How to solve the Bellman optimality equations?

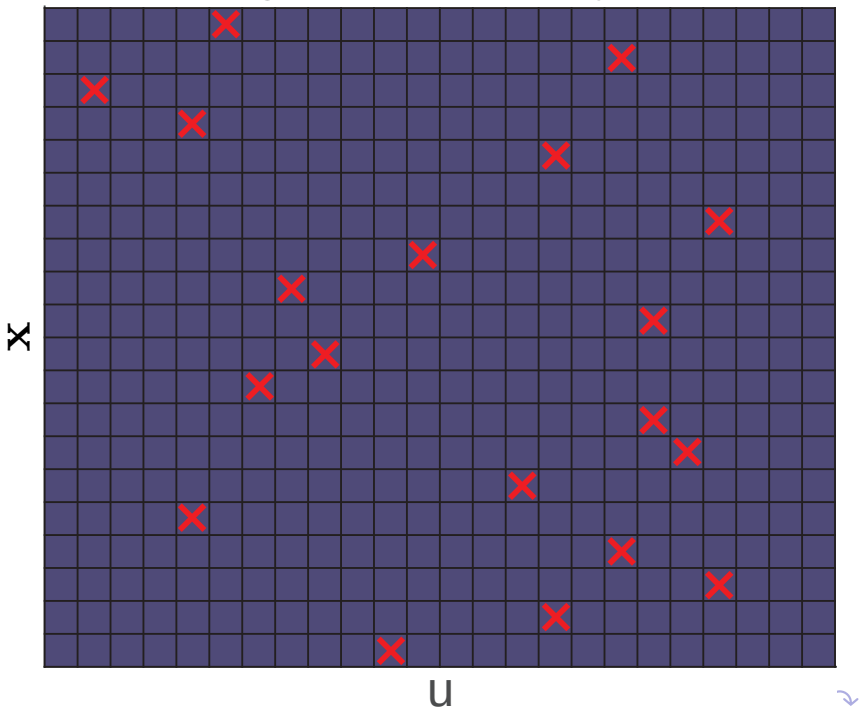
$$V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[V_\star(\mathbf{x}_+) \,|\, \mathbf{x},\mathbf{u}\right] \qquad V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; Q_\star(\mathbf{x},\mathbf{u}),$$

$$Q_\star(\mathbf{x},\mathbf{u}) = L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[\min_{\mathbf{u}'} \; Q_\star(\mathbf{x}_+,\mathbf{u}') \,|\, \mathbf{x},\mathbf{u}\right] \qquad \pi_\star(\mathbf{x}) = \arg\min_{\mathbf{u}} \; Q_\star(\mathbf{x},\mathbf{u})$$

Algorithm: π-improvement

Input: π and $\mathrm{tol} > 0$
$\mathrm{Iter} = \mathrm{True}$
while Iter **do**
  Compute Q_π
  Update $\pi_+(\mathbf{x}) \leftarrow \arg\min_{\mathbf{u}} \; Q_\pi(\mathbf{x},\mathbf{u})$
  if $\|\pi_+ - \pi\| < \mathrm{tol}$ **then**
    $\mathrm{Iter} = \mathrm{False}$
  $\pi_+ \leftarrow \pi$
return $\pi_\star \leftarrow \pi_+$

E.g. random policy π



How to compute optimal policies?

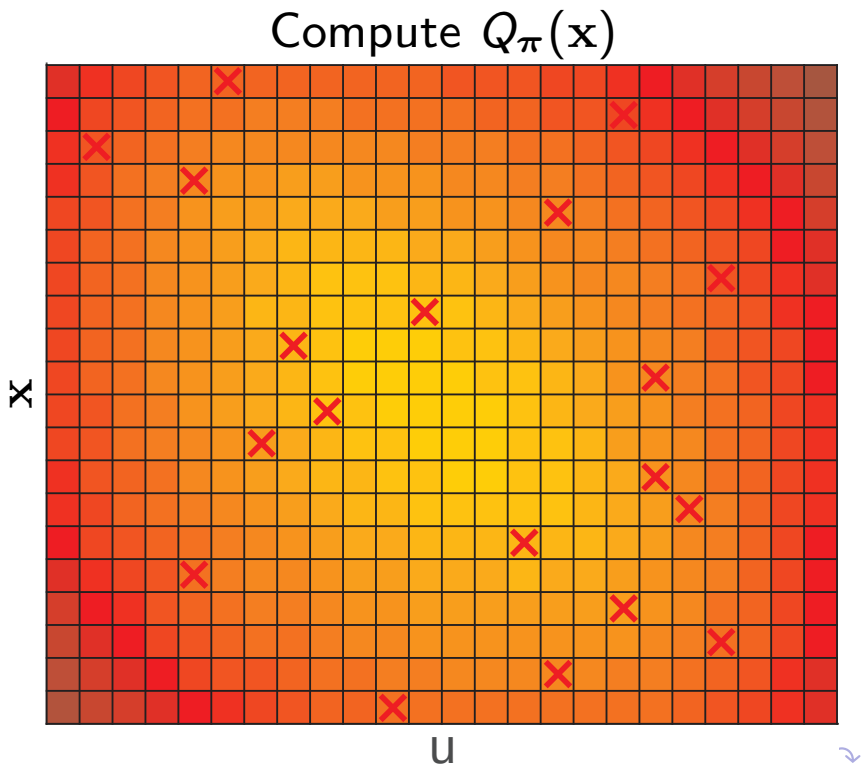
How to solve the Bellman optimality equations?

$$V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[V_\star(\mathbf{x}_+) \,|\, \mathbf{x},\mathbf{u}\right] \qquad V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; Q_\star(\mathbf{x},\mathbf{u}),$$

$$Q_\star(\mathbf{x},\mathbf{u}) = L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[\min_{\mathbf{u}'} \; Q_\star(\mathbf{x}_+,\mathbf{u}') \,|\, \mathbf{x},\mathbf{u}\right] \qquad \boldsymbol{\pi}_\star(\mathbf{x}) = \arg\min_{\mathbf{u}} \; Q_\star(\mathbf{x},\mathbf{u})$$

Algorithm: $\boldsymbol{\pi}$-improvement

Input: $\boldsymbol{\pi}$ and $\mathrm{tol} > 0$
$\mathrm{Iter} = \mathrm{True}$
while Iter **do**
  Compute $Q_{\boldsymbol{\pi}}$
  Update $\boldsymbol{\pi}_+(\mathbf{x}) \leftarrow \arg\min_{\mathbf{u}} \; Q_{\boldsymbol{\pi}}(\mathbf{x},\mathbf{u})$
  if $\|\boldsymbol{\pi}_+ - \boldsymbol{\pi}\| < \mathrm{tol}$ **then**
    $\mathrm{Iter} = \mathrm{False}$
  $\boldsymbol{\pi}_+ \leftarrow \boldsymbol{\pi}$
return $\boldsymbol{\pi}_\star \leftarrow \boldsymbol{\pi}_+$



How to compute optimal policies?

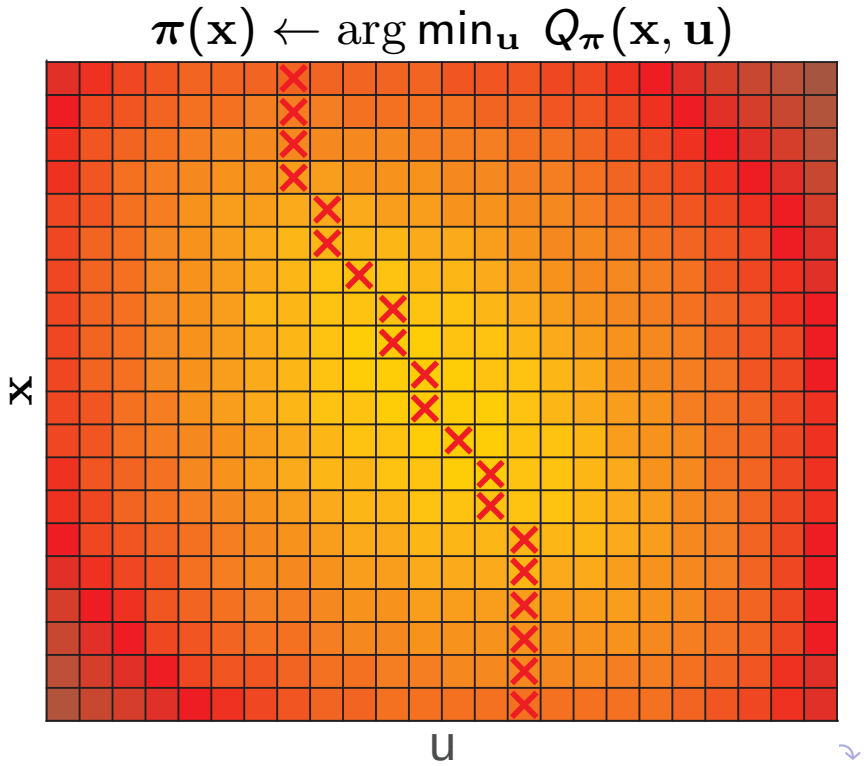
How to solve the Bellman optimality equations?

$$V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[V_\star(\mathbf{x}_+) \,|\, \mathbf{x},\mathbf{u}\right] \qquad V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; Q_\star(\mathbf{x},\mathbf{u}),$$

$$Q_\star(\mathbf{x},\mathbf{u}) = L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[\min_{\mathbf{u}'} \; Q_\star(\mathbf{x}_+,\mathbf{u}') \,|\, \mathbf{x},\mathbf{u}\right] \qquad \boldsymbol{\pi}_\star(\mathbf{x}) = \arg\min_{\mathbf{u}} \; Q_\star(\mathbf{x},\mathbf{u})$$

Algorithm: $\boldsymbol{\pi}$-improvement

Input: $\boldsymbol{\pi}$ and $\mathrm{tol} > 0$
$\mathrm{Iter} = \mathrm{True}$
while Iter **do**
  Compute $Q_{\boldsymbol{\pi}}$
  Update $\boldsymbol{\pi}_+(\mathbf{x}) \leftarrow \arg\min_{\mathbf{u}} \; Q_{\boldsymbol{\pi}}(\mathbf{x},\mathbf{u})$
  if $\|\boldsymbol{\pi}_+ - \boldsymbol{\pi}\| < \mathrm{tol}$ **then**
    $\mathrm{Iter} = \mathrm{False}$
  $\boldsymbol{\pi}_+ \leftarrow \boldsymbol{\pi}$
return $\boldsymbol{\pi}_\star \leftarrow \boldsymbol{\pi}_+$



How to compute optimal policies?

