Problem definition

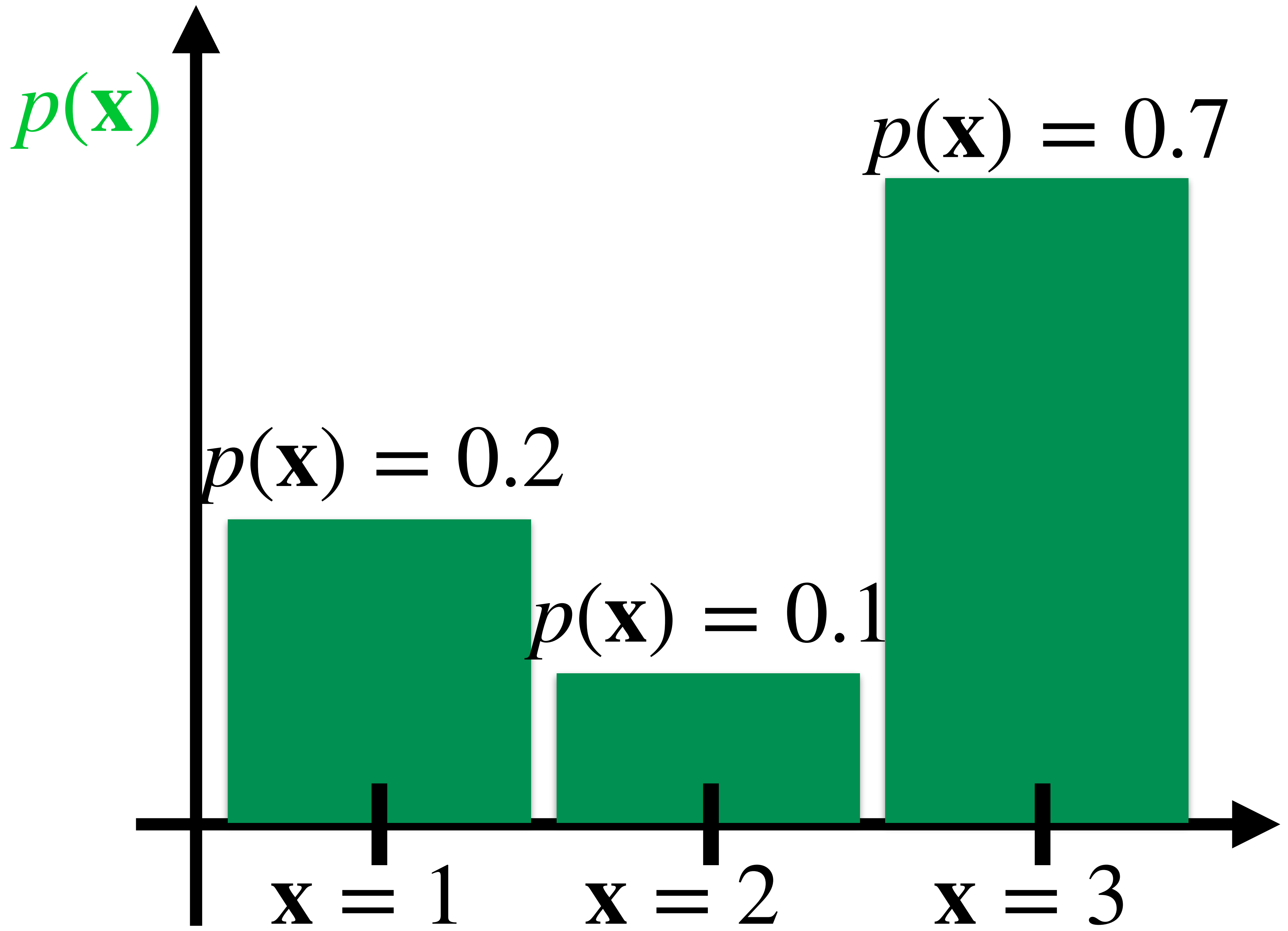
Karel Zimmermann

Prerequisites

- **Mathematical analysis (B0B01MA2):** gradient, Jacobian, Hessian, multidimensional Taylor polynomial
- **Optimization (B0B33OPT):** Gauss-Newton method, Levenberg Marquardt method, full Newton method
- **Linear algebra (B0B01LAG):** pseudo-inverse, SVD decomposition, least-squares method
- **Probability theory (B0B01PST):** multivariate gaussian probability, Bayes theorem
- **Statistics (B0B01PST):** maximum likelihood and maximum aposteriori estimate
- **Programming (B3B33ALP + B3B36PRG):** python + linux