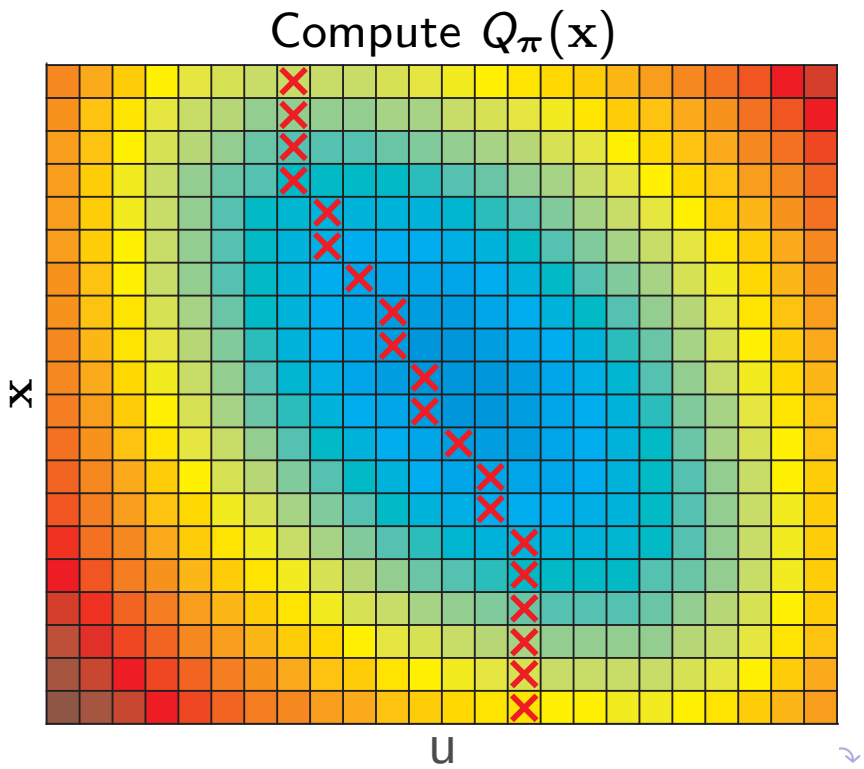
How to solve the Bellman optimality equations?

$$V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[V_\star(\mathbf{x}_+) \,|\, \mathbf{x},\mathbf{u}\right] \qquad V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; Q_\star(\mathbf{x},\mathbf{u}),$$

$$Q_\star(\mathbf{x},\mathbf{u}) = L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[\min_{\mathbf{u}'} \; Q_\star(\mathbf{x}_+,\mathbf{u}') \,|\, \mathbf{x},\mathbf{u}\right] \qquad \boldsymbol{\pi}_\star(\mathbf{x}) = \arg\min_{\mathbf{u}} \; Q_\star(\mathbf{x},\mathbf{u})$$

Algorithm: $\boldsymbol{\pi}$-improvement

Input: $\boldsymbol{\pi}$ and $\mathrm{tol} > 0$
$\mathrm{Iter} = \mathrm{True}$
while Iter **do**
  Compute $Q_{\boldsymbol{\pi}}$
  Update $\boldsymbol{\pi}_+(\mathbf{x}) \leftarrow \arg\min_{\mathbf{u}} \; Q_{\boldsymbol{\pi}}(\mathbf{x},\mathbf{u})$
  if $\|\boldsymbol{\pi}_+ - \boldsymbol{\pi}\| < \mathrm{tol}$ **then**
    $\mathrm{Iter} = \mathrm{False}$
  $\boldsymbol{\pi}_+ \leftarrow \boldsymbol{\pi}$
return $\boldsymbol{\pi}_\star \leftarrow \boldsymbol{\pi}_+$



How to compute optimal policies?

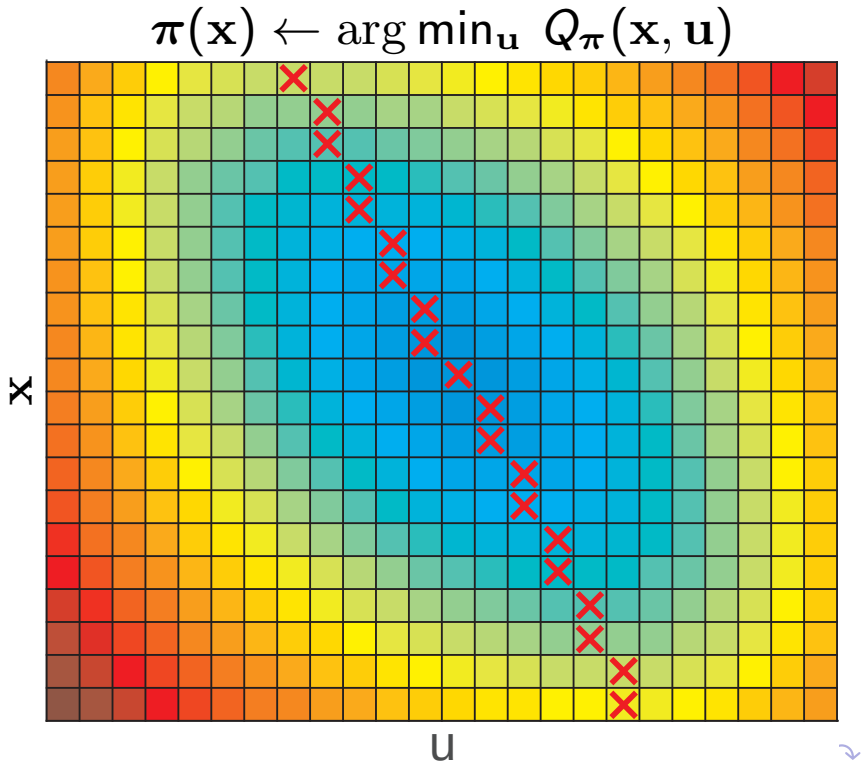
How to solve the Bellman optimality equations?

$$V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[V_\star(\mathbf{x}_+) \,|\, \mathbf{x},\mathbf{u}\right] \qquad V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; Q_\star(\mathbf{x},\mathbf{u}),$$

$$Q_\star(\mathbf{x},\mathbf{u}) = L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[\min_{\mathbf{u}'} \; Q_\star(\mathbf{x}_+,\mathbf{u}') \,|\, \mathbf{x},\mathbf{u}\right] \qquad \boldsymbol{\pi}_\star(\mathbf{x}) = \arg\min_{\mathbf{u}} \; Q_\star(\mathbf{x},\mathbf{u})$$

Algorithm: $\boldsymbol{\pi}$-improvement

Input: $\boldsymbol{\pi}$ and $\mathrm{tol} > 0$
$\mathrm{Iter} = \mathrm{True}$
while Iter **do**
  Compute $Q_{\boldsymbol{\pi}}$
  Update $\boldsymbol{\pi}_+(\mathbf{x}) \leftarrow \arg\min_{\mathbf{u}} \; Q_{\boldsymbol{\pi}}(\mathbf{x},\mathbf{u})$
  if $\|\boldsymbol{\pi}_+ - \boldsymbol{\pi}\| < \mathrm{tol}$ **then**
    $\mathrm{Iter} = \mathrm{False}$
  $\boldsymbol{\pi}_+ \leftarrow \boldsymbol{\pi}$
return $\boldsymbol{\pi}_\star \leftarrow \boldsymbol{\pi}_+$



How to compute optimal policies?

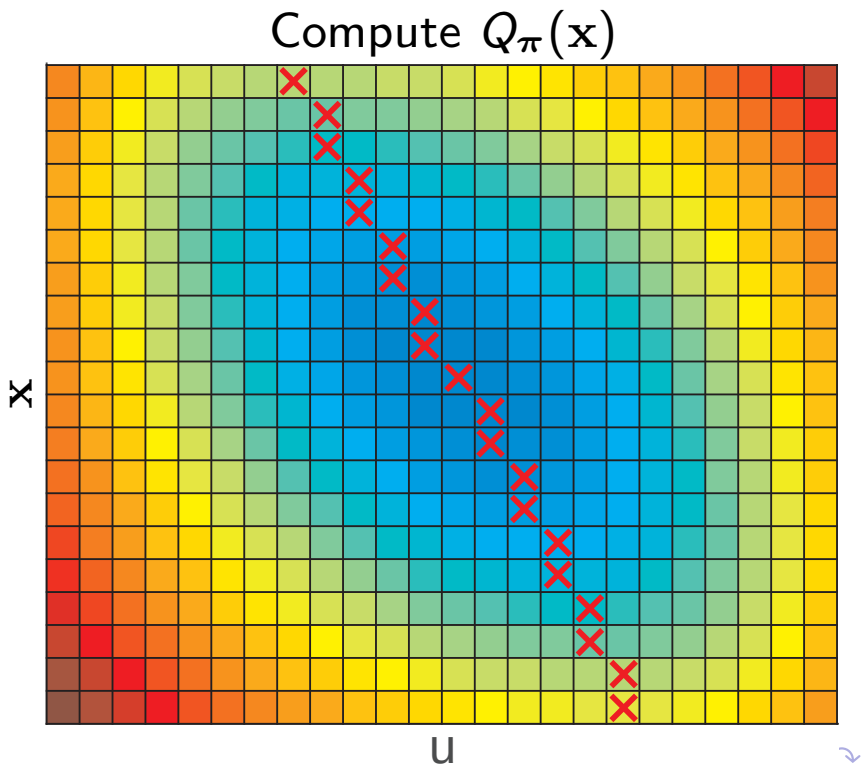
How to solve the Bellman optimality equations?

$$V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[V_\star(\mathbf{x}_+) \,|\, \mathbf{x},\mathbf{u}\right] \qquad V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; Q_\star(\mathbf{x},\mathbf{u}),$$

$$Q_\star(\mathbf{x},\mathbf{u}) = L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[\min_{\mathbf{u}'} \; Q_\star(\mathbf{x}_+,\mathbf{u}') \,|\, \mathbf{x},\mathbf{u}\right] \qquad \boldsymbol{\pi}_\star(\mathbf{x}) = \arg\min_{\mathbf{u}} \; Q_\star(\mathbf{x},\mathbf{u})$$

Algorithm: $\boldsymbol{\pi}$-improvement

Input: $\boldsymbol{\pi}$ and $\mathrm{tol} > 0$
$\mathrm{Iter} = \mathrm{True}$
while Iter **do**
  Compute $Q_{\boldsymbol{\pi}}$
  Update $\boldsymbol{\pi}_+(\mathbf{x}) \leftarrow \arg\min_{\mathbf{u}} \; Q_{\boldsymbol{\pi}}(\mathbf{x},\mathbf{u})$
  if $\|\boldsymbol{\pi}_+ - \boldsymbol{\pi}\| < \mathrm{tol}$ **then**
    $\mathrm{Iter} = \mathrm{False}$
  $\boldsymbol{\pi}_+ \leftarrow \boldsymbol{\pi}$
return $\boldsymbol{\pi}_\star \leftarrow \boldsymbol{\pi}_+$



How to compute optimal policies?

How to solve the Bellman optimality equations?

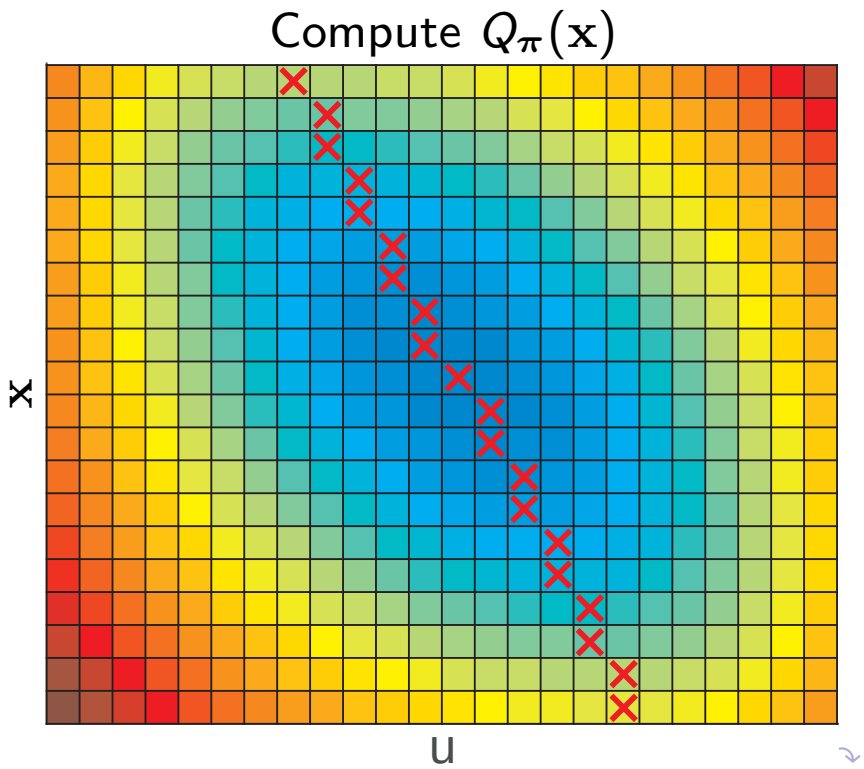
$$V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[V_\star(\mathbf{x}_+) \,|\, \mathbf{x},\mathbf{u}\right] \qquad V_\star(\mathbf{x}) = \min_{\mathbf{u}} \; Q_\star(\mathbf{x},\mathbf{u}),$$

$$Q_\star(\mathbf{x},\mathbf{u}) = L(\mathbf{x},\mathbf{u}) + \gamma \mathbb{E}\left[\min_{\mathbf{u}'} \; Q_\star(\mathbf{x}_+,\mathbf{u}') \,|\, \mathbf{x},\mathbf{u}\right] \qquad \boldsymbol{\pi}_\star(\mathbf{x}) = \arg\min_{\mathbf{u}} \; Q_\star(\mathbf{x},\mathbf{u})$$

Algorithm: $\boldsymbol{\pi}$-improvement

Input: $\boldsymbol{\pi}$ and $\mathrm{tol} > 0$
$\mathrm{Iter} = \mathrm{True}$
while Iter **do**
  Compute $Q_{\boldsymbol{\pi}}$
  Update $\boldsymbol{\pi}_+(\mathbf{x}) \leftarrow \arg\min_{\mathbf{u}} \; Q_{\boldsymbol{\pi}}(\mathbf{x},\mathbf{u})$
  if $\|\boldsymbol{\pi}_+ - \boldsymbol{\pi}\| < \mathrm{tol}$ **then**
    $\mathrm{Iter} = \mathrm{False}$
  $\boldsymbol{\pi}_+ \leftarrow \boldsymbol{\pi}$
return $\boldsymbol{\pi}_\star \leftarrow \boldsymbol{\pi}_+$

Converges to $\boldsymbol{\pi}_\star$ if initial policy yields $Q_{\boldsymbol{\pi}} < \infty$ for some $\mathbf{x}, \mathbf{u}$



Dynamic Programming

Algorithm: π-improvement

Input: π and $\text{tol} > 0$

$\text{Iter} = \text{True}$

while Iter **do**

  Compute Q_π

  Update

  $\pi_+(\mathbf{x}) \leftarrow \arg\min_{\mathbf{u}} \; Q_\pi(\mathbf{x}, \mathbf{u})$

  if $\|\pi_+ - \pi\| < \text{tol}$ **then**

    $\text{Iter} = \text{False}$

  $\pi_+ \leftarrow \pi$

return $\pi_\star \leftarrow \pi_+$

Dynamic Programming

Algorithm: π-improvement

Input: π and $\text{tol} > 0$
$\text{Iter} = \text{True}$
while Iter **do**
  Compute Q_π
  Update
  $\pi_+(\mathbf{x}) \leftarrow \arg\min_{\mathbf{u}} \ Q_\pi(\mathbf{x}, \mathbf{u})$
  if $\|\pi_+ - \pi\| < \text{tol}$ **then**
    $\text{Iter} = \text{False}$
  $\pi_+ \leftarrow \pi$
return $\pi_\star \leftarrow \pi_+$

Algorithm: Value iteration

Input: V and $\text{tol} > 0$
$\text{Iter} = \text{True}, \ Q = 0$
while Iter **do**
  Compute
  $Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$
  $V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$
  if $\|Q_+ - Q\| < \text{tol}$ **then**
    $\text{Iter} = \text{False}$
  $Q \leftarrow Q_+$
return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$

Dynamic Programming

Algorithm: π-improvement

Input: π and $\text{tol} > 0$
$\text{Iter} = \text{True}$
while Iter **do**
  Compute Q_π
  Update
  $\pi_+(\mathbf{x}) \leftarrow \arg\min_{\mathbf{u}} \; Q_\pi(\mathbf{x}, \mathbf{u})$
  if $\|\pi_+ - \pi\| < \text{tol}$ **then**
    $\text{Iter} = \text{False}$
  $\pi_+ \leftarrow \pi$
return $\pi_\star \leftarrow \pi_+$

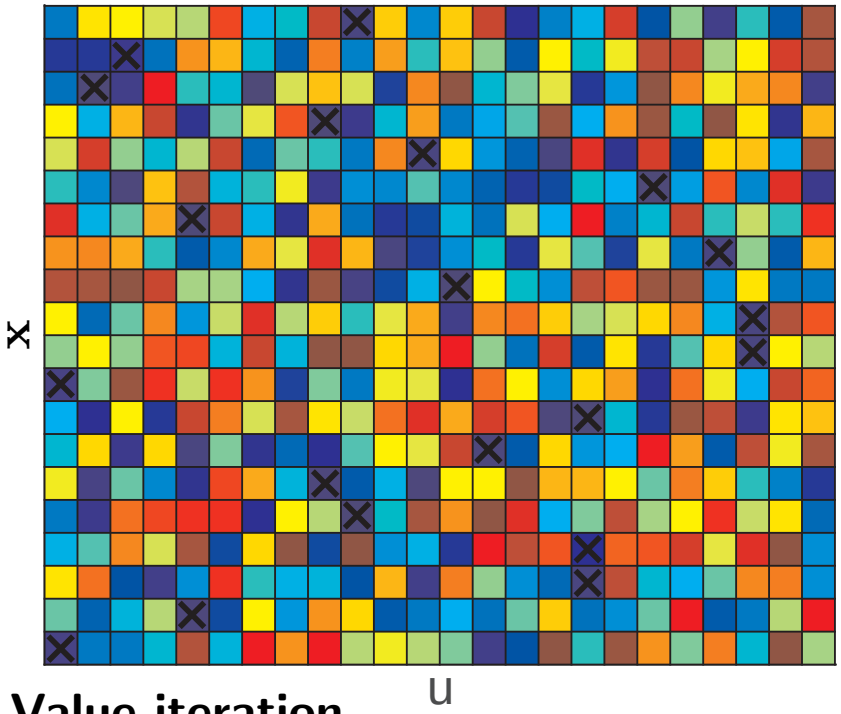
Algorithm: Value iteration

Input: V and $\text{tol} > 0$
$\text{Iter} = \text{True}, \; Q = 0$
while Iter **do**
  Compute
  $$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$
  $$V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \; Q_+(\mathbf{x}, \mathbf{u})$$
  if $\|Q_+ - Q\| < \text{tol}$ **then**
    $\text{Iter} = \text{False}$
  $Q \leftarrow Q_+$
return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \; Q_+(\mathbf{x}, \mathbf{u})$

Value iteration

- Iteration often written as $V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \; L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$
- No need to compute Q_π (saves a lot of computation)
- Before convergence $V(\mathbf{x})$ may not correspond to any policy
- For $\gamma < 1$, converges to $V_\star$, $Q_\star$, $\pi_\star$ for any starting V (finite)
- For $\gamma = 1$, convergence requires extra assumptions (may depend on starting V)

Dynamic Programming



Value iteration

Algorithm: Value iteration

Input: V and $\text{tol} > 0$

$\text{Iter} = \text{True}, \ Q = 0$

while Iter **do**

  Compute

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$

$$V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$$

  if $\|Q_+ - Q\| < \text{tol}$ **then**

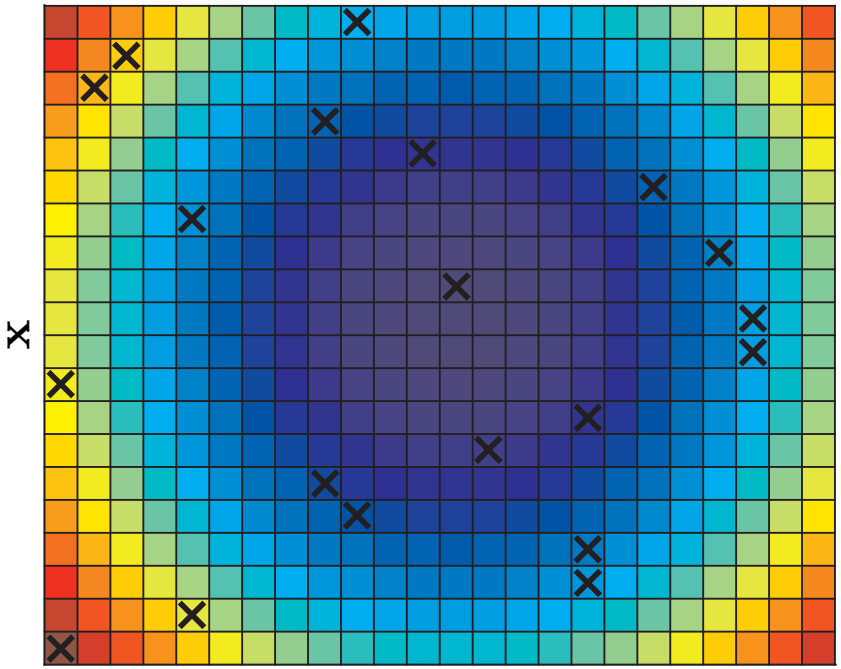
    $\text{Iter} = \text{False}$

  $Q \leftarrow Q_+$

return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$

- Iteration often written as $V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$
- No need to compute Q_π (saves a lot of computation)
- Before convergence $V(\mathbf{x})$ may not correspond to any policy
- For $\gamma < 1$, converges to $V_\star$, $Q_\star$, $\pi_\star$ for any starting V (finite)
- For $\gamma = 1$, convergence requires extra assumptions (may depend on starting V)

Dynamic Programming



Value iteration

Algorithm: Value iteration

Input: V and $\text{tol} > 0$

$\text{Iter} = \text{True}, \ Q = 0$

while Iter **do**

  Compute

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$

$$V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$$

  if $\|Q_+ - Q\| < \text{tol}$ **then**

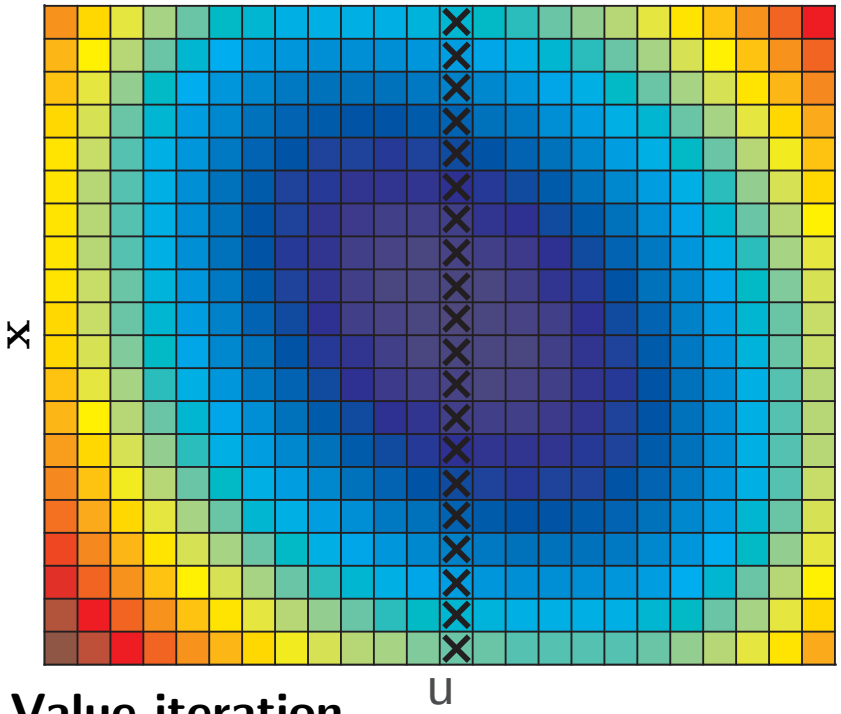
    $\text{Iter} = \text{False}$

  $Q \leftarrow Q_+$

return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$

- Iteration often written as $V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$
- No need to compute Q_π (saves a lot of computation)
- Before convergence $V(\mathbf{x})$ may not correspond to any policy
- For $\gamma < 1$, converges to $V_\star$, $Q_\star$, $\pi_\star$ for any starting V (finite)
- For $\gamma = 1$, convergence requires extra assumptions (may depend on starting V)