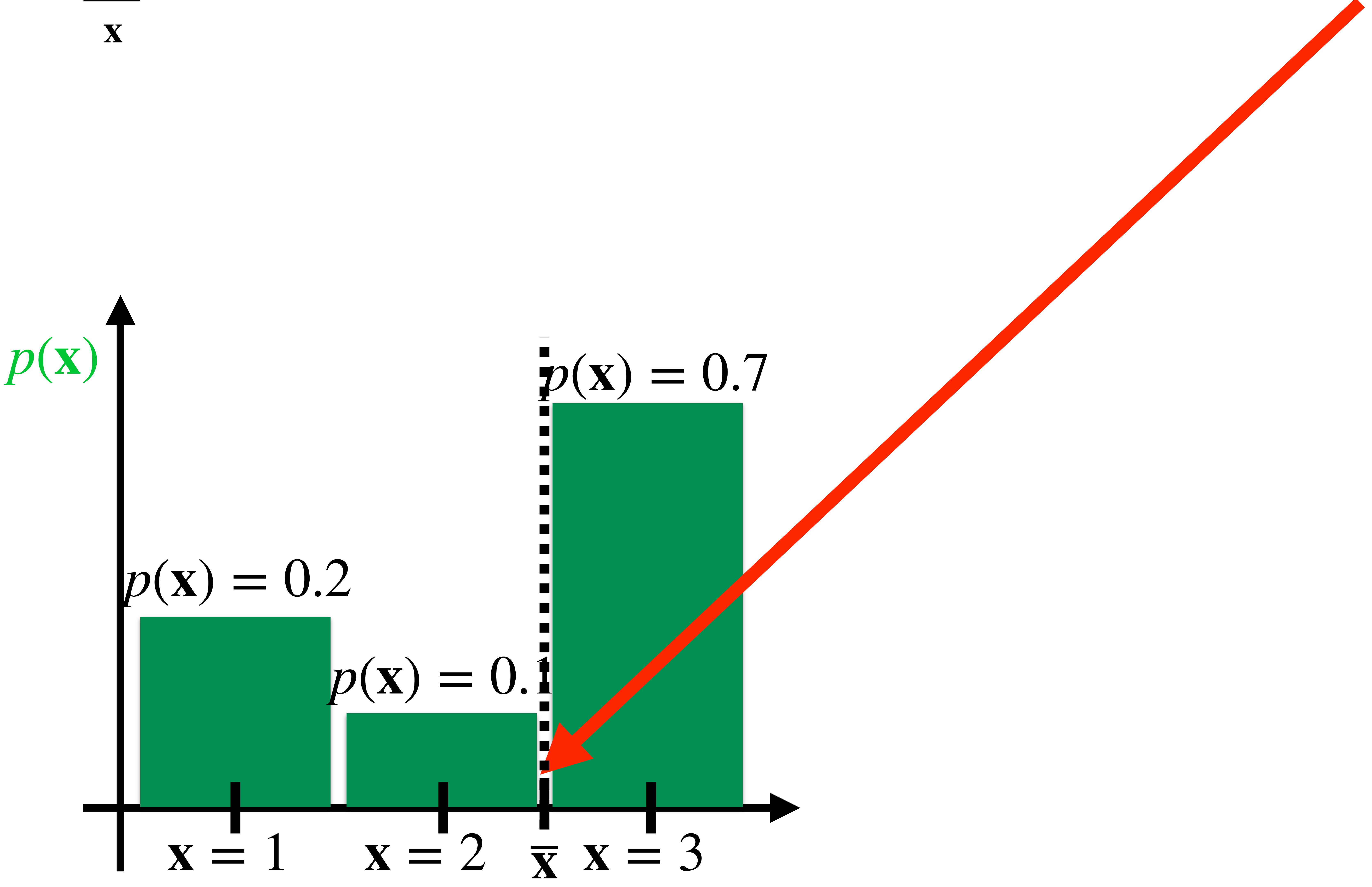
Prerequisites: Mean and average

$$\bar{\mathbf{x}} = \sum_{\mathbf{x}} p(\mathbf{x}) \cdot \mathbf{x} = \mathbb{E}_{\mathbf{x} \sim p(\mathbf{x})}\left[\mathbf{x}\right] = ??$$



Prerequisites: Mean and average

$$\bar{\mathbf{x}} = \sum_{\mathbf{x}} p(\mathbf{x}) \cdot \mathbf{x} = \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}\left[\mathbf{x}\right] = 0.2 \cdot 1 \ + \ 0.1 \cdot 2 \ + \ 0.7 \cdot 3 = 2.5$$



Prerequisites: Mean and average

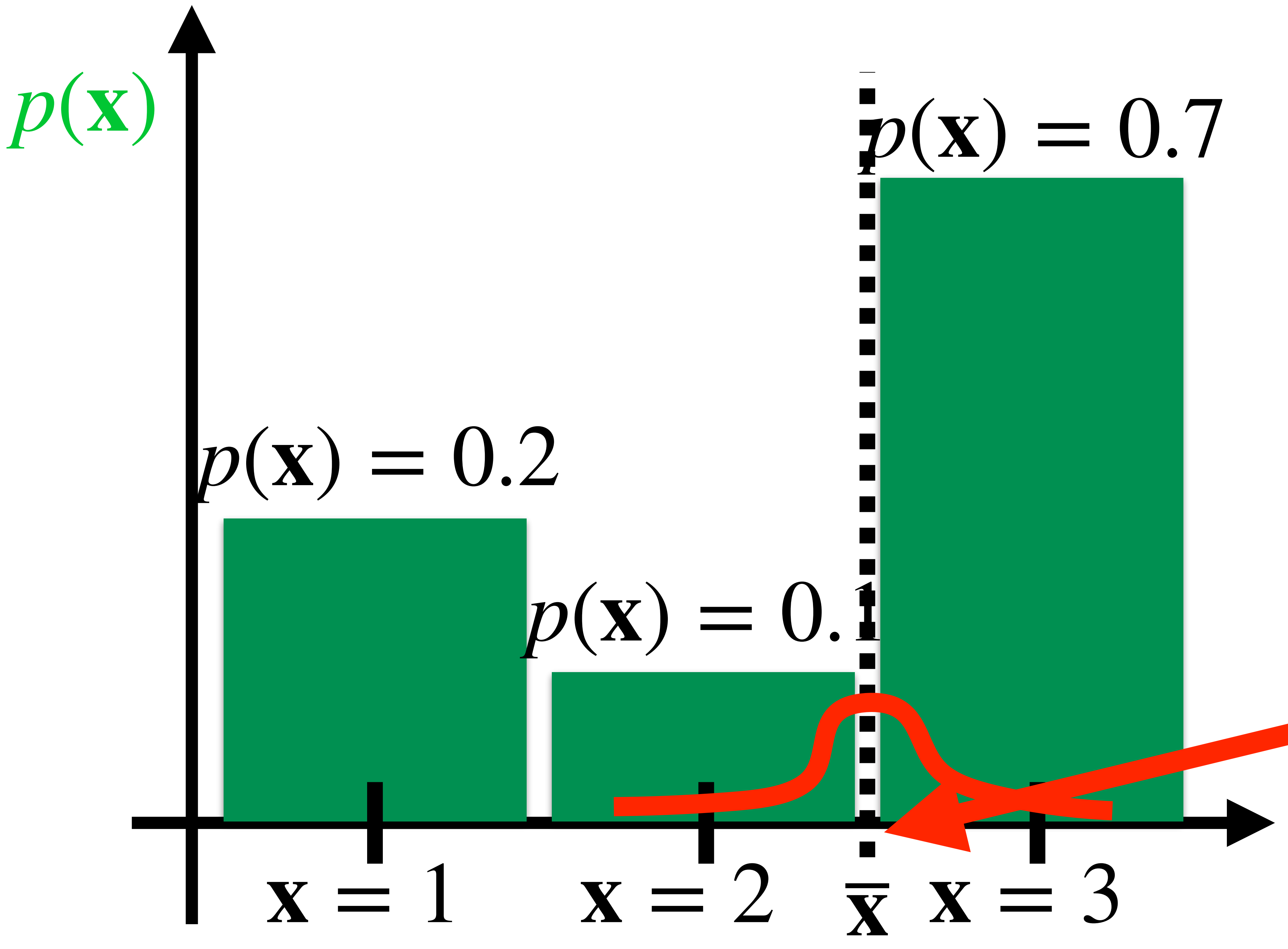
$$\bar{\mathbf{x}} = \sum_{\mathbf{x}} p(\mathbf{x}) \cdot \mathbf{x} = \mathbb{E}_{\mathbf{x} \sim p(\mathbf{x})}\left[\mathbf{x}\right] = 0.2 \cdot 1 \; + \; 0.1 \cdot 2 \; + \; 0.7 \cdot 3 = 2.5$$

$$\approx \frac{1}{N} \sum_i \mathbf{x}_i \; = \frac{1}{10}(1 + 1 + 2 + 3 + 3 + 3 + 3 + 3 + 3 + 3) = 2.5$$

where $\mathbf{x}_i \sim p$

For $N \to \infty$

$$\mathcal{N}(\bar{\mathbf{x}}_i; \;\; \bar{\mathbf{x}}, \frac{\sigma^2_{\mathbf{x}}}{\sqrt{N}})$$



$$\bar{\mathbf{x}}_1 = \frac{1}{10}(1 + 1 + 1 + 1 + 3 + 3 + 3 + 3 + 3 + 3) = 2.2$$

$$\bar{\mathbf{x}}_2 = \frac{1}{10}(3 + 3 + 3 + 3 + 3 + 3 + 3 + 3 + 3 + 3) = 3.0$$

$$\bar{\mathbf{x}}_3 = \frac{1}{10}(2 + 2 + 2 + 2 + 3 + 3 + 3 + 3 + 3 + 3) = 2.6$$

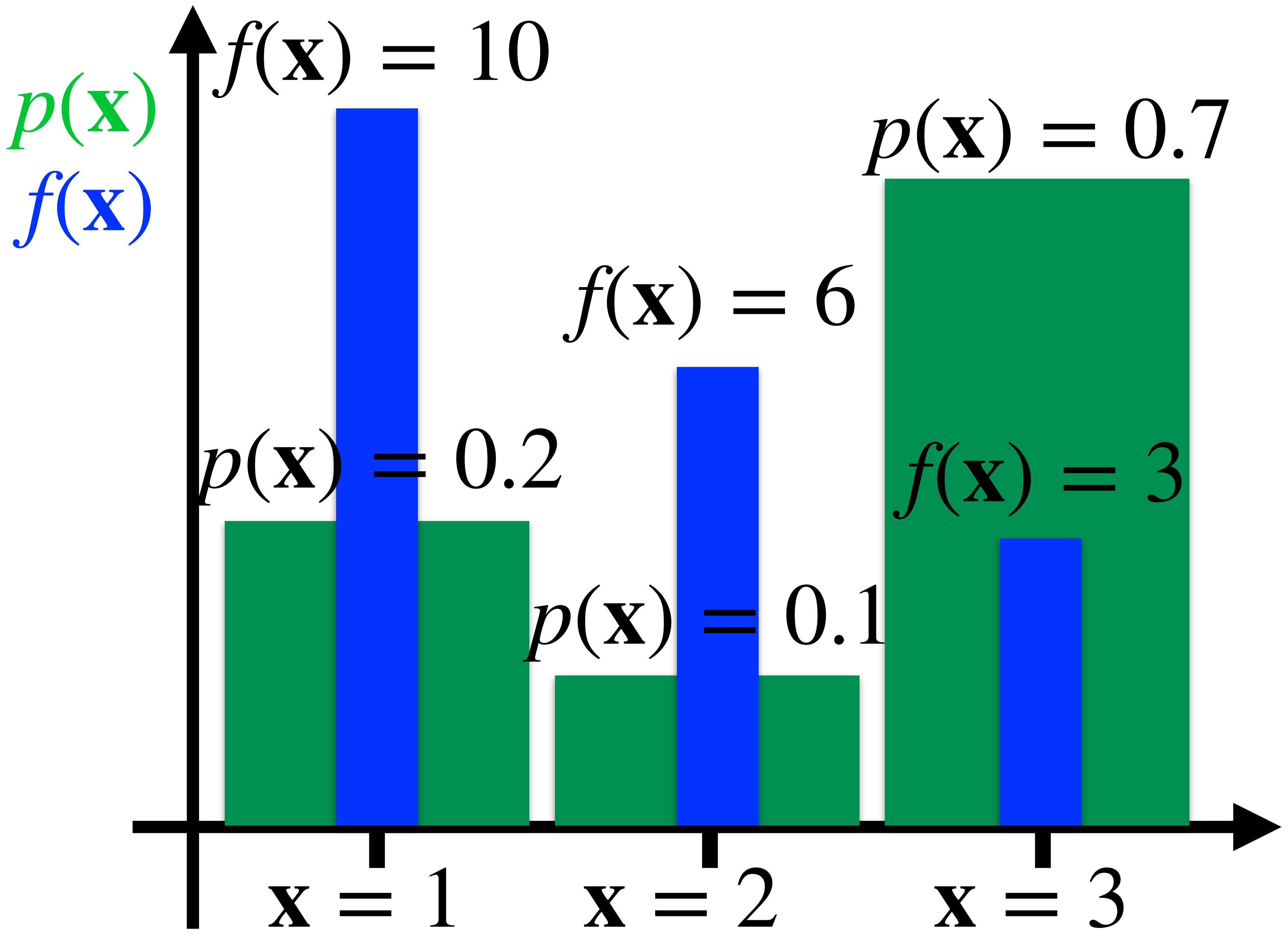
$$\bar{\mathbf{x}}_4 = \frac{1}{10}(1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 2 + 2) = 1.2$$

$$\bar{\mathbf{x}}_5 = \frac{1}{10}(1 + 1 + 1 + 1 + 1 + 3 + 3 + 3 + 3 + 3) = 2.0$$