Dynamic Programming



Value iteration

Algorithm: Value iteration

Input: V and $\text{tol} > 0$

$\text{Iter} = \text{True}, \ Q = 0$

while Iter **do**

  Compute

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$

$$V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$$

  if $\|Q_+ - Q\| < \text{tol}$ **then**

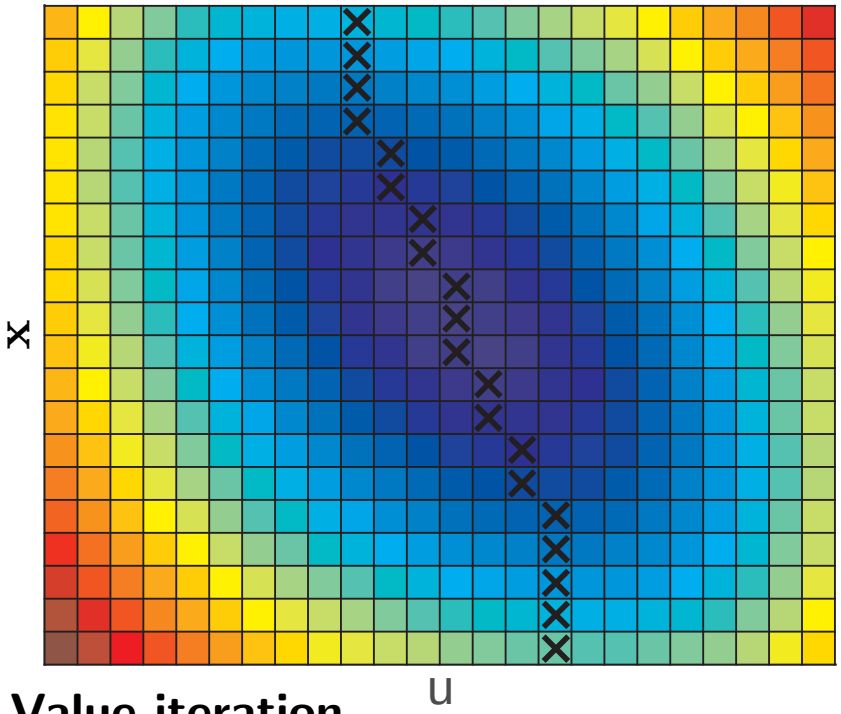
    $\text{Iter} = \text{False}$

  $Q \leftarrow Q_+$

return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$

- Iteration often written as $V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$
- No need to compute Q_π (saves a lot of computation)
- Before convergence $V(\mathbf{x})$ may not correspond to any policy
- For $\gamma < 1$, converges to $V_\star$, $Q_\star$, $\pi_\star$ for any starting V (finite)
- For $\gamma = 1$, convergence requires extra assumptions (may depend on starting V)

Dynamic Programming



Value iteration

Algorithm: Value iteration

Input: V and $\text{tol} > 0$

$\text{Iter} = \text{True}, \ Q = 0$

while Iter **do**

  Compute

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$

$$V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$$

  if $\|Q_+ - Q\| < \text{tol}$ **then**

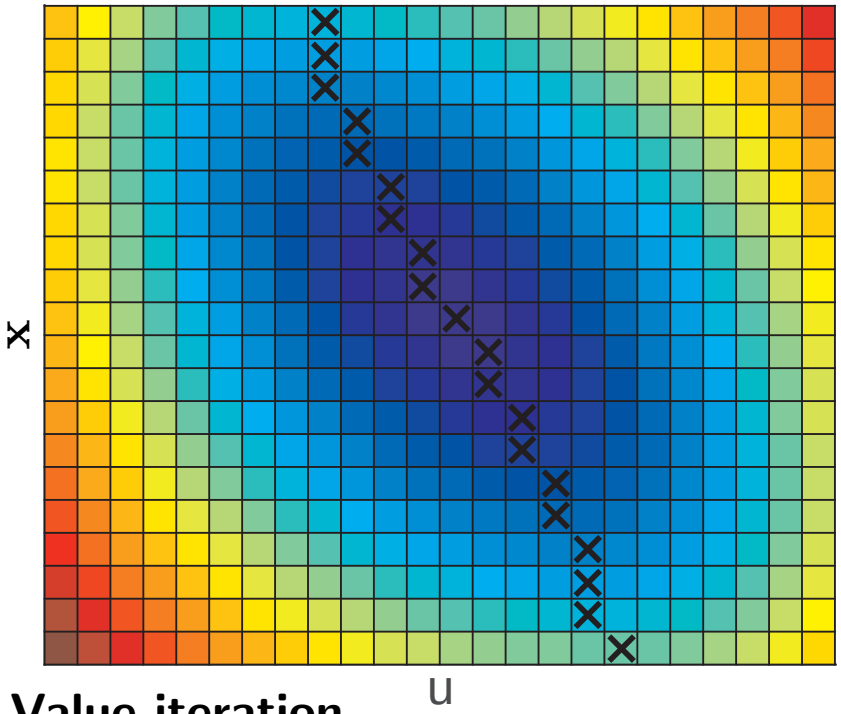
    $\text{Iter} = \text{False}$

  $Q \leftarrow Q_+$

return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$

- Iteration often written as $V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$
- No need to compute Q_π (saves a lot of computation)
- Before convergence $V(\mathbf{x})$ may not correspond to any policy
- For $\gamma < 1$, converges to $V_\star$, $Q_\star$, $\pi_\star$ for any starting V (finite)
- For $\gamma = 1$, convergence requires extra assumptions (may depend on starting V)

Dynamic Programming



Value iteration

Algorithm: Value iteration

Input: V and $\text{tol} > 0$

$\text{Iter} = \text{True}, \ Q = 0$

while Iter **do**

  Compute

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$

$$V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$$

  if $\|Q_+ - Q\| < \text{tol}$ **then**

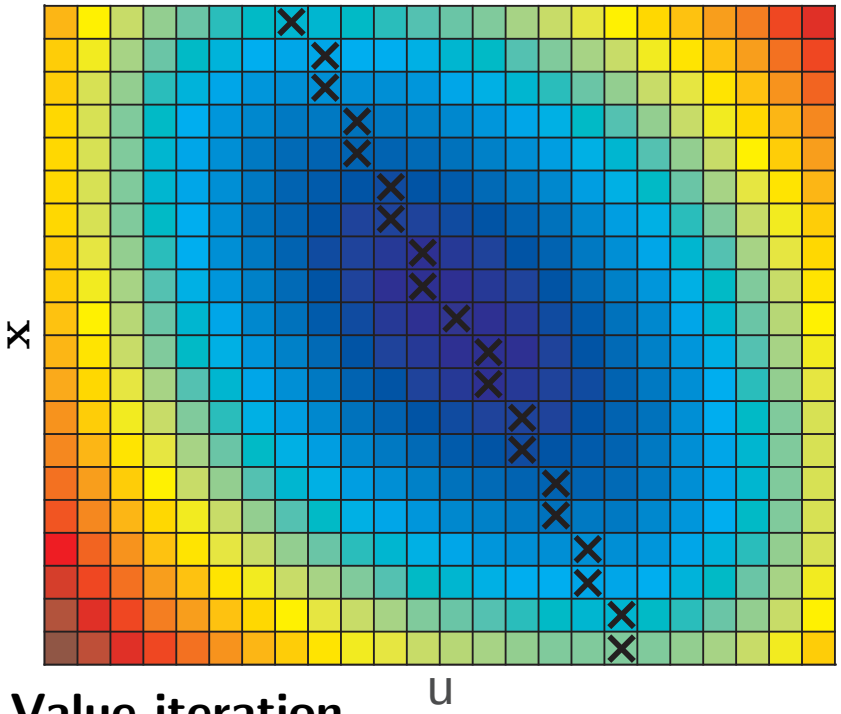
    $\text{Iter} = \text{False}$

  $Q \leftarrow Q_+$

return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$

- Iteration often written as $V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$
- No need to compute Q_π (saves a lot of computation)
- Before convergence $V(\mathbf{x})$ may not correspond to any policy
- For $\gamma < 1$, converges to $V_\star$, $Q_\star$, $\pi_\star$ for any starting V (finite)
- For $\gamma = 1$, convergence requires extra assumptions (may depend on starting V)

Dynamic Programming



Value iteration

Algorithm: Value iteration

Input: V and $\text{tol} > 0$

$\text{Iter} = \text{True}, \ Q = 0$

while Iter **do**

  Compute

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$

$$V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$$

  if $\|Q_+ - Q\| < \text{tol}$ **then**

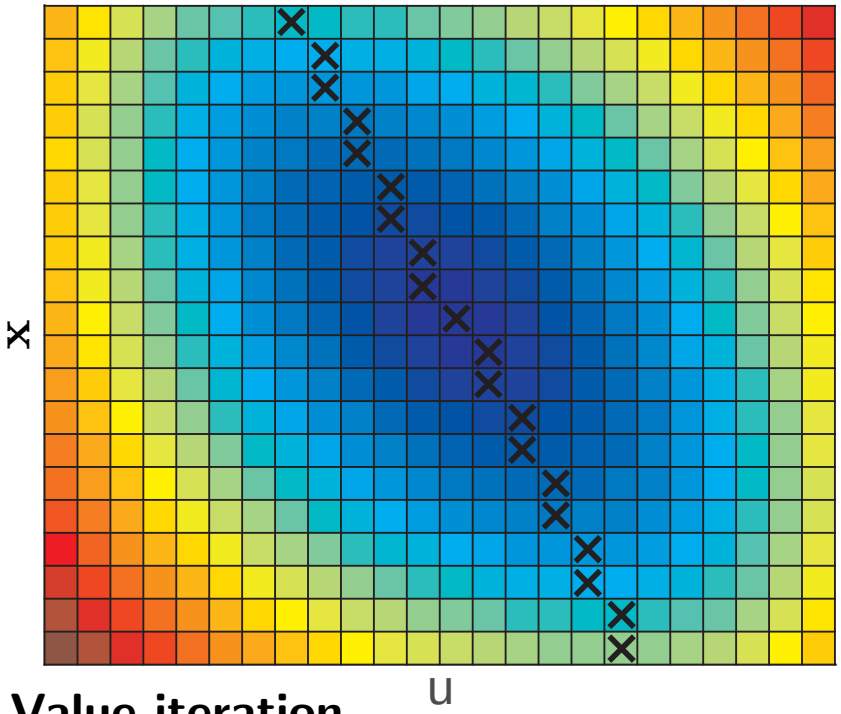
    $\text{Iter} = \text{False}$

  $Q \leftarrow Q_+$

return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$

- Iteration often written as $V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$
- No need to compute Q_π (saves a lot of computation)
- Before convergence $V(\mathbf{x})$ may not correspond to any policy
- For $\gamma < 1$, converges to $V_\star$, $Q_\star$, $\pi_\star$ for any starting V (finite)
- For $\gamma = 1$, convergence requires extra assumptions (may depend on starting V)

Dynamic Programming



Value iteration

Algorithm: Value iteration

Input: V and $\text{tol} > 0$

$\text{Iter} = \text{True}, \ Q = 0$

while Iter **do**

  Compute

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$

$$V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$$

  if $\|Q_+ - Q\| < \text{tol}$ **then**

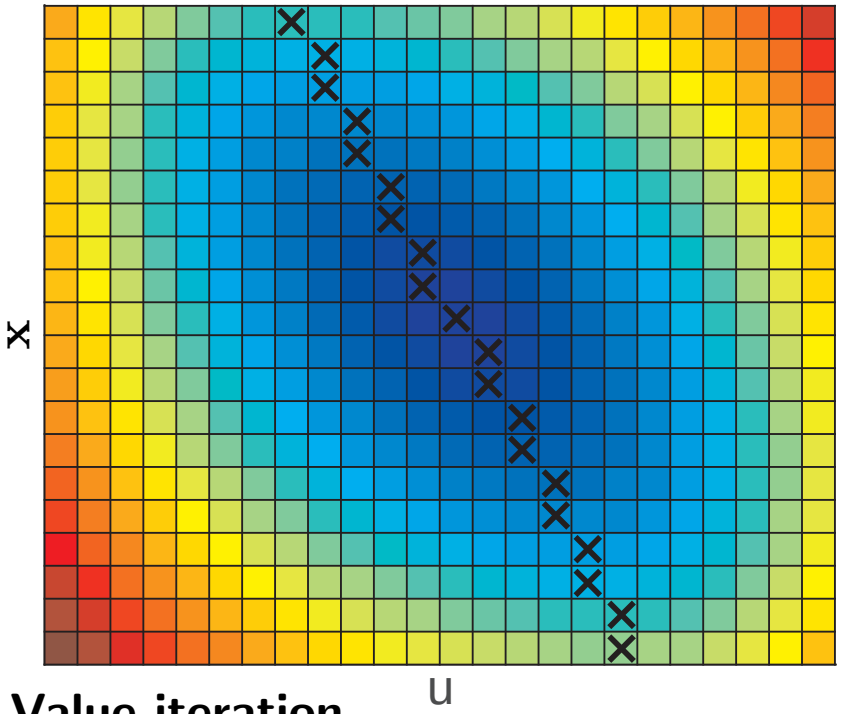
    $\text{Iter} = \text{False}$

  $Q \leftarrow Q_+$

return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$

- Iteration often written as $V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$
- No need to compute Q_π (saves a lot of computation)
- Before convergence $V(\mathbf{x})$ may not correspond to any policy
- For $\gamma < 1$, converges to $V_\star$, $Q_\star$, $\pi_\star$ for any starting V (finite)
- For $\gamma = 1$, convergence requires extra assumptions (may depend on starting V)