Prerequisites: Mean and average

$$\bar{f} = \sum_{\mathbf{x}} p(\mathbf{x}) \cdot f(\mathbf{x}) = \mathbb{E}_{\mathbf{x} \sim p(\mathbf{x})}\left[f(\mathbf{x})\right] = ??$$

where $\mathbf{x}_i \sim p$

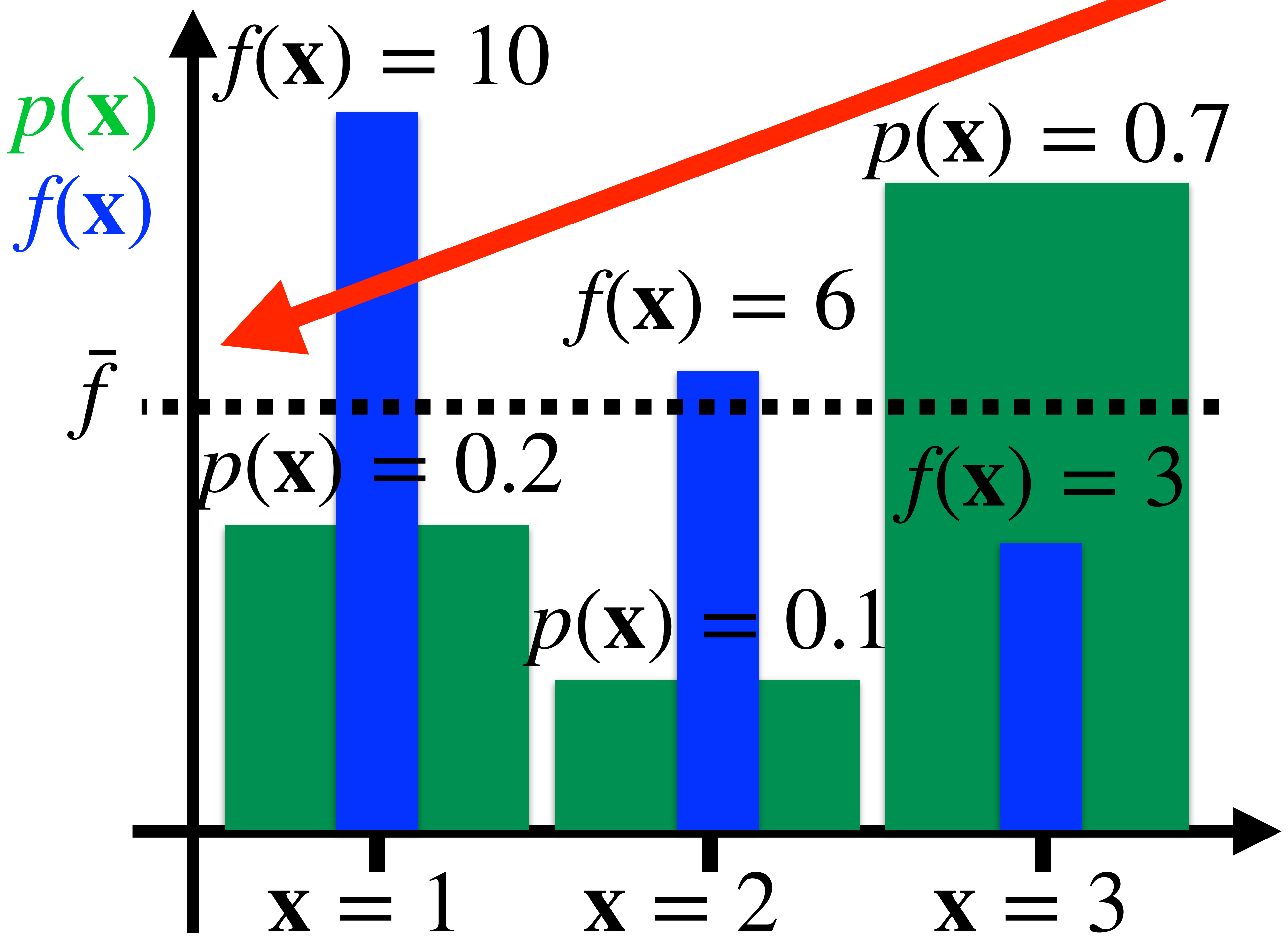


Prerequisites: Mean and average

$$\bar{f} = \sum_{\mathbf{x}} p(\mathbf{x}) \cdot f(\mathbf{x}) = \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}\left[f(\mathbf{x})\right] = 0.2 \cdot 10 \; + \; 0.1 \cdot 6 \; + \; 0.7 \cdot 3 = 4.7$$

$$\approx \frac{1}{N}\sum_i f(\mathbf{x}_i) = \frac{1}{10}(10 + 10 + 6 + 3 + 3 + 3 + 3 + 3 + 3 + 3) = 4.7$$

where $\mathbf{x}_i \sim p$



Prerequisites: Mean and average

Discrete:

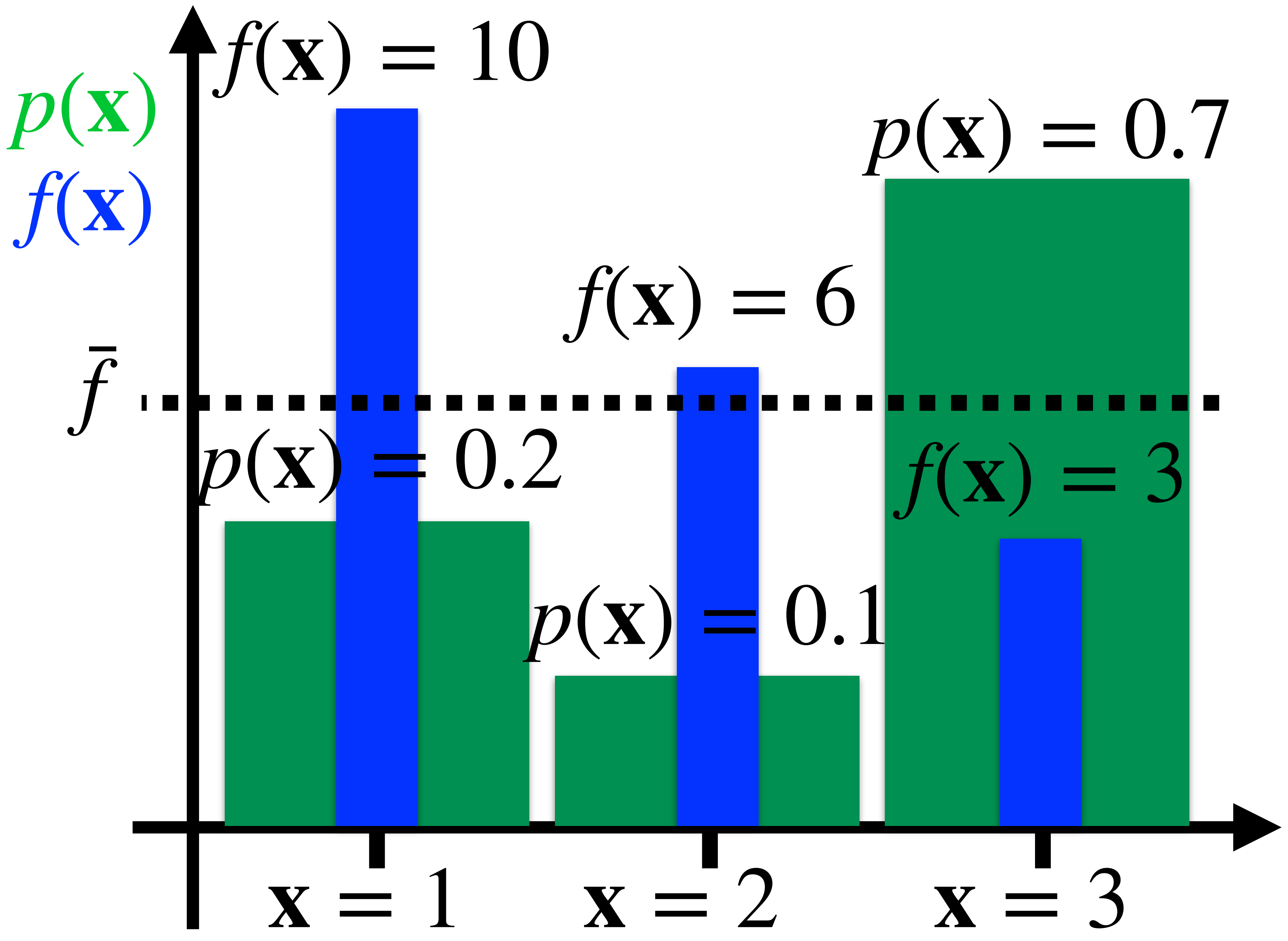
$$\bar{f} = \sum_{\mathbf{x}} p(\mathbf{x}) \cdot f(\mathbf{x}) = \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}\left[f(\mathbf{x})\right]$$

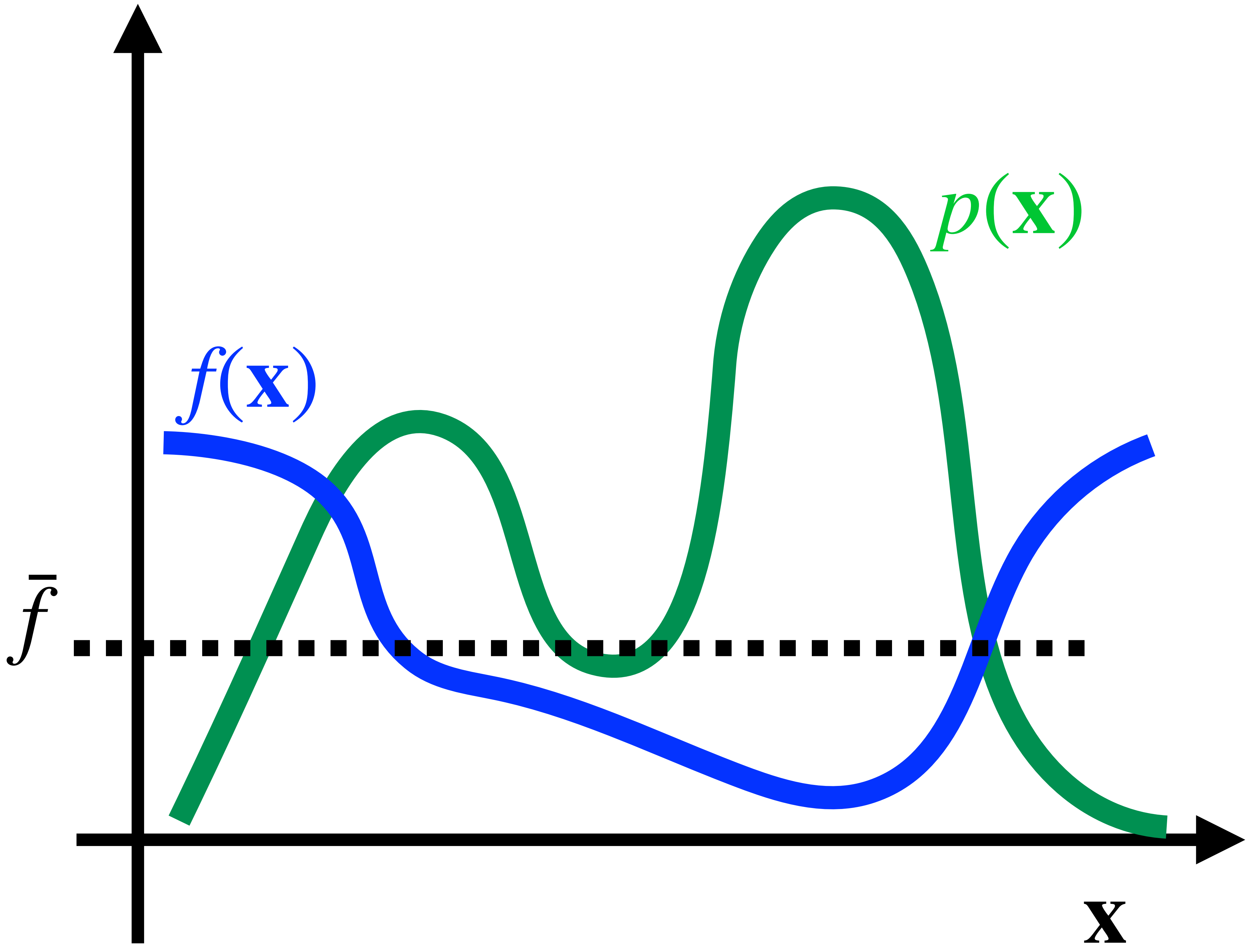
$$\approx \frac{1}{N}\sum_i f(\mathbf{x}_i) \quad \text{where } \mathbf{x}_i \sim p$$

Continuous:

$$\bar{f} = \int_{\mathbf{x}} p(\mathbf{x}) \cdot f(\mathbf{x})\, d\mathbf{x} = \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}\left[f(\mathbf{x})\right]$$

$$\approx \frac{1}{N}\sum_i f(\mathbf{x}_i) \quad \text{where } \mathbf{x}_i \sim p$$



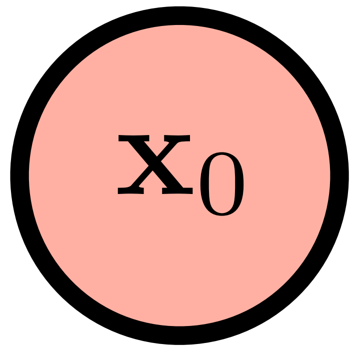


Problem definition

States: $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in \mathcal{R}^n$ sufficient description of the system

What does the state consists of?