Dynamic Programming



Value iteration

Algorithm: Value iteration

Input: V and $\text{tol} > 0$

$\text{Iter} = \text{True}, \ Q = 0$

while Iter **do**

  Compute

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$

$$V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$$

  if $\|Q_+ - Q\| < \text{tol}$ **then**

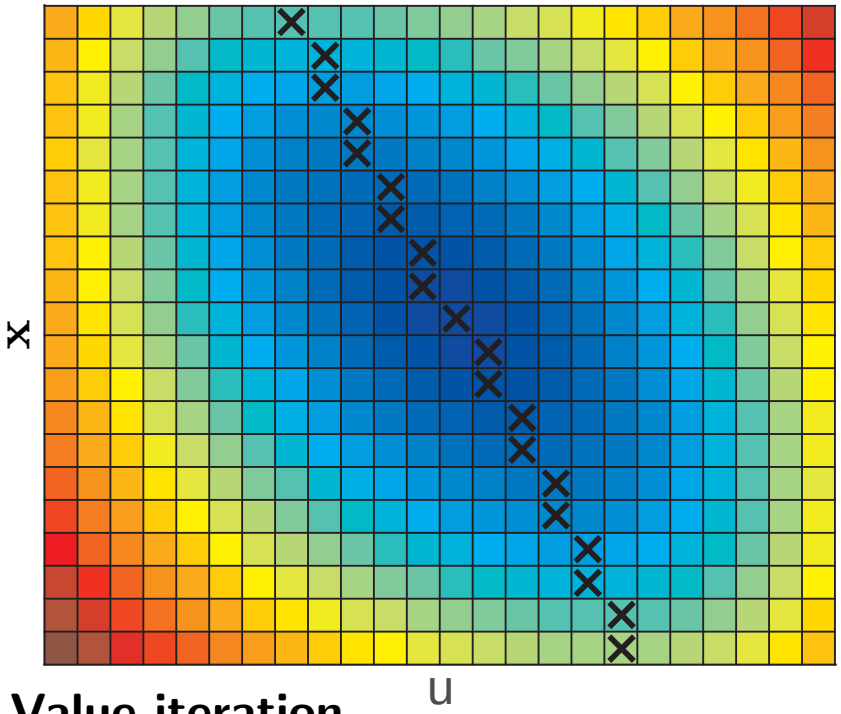
    $\text{Iter} = \text{False}$

  $Q \leftarrow Q_+$

return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$

- Iteration often written as $V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$
- No need to compute Q_π (saves a lot of computation)
- Before convergence $V(\mathbf{x})$ may not correspond to any policy
- For $\gamma < 1$, converges to $V_\star$, $Q_\star$, $\pi_\star$ for any starting V (finite)
- For $\gamma = 1$, convergence requires extra assumptions (may depend on starting V)

Dynamic Programming



Value iteration

Algorithm: Value iteration

Input: V and $\text{tol} > 0$

$\text{Iter} = \text{True}, \ Q = 0$

while Iter **do**

  Compute

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$

$$V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$$

  if $\|Q_+ - Q\| < \text{tol}$ **then**

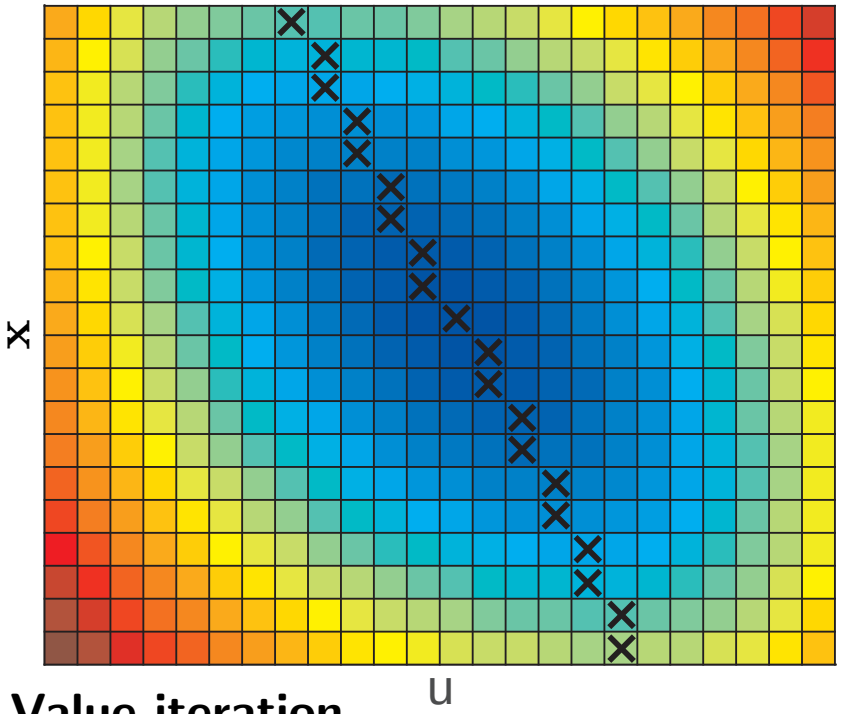
    $\text{Iter} = \text{False}$

  $Q \leftarrow Q_+$

return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$

- Iteration often written as $V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$
- No need to compute Q_π (saves a lot of computation)
- Before convergence $V(\mathbf{x})$ may not correspond to any policy
- For $\gamma < 1$, converges to $V_\star$, $Q_\star$, $\pi_\star$ for any starting V (finite)
- For $\gamma = 1$, convergence requires extra assumptions (may depend on starting V)

Dynamic Programming



Value iteration

Algorithm: Value iteration

Input: V and $\text{tol} > 0$

$\text{Iter} = \text{True}, \ Q = 0$

while Iter **do**

  Compute

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$

$$V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$$

  if $\|Q_+ - Q\| < \text{tol}$ **then**

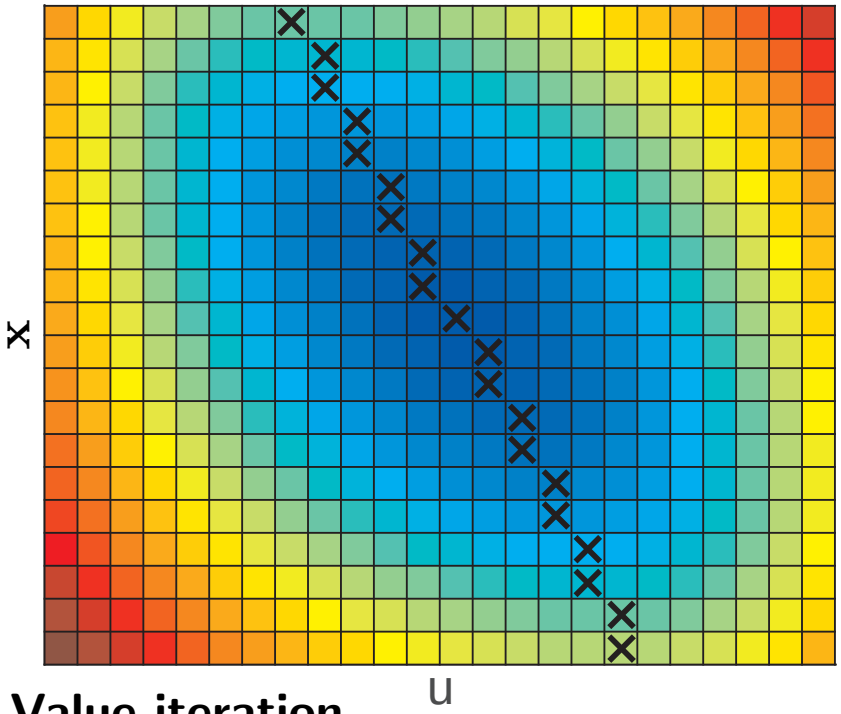
    $\text{Iter} = \text{False}$

  $Q \leftarrow Q_+$

return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$

- Iteration often written as $V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$
- No need to compute Q_π (saves a lot of computation)
- Before convergence $V(\mathbf{x})$ may not correspond to any policy
- For $\gamma < 1$, converges to $V_\star$, $Q_\star$, $\pi_\star$ for any starting V (finite)
- For $\gamma = 1$, convergence requires extra assumptions (may depend on starting V)

Dynamic Programming



Value iteration

Algorithm: Value iteration

Input: V and $\text{tol} > 0$

$\text{Iter} = \text{True}, \ Q = 0$

while Iter **do**

  Compute

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$

$$V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$$

  if $\|Q_+ - Q\| < \text{tol}$ **then**

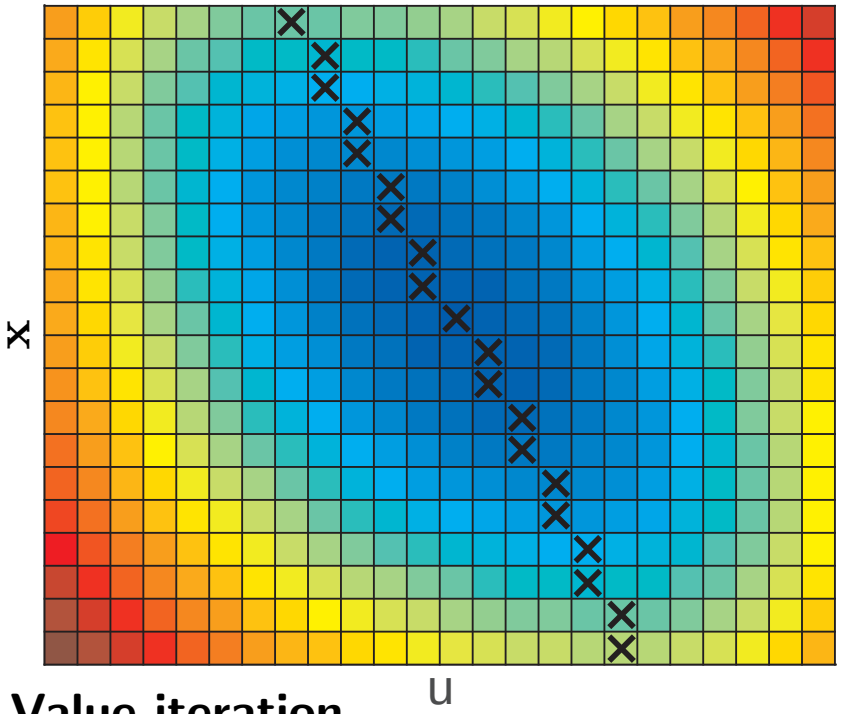
    $\text{Iter} = \text{False}$

  $Q \leftarrow Q_+$

return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$

- Iteration often written as $V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$
- No need to compute Q_π (saves a lot of computation)
- Before convergence $V(\mathbf{x})$ may not correspond to any policy
- For $\gamma < 1$, converges to $V_\star$, $Q_\star$, $\pi_\star$ for any starting V (finite)
- For $\gamma = 1$, convergence requires extra assumptions (may depend on starting V)