- Robot’s pose/velocity (x,y,z, roll, pitch, yaw)
- Robot’s configuration (position and velocity of its joints)
- Pose/velocity and features of surrounding environment (walls, cars, people)
- State of battery, broken sensor/actuary
- Complete state - best predictor of the future (includes everything important from past)



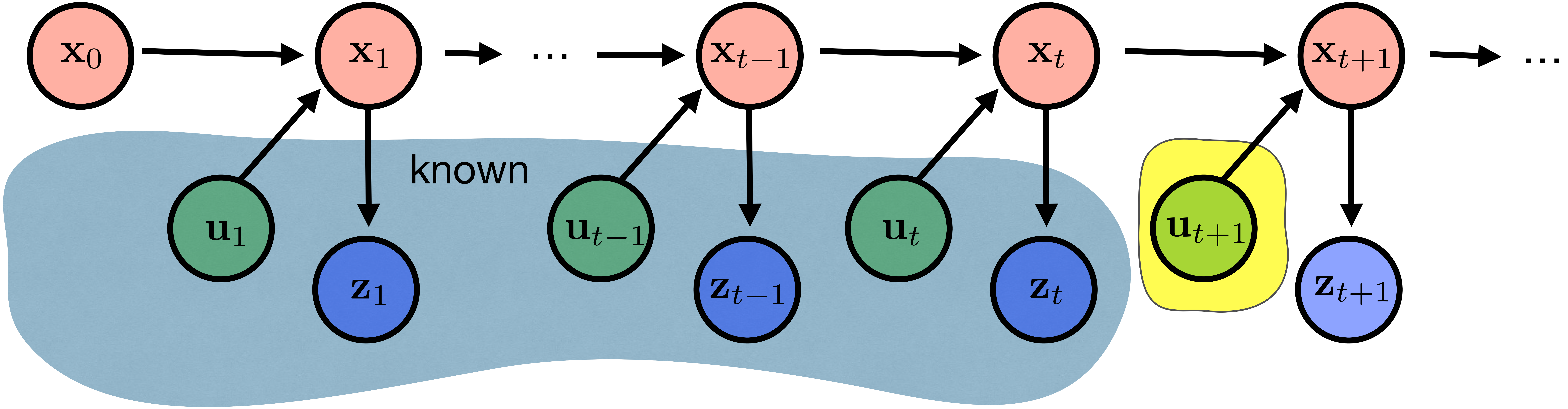
Problem definition

States: $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in \mathcal{R}^n$

Actions: $\mathbf{u}_1, \ldots, \mathbf{u}_t \in \mathcal{R}^m$

Measurements: $\mathbf{z}_1, \ldots, \mathbf{z}_t \in \mathcal{R}^k$

Algorithm: $\mathbf{u}_{t+1} = \pi(\mathbf{z}_{1:t}, \mathbf{u}_{1:t})$



Problem definition

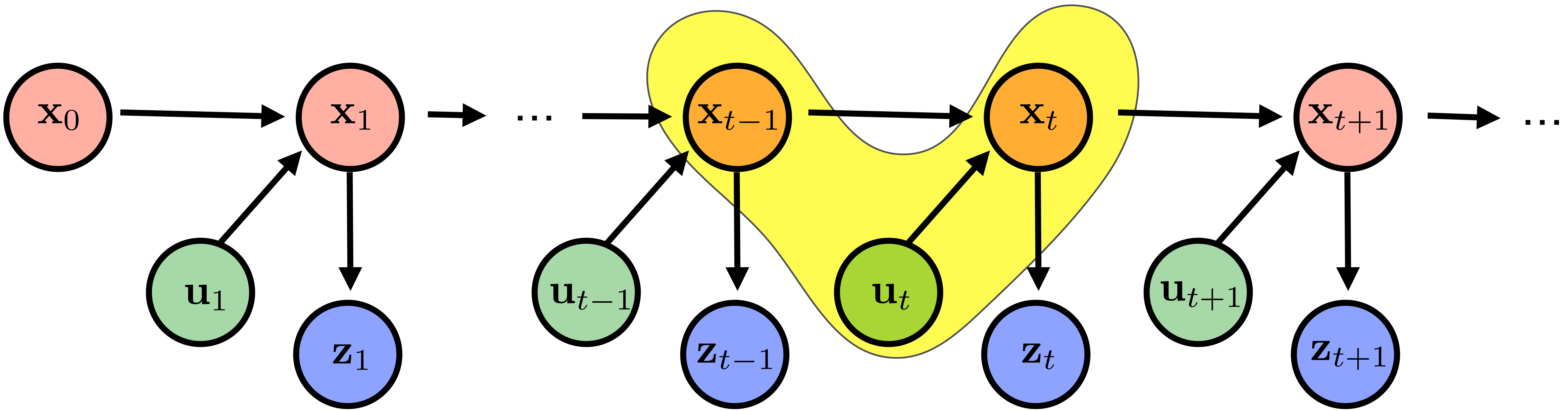
States: $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in \mathcal{R}^n$

Actions: $\mathbf{u}_1, \ldots, \mathbf{u}_t \in \mathcal{R}^m$

Measurements: $\mathbf{z}_1, \ldots, \mathbf{z}_t \in \mathcal{R}^k$

Algorithm: $\mathbf{u}_{t+1} = \pi(\mathbf{z}_{1:t}, \mathbf{u}_{1:t})$

Rewards: $r_t = r(\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{x}_t) \in \mathcal{R}$



Problem definition

States: $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in \mathcal{R}^n$

Actions: $\mathbf{u}_1, \ldots, \mathbf{u}_t \in \mathcal{R}^m$

Measurements: $\mathbf{z}_1, \ldots, \mathbf{z}_t \in \mathcal{R}^k$

Algorithm: $\mathbf{u}_{t+1} = \pi(\mathbf{z}_{1:t}, \mathbf{u}_{1:t})$

Rewards: $r_t = r(\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{x}_t) \in \mathcal{R}$

Problem definition

States: $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in \mathcal{R}^n$

Actions: $\mathbf{u}_1, \ldots, \mathbf{u}_t \in \mathcal{R}^m$

Measurements: $\mathbf{z}_1, \ldots, \mathbf{z}_t \in \mathcal{R}^k$

Algorithm: $\mathbf{u}_{t+1} = \pi(\mathbf{z}_{1:t}, \mathbf{u}_{1:t})$

Rewards: $r_t = r(\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{x}_t) \in \mathcal{R}$

Criterion: $J_\pi = \mathbb{E}_{\tau \sim \pi}\{\sum_{r_t \sim \tau} \gamma^t r_t\} \in \mathcal{R}$

Goal: $\pi^* = \arg\max_\pi J_\pi$

Problem definition

States: $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in \mathcal{R}^n$

Actions: $\mathbf{u}_1, \ldots, \mathbf{u}_t \in \mathcal{R}^m$

Measurements: $\mathbf{z}_1, \ldots, \mathbf{z}_t \in \mathcal{R}^k$

Algorithm: $\mathbf{u}_{t+1} = \pi(\mathbf{z}_{1:t}, \mathbf{u}_{1:t})$

Rewards: $r_t = r(\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{x}_t) \in \mathcal{R}$

Criterion: $J_\pi = \mathbb{E}_{\tau\sim\pi}\{\sum_{r_t\sim\tau} \gamma^t r_t\} \in \mathcal{R}$

Goal: $\pi^* = \arg\max_\pi J_\pi$

Which one is harder?

Algorithm: $\mathbf{z}_0, \mathbf{u}_1, \mathbf{z}_1, \ldots$ => estimate $p(\mathbf{x}_t | \mathbf{z}_{1:t}, \mathbf{u}_{1:t})$
perception (local, SLAM, object det.)

$\pi(\mathbf{x}_t)$ ➡

decide following action $\mathbf{u}_{t+1}$
control (planning, RL, MPC)

Problem definition

States: $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in \mathcal{R}^n$

Actions: $\mathbf{u}_1, \ldots, \mathbf{u}_t \in \mathcal{R}^m$

Measurements: $\mathbf{z}_1, \ldots, \mathbf{z}_t \in \mathcal{R}^k$

Algorithm: $\mathbf{u}_{t+1} = \pi(\mathbf{z}_{1:t}, \mathbf{u}_{1:t})$

Rewards: $r_t = r(\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{x}_t) \in \mathcal{R}$

Criterion: $J_\pi = \mathbb{E}_{\tau \sim \pi}\{\sum_{r_t \sim \tau} \gamma^t r_t\} \in \mathcal{R}$

Goal: $\pi^* = \arg\max_\pi \; J_\pi$

Examples:

Problem definition

States: $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in \mathcal{R}^n$

Actions: $\mathbf{u}_1, \ldots, \mathbf{u}_t \in \mathcal{R}^m$

Measurements: $\mathbf{z}_1, \ldots, \mathbf{z}_t \in \mathcal{R}^k$

Algorithm: $\mathbf{u}_{t+1} = \pi(\mathbf{z}_{1:t}, \mathbf{u}_{1:t})$

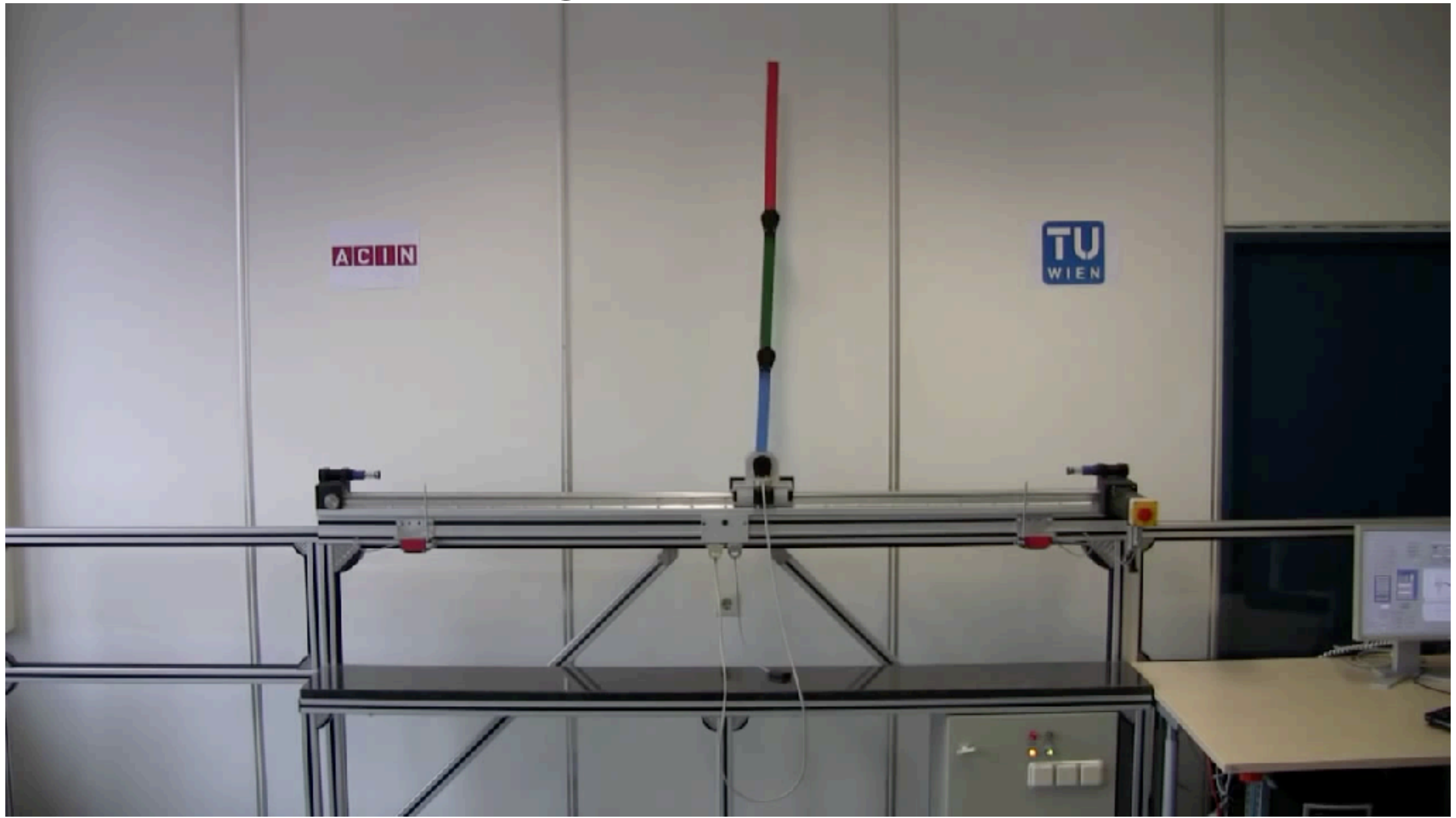
Rewards: $r_t = r(\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{x}_t) \in \mathcal{R}$