Dynamic Programming



Value iteration

Algorithm: Value iteration

Input: V and $\text{tol} > 0$

$\text{Iter} = \text{True}, \ Q = 0$

while Iter **do**

  Compute

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$

$$V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$$

  if $\|Q_+ - Q\| < \text{tol}$ **then**

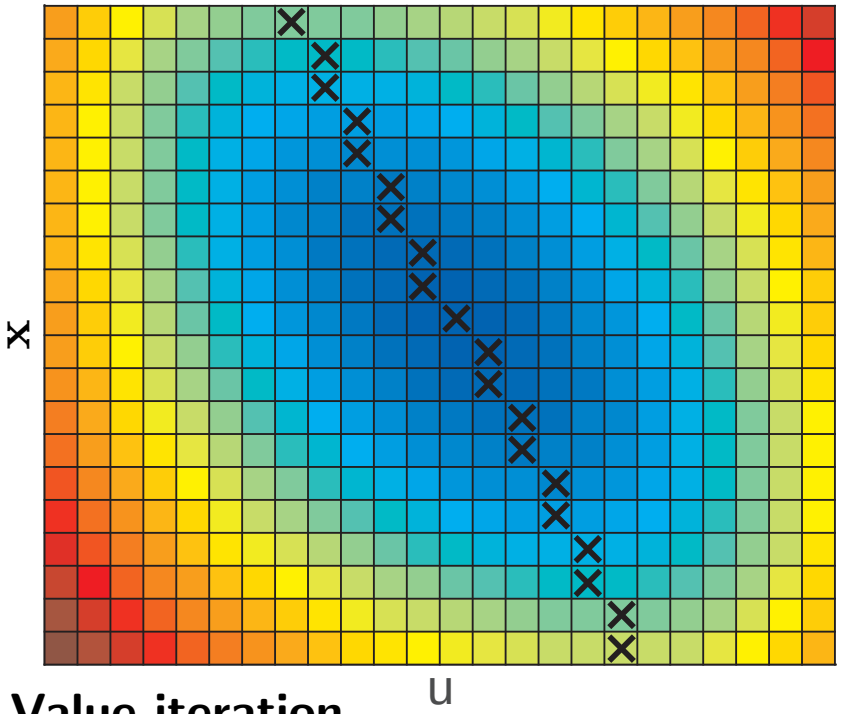
    $\text{Iter} = \text{False}$

  $Q \leftarrow Q_+$

return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$

- Iteration often written as $V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$
- No need to compute Q_π (saves a lot of computation)
- Before convergence $V(\mathbf{x})$ may not correspond to any policy
- For $\gamma < 1$, converges to $V_\star$, $Q_\star$, $\pi_\star$ for any starting V (finite)
- For $\gamma = 1$, convergence requires extra assumptions (may depend on starting V)

Dynamic Programming



Value iteration

Algorithm: Value iteration

Input: V and $\text{tol} > 0$

$\text{Iter} = \text{True}, \ Q = 0$

while Iter **do**

  Compute

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$

$$V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$$

  if $\|Q_+ - Q\| < \text{tol}$ **then**

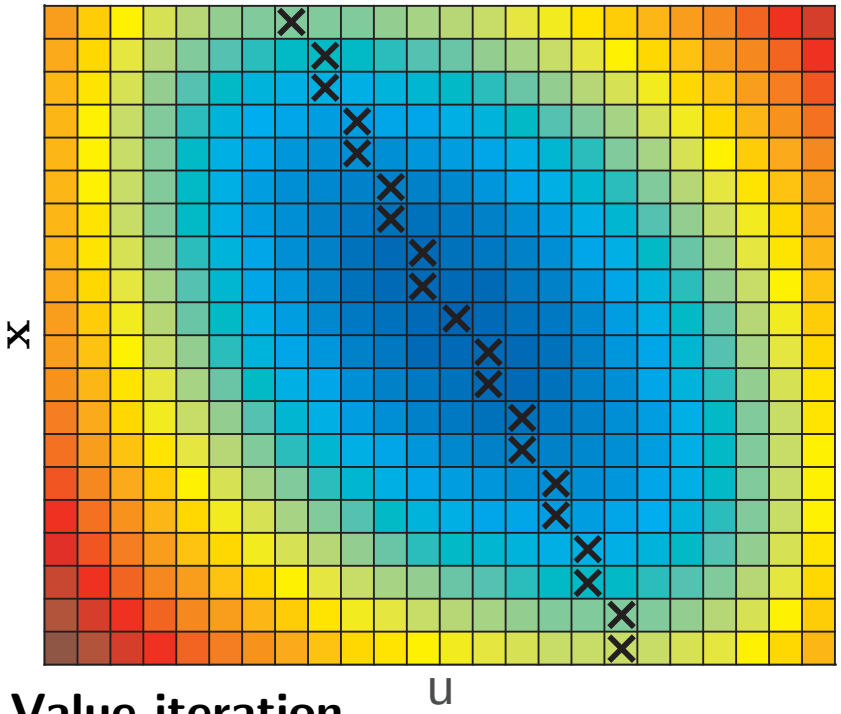
    $\text{Iter} = \text{False}$

  $Q \leftarrow Q_+$

return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$

- Iteration often written as $V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$
- No need to compute Q_π (saves a lot of computation)
- Before convergence $V(\mathbf{x})$ may not correspond to any policy
- For $\gamma < 1$, converges to $V_\star$, $Q_\star$, $\pi_\star$ for any starting V (finite)
- For $\gamma = 1$, convergence requires extra assumptions (may depend on starting V)

Dynamic Programming



Value iteration

Algorithm: Value iteration

Input: V and $\text{tol} > 0$

$\text{Iter} = \text{True}, \ Q = 0$

while Iter **do**

  Compute

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$

$$V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$$

  if $\|Q_+ - Q\| < \text{tol}$ **then**

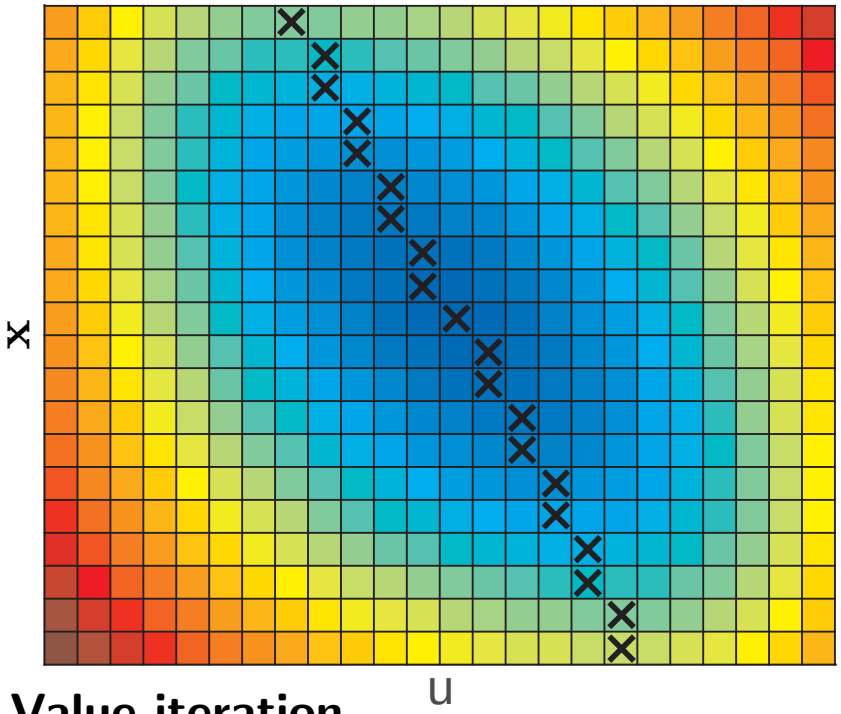
    $\text{Iter} = \text{False}$

  $Q \leftarrow Q_+$

return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$

- Iteration often written as $V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$
- No need to compute Q_π (saves a lot of computation)
- Before convergence $V(\mathbf{x})$ may not correspond to any policy
- For $\gamma < 1$, converges to $V_\star$, $Q_\star$, $\pi_\star$ for any starting V (finite)
- For $\gamma = 1$, convergence requires extra assumptions (may depend on starting V)

Dynamic Programming



Value iteration

Algorithm: Value iteration

Input: V and $\text{tol} > 0$

$\text{Iter} = \text{True}, \ Q = 0$

while Iter **do**

  Compute

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$

$$V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$$

  if $\|Q_+ - Q\| < \text{tol}$ **then**

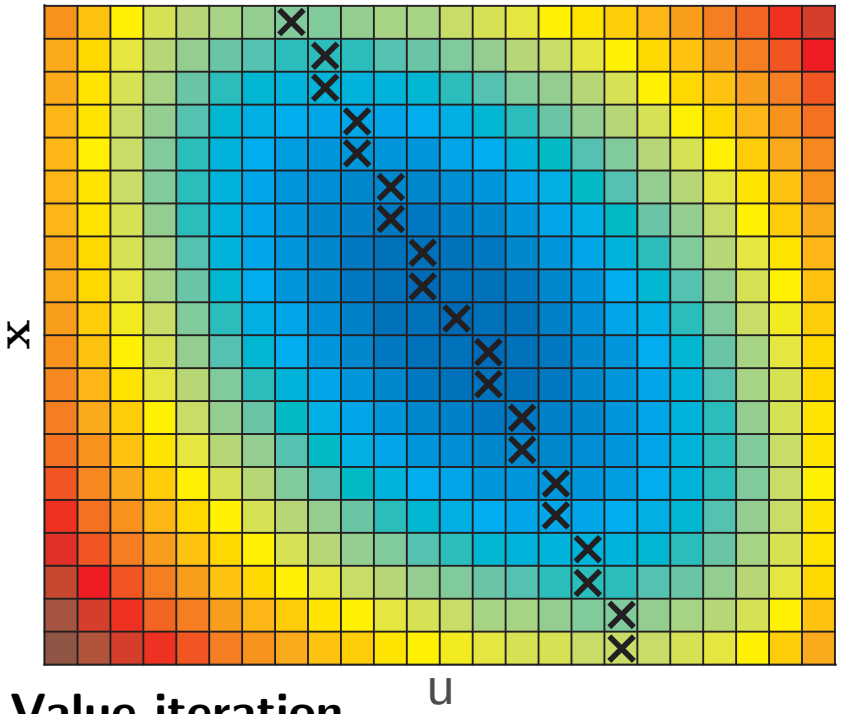
    $\text{Iter} = \text{False}$

  $Q \leftarrow Q_+$

return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$

- Iteration often written as $V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$
- No need to compute Q_π (saves a lot of computation)
- Before convergence $V(\mathbf{x})$ may not correspond to any policy
- For $\gamma < 1$, converges to $V_\star$, $Q_\star$, $\pi_\star$ for any starting V (finite)
- For $\gamma = 1$, convergence requires extra assumptions (may depend on starting V)

Dynamic Programming



Value iteration

Algorithm: Value iteration

Input: V and $\text{tol} > 0$

$\text{Iter} = \text{True}, \ Q = 0$

while Iter **do**

  Compute

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$

$$V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$$

  if $\|Q_+ - Q\| < \text{tol}$ **then**

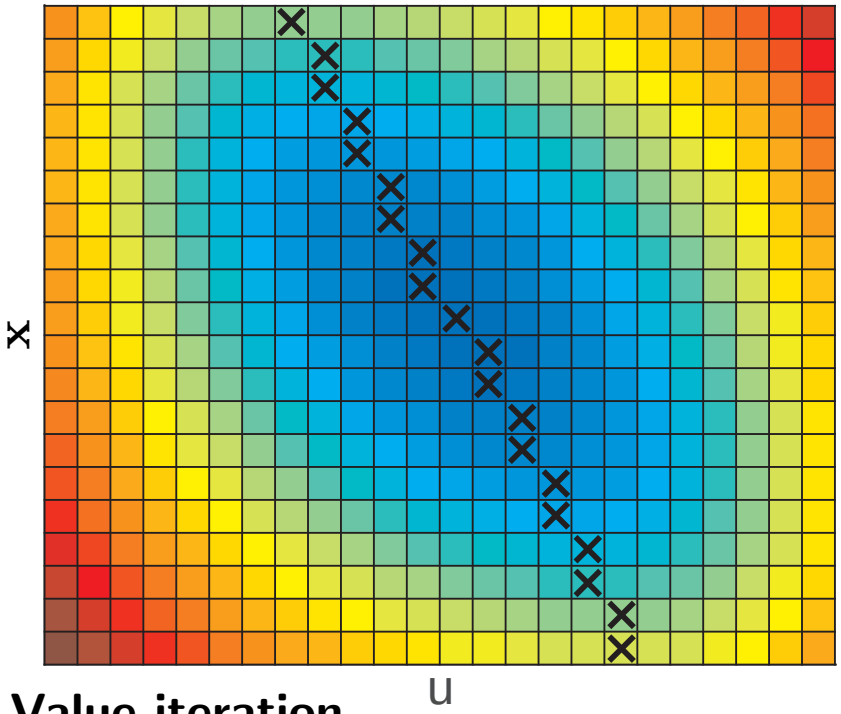
    $\text{Iter} = \text{False}$

  $Q \leftarrow Q_+$

return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$

- Iteration often written as $V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$
- No need to compute Q_π (saves a lot of computation)
- Before convergence $V(\mathbf{x})$ may not correspond to any policy
- For $\gamma < 1$, converges to $V_\star$, $Q_\star$, $\pi_\star$ for any starting V (finite)
- For $\gamma = 1$, convergence requires extra assumptions (may depend on starting V)

Dynamic Programming



Value iteration

Algorithm: Value iteration

Input: V and $\text{tol} > 0$

$\text{Iter} = \text{True}, \ Q = 0$

while Iter **do**

  Compute

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$

$$V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$$

  if $\|Q_+ - Q\| < \text{tol}$ **then**

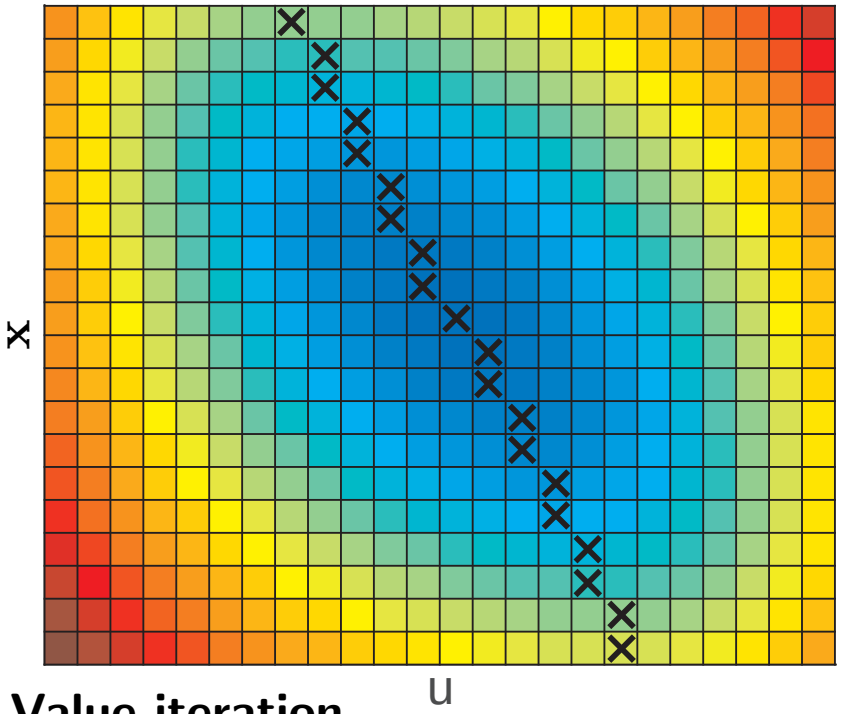
    $\text{Iter} = \text{False}$

  $Q \leftarrow Q_+$

return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$

- Iteration often written as $V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$
- No need to compute Q_π (saves a lot of computation)
- Before convergence $V(\mathbf{x})$ may not correspond to any policy
- For $\gamma < 1$, converges to $V_\star$, $Q_\star$, $\pi_\star$ for any starting V (finite)
- For $\gamma = 1$, convergence requires extra assumptions (may depend on starting V)

Dynamic Programming



Value iteration

Algorithm: Value iteration

Input: V and $\text{tol} > 0$

$\text{Iter} = \text{True}, \ Q = 0$

while Iter **do**

  Compute

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$

$$V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$$

  if $\|Q_+ - Q\| < \text{tol}$ **then**

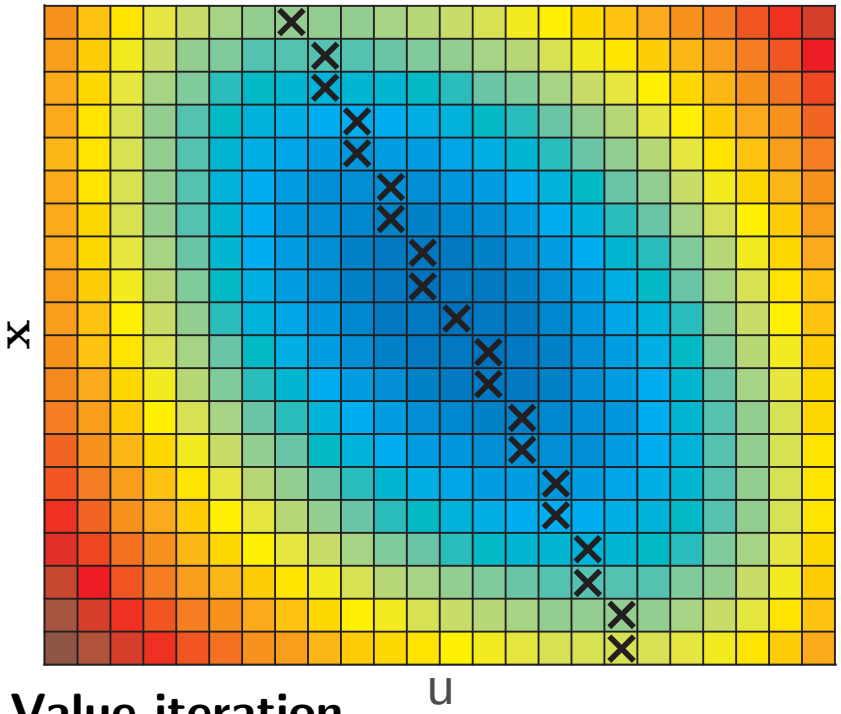
    $\text{Iter} = \text{False}$

  $Q \leftarrow Q_+$

return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$

- Iteration often written as $V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$
- No need to compute Q_π (saves a lot of computation)
- Before convergence $V(\mathbf{x})$ may not correspond to any policy
- For $\gamma < 1$, converges to $V_\star$, $Q_\star$, $\pi_\star$ for any starting V (finite)
- For $\gamma = 1$, convergence requires extra assumptions (may depend on starting V)

Dynamic Programming



Value iteration

Algorithm: Value iteration

Input: V and $\text{tol} > 0$

$\text{Iter} = \text{True}, \ Q = 0$

while Iter **do**

  Compute

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$

$$V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$$

  if $\|Q_+ - Q\| < \text{tol}$ **then**

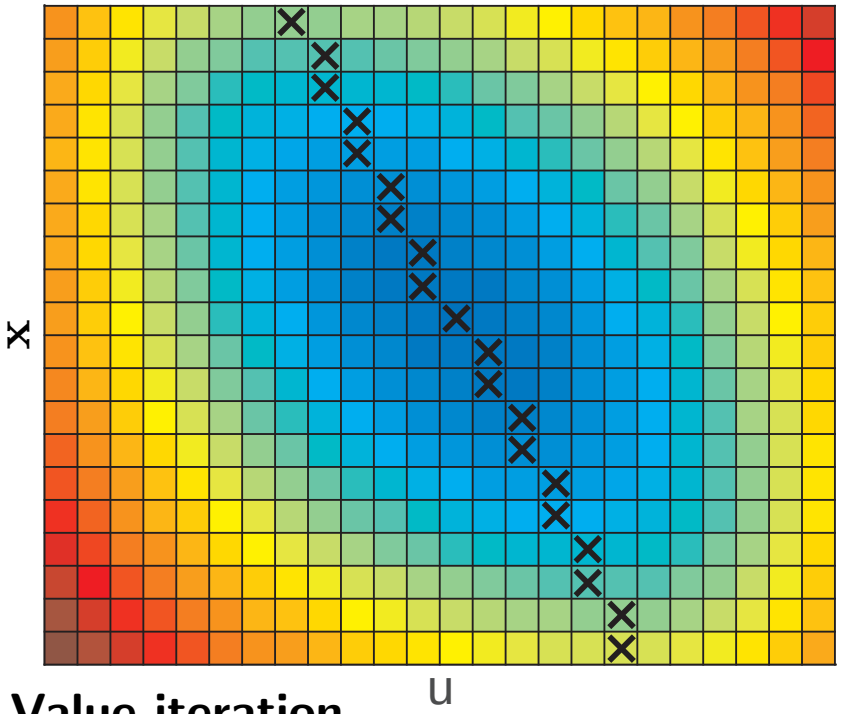
    $\text{Iter} = \text{False}$

  $Q \leftarrow Q_+$

return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$

- Iteration often written as $V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$
- No need to compute Q_π (saves a lot of computation)
- Before convergence $V(\mathbf{x})$ may not correspond to any policy
- For $\gamma < 1$, converges to $V_\star$, $Q_\star$, $\pi_\star$ for any starting V (finite)
- For $\gamma = 1$, convergence requires extra assumptions (may depend on starting V)

Dynamic Programming



Value iteration

Algorithm: Value iteration

Input: V and $\text{tol} > 0$

$\text{Iter} = \text{True}, \ Q = 0$

while Iter **do**

  Compute

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$

$$V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$$

  if $\|Q_+ - Q\| < \text{tol}$ **then**

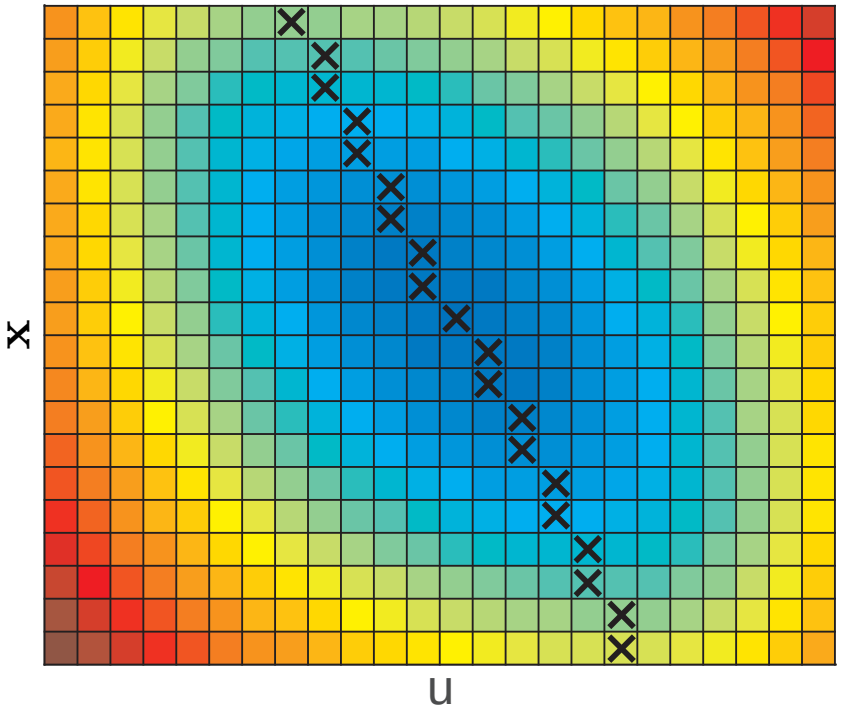
    $\text{Iter} = \text{False}$

  $Q \leftarrow Q_+$

return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$

- Iteration often written as $V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$
- No need to compute Q_π (saves a lot of computation)
- Before convergence $V(\mathbf{x})$ may not correspond to any policy
- For $\gamma < 1$, converges to $V_\star$, $Q_\star$, $\pi_\star$ for any starting V (finite)
- For $\gamma = 1$, convergence requires extra assumptions (may depend on starting V)

Dynamic Programming



Algorithm: Value iteration

Input: V and $\mathrm{tol} > 0$

$\mathrm{Iter} = \mathrm{True}, \ Q = 0$

while Iter **do**

  Compute

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V\left(\mathbf{x}_+\right) \mid \mathbf{x}, \mathbf{u}\right]$$

$$V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$$

  if $\|Q_+ - Q\| < \mathrm{tol}$ **then**

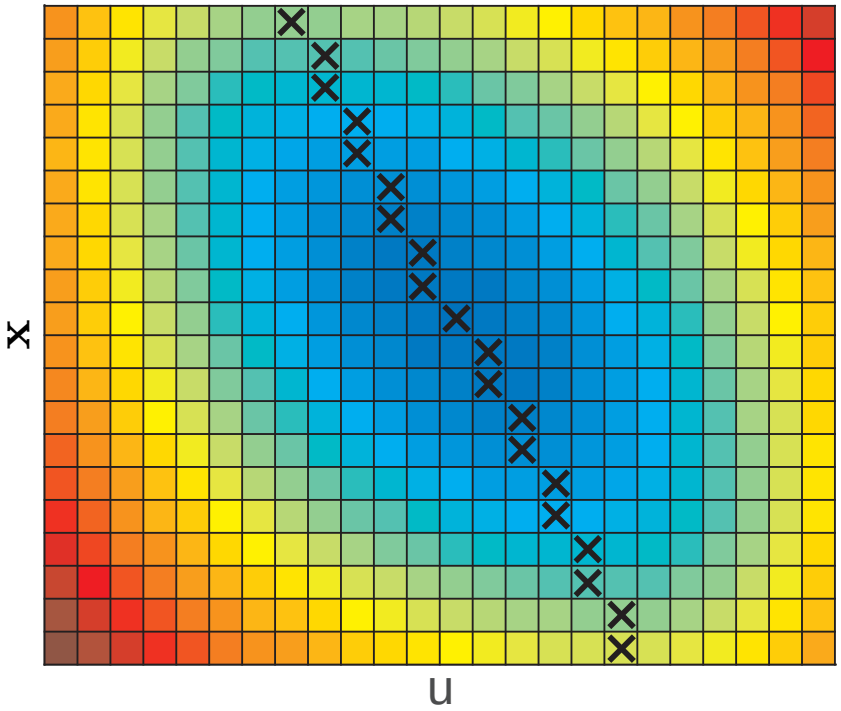
    $\mathrm{Iter} = \mathrm{False}$

  $Q \leftarrow Q_+$

return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$

- Policy π converges to $\pi_\star$ after a few iterations
- Action-value function Q converges to $Q_\star$ after (many) more iterations
- Hence π becomes correct "long" before Q_π is fully converged

Dynamic Programming



Algorithm: Value iteration

Input: V and $\mathrm{tol} > 0$

$\mathrm{Iter} = \mathrm{True}, \ Q = 0$

while Iter **do**

  Compute

$$Q_+(\mathbf{x}, \mathbf{u}) \leftarrow L(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}\left[V(\mathbf{x}_+) \mid \mathbf{x}, \mathbf{u}\right]$$

$$V(\mathbf{x}) \leftarrow \min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$$

  if $\|Q_+ - Q\| < \mathrm{tol}$ **then**

    $\mathrm{Iter} = \mathrm{False}$

  $Q \leftarrow Q_+$

return $\pi_\star \leftarrow \arg\min_{\mathbf{u}} \ Q_+(\mathbf{x}, \mathbf{u})$

- Policy π converges to $\pi_\star$ after a few iterations
- Action-value function Q converges to $Q_\star$ after (many) more iterations
- Hence π becomes correct "long" before Q_π is fully converged

Value iteration often labelled Dynamic Programming, for $\gamma < 1$ convergence to optimal policy is guaranteed for any (finite) starting V

Curse of dimensionality

The Bellman equations & DP are very powerful mathematical tools

- One can treat *any* (possibly) stochastic optimal control problem
- The global optimum is achieved
- Policy $\mathbf{u} = \boldsymbol{\pi}(\mathbf{x})$ provides a feedback law, no extra online work needed

why don't we just use that?!

Curse of dimensionality

The Bellman equations & DP are very powerful mathematical tools

- One can treat *any* (possibly) stochastic optimal control problem
- The global optimum is achieved
- Policy $\mathbf{u} = \boldsymbol{\pi}(\mathbf{x})$ provides a feedback law, no extra online work needed

why don't we just use that?!

Dynamic Programming is "cursed"

Curse of dimensionality

The Bellman equations & DP are very powerful mathematical tools

- One can treat *any* (possibly) stochastic optimal control problem
- The global optimum is achieved
- Policy $\mathbf{u} = \boldsymbol{\pi}(\mathbf{x})$ provides a feedback law, no extra online work needed

why don't we just use that?!

Dynamic Programming is "cursed"

E.g.

- Imagine a system with **6 states, 2 inputs**
- Create **grids of 100 discrete values** for each

Curse of dimensionality

The Bellman equations & DP are very powerful mathematical tools

- One can treat *any* (possibly) stochastic optimal control problem
- The global optimum is achieved
- Policy $\mathbf{u} = \boldsymbol{\pi}(\mathbf{x})$ provides a feedback law, no extra online work needed

why don't we just use that?!

Dynamic Programming is "cursed"

E.g.

- Imagine a system with **6 states, 2 inputs**
- Create **grids of 100 discrete values** for each

then e.g. storing $Q(\mathbf{x}, \mathbf{u})$ requires:

$$100^8 = 10^{16} \text{ numbers } \equiv 10000 \text{ terabytes of memory}$$

and each DP iteration requires going through 10^{32} numbers

Outline

Contraction of Bellman backup

- We can define the Bellman backup operator $\mathcal{T}_{\pi}$:

$$\mathcal{T}_{\pi}(V) = L(\,.\,, \pi(\,.\,)) + \mathbb{E}[V(\mathbf{x}_{+}) \,|\, .\,, \pi(\,.\,)]$$

that transforms a function V into a new V. E.g. π-evaluation iterates $V \leftarrow \mathcal{T}_{\pi}(V)$

[1] otherwise we need a bit of measure theory

Contraction of Bellman backup

- We can define the Bellman backup operator $\mathcal{T}_{\boldsymbol{\pi}}$:

$$\mathcal{T}_{\boldsymbol{\pi}}(V) = L(\,.\,, \boldsymbol{\pi}(\,.\,)) + \mathbb{E}[V(\mathbf{x}_+) \,|\, .\,, \boldsymbol{\pi}(\,.\,)]$$

  that transforms a function V into a new V. E.g. $\boldsymbol{\pi}$-evaluation iterates $V \leftarrow \mathcal{T}_{\boldsymbol{\pi}}(V)$

- We can define a norm over functions $\|V\|_\infty = \max_{\mathbf{x}} |V(\mathbf{x})|$

[1] otherwise we need a bit of measure theory

Contraction of Bellman backup

- We can define the Bellman backup operator $\mathcal{T}_{\boldsymbol{\pi}}$:

$$\mathcal{T}_{\boldsymbol{\pi}}(V) = L(\,.\,, \boldsymbol{\pi}(\,.\,)) + \mathbb{E}[V(\mathbf{x}_+) \,|\, .\,, \boldsymbol{\pi}(\,.\,)]$$

  that transforms a function V into a new V. E.g. $\boldsymbol{\pi}$-evaluation iterates $V \leftarrow \mathcal{T}_{\boldsymbol{\pi}}(V)$

- We can define a norm over functions $\|V\|_\infty = \max_{\mathbf{x}} |V(\mathbf{x})|$

Then for V bounded[1]...

The operator $\mathcal{T}_{\boldsymbol{\pi}}$ is a γ-**contraction**, i.e.

$$\left\|\mathcal{T}_{\boldsymbol{\pi}}(V) - \mathcal{T}_{\boldsymbol{\pi}}(V')\right\|_\infty \leq \gamma \left\|V - V'\right\|_\infty$$

This follows from observing that for any function V and any distribution, $\mathbb{E}[V] \leq \|V\|_\infty$

[1] otherwise we need a bit of measure theory

Contraction of Bellman backup

- We can define the Bellman backup operator $\mathcal{T}_{\pi}$:

$$\mathcal{T}_{\pi}(V) = L(\,.\,, \pi(\,.\,)) + \mathbb{E}[V(\mathbf{x}_{+}) \,|\, .\,, \pi(\,.\,)]$$

  that transforms a function V into a new V. E.g. π-evaluation iterates $V \leftarrow \mathcal{T}_{\pi}(V)$
- We can define a norm over functions $\|V\|_{\infty} = \max_{\mathbf{x}} |V(\mathbf{x})|$

Then for V bounded[1]...

The operator $\mathcal{T}_{\pi}$ is a γ-**contraction**, i.e.

$$\left\|\mathcal{T}_{\pi}(V) - \mathcal{T}_{\pi}(V')\right\|_{\infty} \leq \gamma \left\|V - V'\right\|_{\infty}$$

This follows from observing that for any function V and any distribution, $\mathbb{E}[V] \leq \|V\|_{\infty}$

Consequences

- For $\gamma < 1$ π-evaluation techniques converge to V_{π} regardless of the starting V
- Convergence is linear

[1] otherwise we need a bit of measure theory

Contraction of Bellman Optimiality backup

We can define the Bellman optimality backup operator $\mathcal{T}_\star$:

$$\mathcal{T}_\star(V) = \min_{\mathbf{u}} \; L(\,.\,,\mathbf{u}) + \mathbb{E}[V(\mathbf{x}_+)\,|\,.\,,\mathbf{u}]$$

that transforms a function V into a new V. E.g. DP iterates $V \leftarrow \mathcal{T}_\star(V)$

[2] otherwise we need a bit of measure theory

Contraction of Bellman Optimiality backup

We can define the Bellman optimality backup operator $\mathcal{T}_\star$:

$$\mathcal{T}_\star(V) = \min_{\mathbf{u}} L(\,.\,,\mathbf{u}) + \mathbb{E}[V(\mathbf{x}_+) \,|\, .\,,\mathbf{u}]$$

that transforms a function V into a new V. E.g. DP iterates $V \leftarrow \mathcal{T}_\star(V)$

Then for V bounded[2]...

The operator $\mathcal{T}_\star$ is a γ-**contraction**, i.e.

$$\left\|\mathcal{T}_\star(V) - \mathcal{T}_\star(V')\right\|_\infty \leq \gamma \left\|V - V'\right\|_\infty$$

[2] otherwise we need a bit of measure theory

Contraction of Bellman Optimiality backup

We can define the Bellman optimality backup operator $\mathcal{T}_\star$:

$$\mathcal{T}_\star(V) = \min_{\mathbf{u}} L(\,.\,,\mathbf{u}) + \mathbb{E}[V(\mathbf{x}_+) \,|\, .\,,\mathbf{u}]$$

that transforms a function V into a new V. E.g. DP iterates $V \leftarrow \mathcal{T}_\star(V)$

Then for V bounded[2]...

The operator $\mathcal{T}_\star$ is a γ-**contraction**, i.e.

$$\left\|\mathcal{T}_\star(V) - \mathcal{T}_\star(V')\right\|_\infty \leq \gamma \left\|V - V'\right\|_\infty$$

Consequences

- For $\gamma < 1$ DP techniques converge to $V_\star$ regardless of the starting V
- Convergence is linear

[2] otherwise we need a bit of measure theory