Criterion: $J_\pi = \mathbb{E}_{\tau \sim \pi}\{\sum_{r_t \sim \tau} \gamma^t r_t\} \in \mathcal{R}$

Goal: $\pi^* = \arg\max_\pi J_\pi$

- Balancing pendulum (z=x: joint angles/velocities/accelerations, u: base vel., r: height)

Examples:



Problem definition

States: $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in \mathcal{R}^n$

Actions: $\mathbf{u}_1, \ldots, \mathbf{u}_t \in \mathcal{R}^m$

Measurements: $\mathbf{z}_1, \ldots, \mathbf{z}_t \in \mathcal{R}^k$

Algorithm: $\mathbf{u}_{t+1} = \pi(\mathbf{z}_{1:t}, \mathbf{u}_{1:t})$

Rewards: $r_t = r(\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{x}_t) \in \mathcal{R}$

Criterion: $J_\pi = \mathbb{E}_{\tau \sim \pi}\{\sum_{r_t \sim \tau} \gamma^t r_t\} \in \mathcal{R}$

Goal: $\pi^* = \arg\max_\pi J_\pi$

Examples:

- Balancing pendulum (z=x: joint angles/velocities/accelerations, u: base vel., r: height)

Problem definition

States: $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in \mathcal{R}^n$

Actions: $\mathbf{u}_1, \ldots, \mathbf{u}_t \in \mathcal{R}^m$

Measurements: $\mathbf{z}_1, \ldots, \mathbf{z}_t \in \mathcal{R}^k$

Algorithm: $\mathbf{u}_{t+1} = \pi(\mathbf{z}_{1:t}, \mathbf{u}_{1:t})$

Rewards: $r_t = r(\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{x}_t) \in \mathcal{R}$

Criterion: $J_\pi = \mathbb{E}_{\tau\sim\pi}\{\sum_{r_t\sim\tau} \gamma^t r_t\} \in \mathcal{R}$

Goal: $\pi^* = \arg\max_\pi J_\pi$

- RC helicopter tricks (z: body pose+angles/vel/acc, u: motor torques, r: ???)

Examples:

Problem definition

States: $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in \mathcal{R}^n$

Actions: $\mathbf{u}_1, \ldots, \mathbf{u}_t \in \mathcal{R}^m$

Measurements: $\mathbf{z}_1, \ldots, \mathbf{z}_t \in \mathcal{R}^k$

Algorithm: $\mathbf{u}_{t+1} = \pi(\mathbf{z}_{1:t}, \mathbf{u}_{1:t})$

Rewards: $r_t = r(\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{x}_t) \in \mathcal{R}$

Criterion: $J_\pi = \mathbb{E}_{\tau \sim \pi}\{\sum_{r_t \sim \tau} \gamma^t r_t\} \in \mathcal{R}$

Goal: $\pi^* = \arg\max_\pi J_\pi$

Examples:

- Balancing pendulum (z=x: joint angles/velocities/accelerations, u: base vel., r: height)
- RC helicopter tricks (z: body pose+angles/vel/acc, u: motor torques, r: ???)

Problem definition

States: $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in \mathcal{R}^n$

Actions: $\mathbf{u}_1, \ldots, \mathbf{u}_t \in \mathcal{R}^m$

Measurements: $\mathbf{z}_1, \ldots, \mathbf{z}_t \in \mathcal{R}^k$

Algorithm: $\mathbf{u}_{t+1} = \pi(\mathbf{z}_{1:t}, \mathbf{u}_{1:t})$

Rewards: $r_t = r(\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{x}_t) \in \mathcal{R}$

Criterion: $J_\pi = \mathbb{E}_{\tau \sim \pi}\{\sum_{r_t \sim \tau} \gamma^t r_t\} \in \mathcal{R}$

Goal: $\pi^* = \arg\max_\pi J_\pi$

- Agent playing Atari games (z: RGB, u: joystick, r: score/lives, x: ???)

Examples:

Problem definition

States: $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in \mathcal{R}^n$

Actions: $\mathbf{u}_1, \ldots, \mathbf{u}_t \in \mathcal{R}^m$

Measurements: $\mathbf{z}_1, \ldots, \mathbf{z}_t \in \mathcal{R}^k$

Algorithm: $\mathbf{u}_{t+1} = \pi(\mathbf{z}_{1:t}, \mathbf{u}_{1:t})$

Rewards: $r_t = r(\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{x}_t) \in \mathcal{R}$

Criterion: $J_\pi = \mathbb{E}_{\tau \sim \pi}\{\sum_{r_t \sim \tau} \gamma^t r_t\} \in \mathcal{R}$

Goal: $\pi^* = \arg\max_\pi J_\pi$

Examples:

- Balancing pendulum (z=x: joint angles/velocities/accelerations, u: base vel., r: height)
- RC helicopter tricks (z: body pose+angles/vel/acc, u: motor torques, r: ???)
- Agent playing Atari games (z: RGB, u: joystick, r: score/lives, x: ???)

Problem definition

States: $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in \mathcal{R}^n$

Actions: $\mathbf{u}_1, \ldots, \mathbf{u}_t \in \mathcal{R}^m$

Measurements: $\mathbf{z}_1, \ldots, \mathbf{z}_t \in \mathcal{R}^k$

Algorithm: $\mathbf{u}_{t+1} = \pi(\mathbf{z}_{1:t}, \mathbf{u}_{1:t})$

Rewards: $r_t = r(\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{x}_t) \in \mathcal{R}$

Criterion: $J_\pi = \mathbb{E}_{\tau\sim\pi}\{\sum_{r_t\sim\tau} \gamma^t r_t\} \in \mathcal{R}$

Goal: $\pi^* = \arg\max_\pi J_\pi$

Examples:

x: ???

z:

Problem definition

States: $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in \mathcal{R}^n$

Actions: $\mathbf{u}_1, \ldots, \mathbf{u}_t \in \mathcal{R}^m$

Measurements: $\mathbf{z}_1, \ldots, \mathbf{z}_t \in \mathcal{R}^k$

Algorithm: $\mathbf{u}_{t+1} = \pi(\mathbf{z}_{1:t}, \mathbf{u}_{1:t})$

Rewards: $r_t = r(\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{x}_t) \in \mathcal{R}$

Criterion: $J_\pi = \mathbb{E}_{\tau \sim \pi}\{\sum_{r_t \sim \tau} \gamma^t r_t\} \in \mathcal{R}$

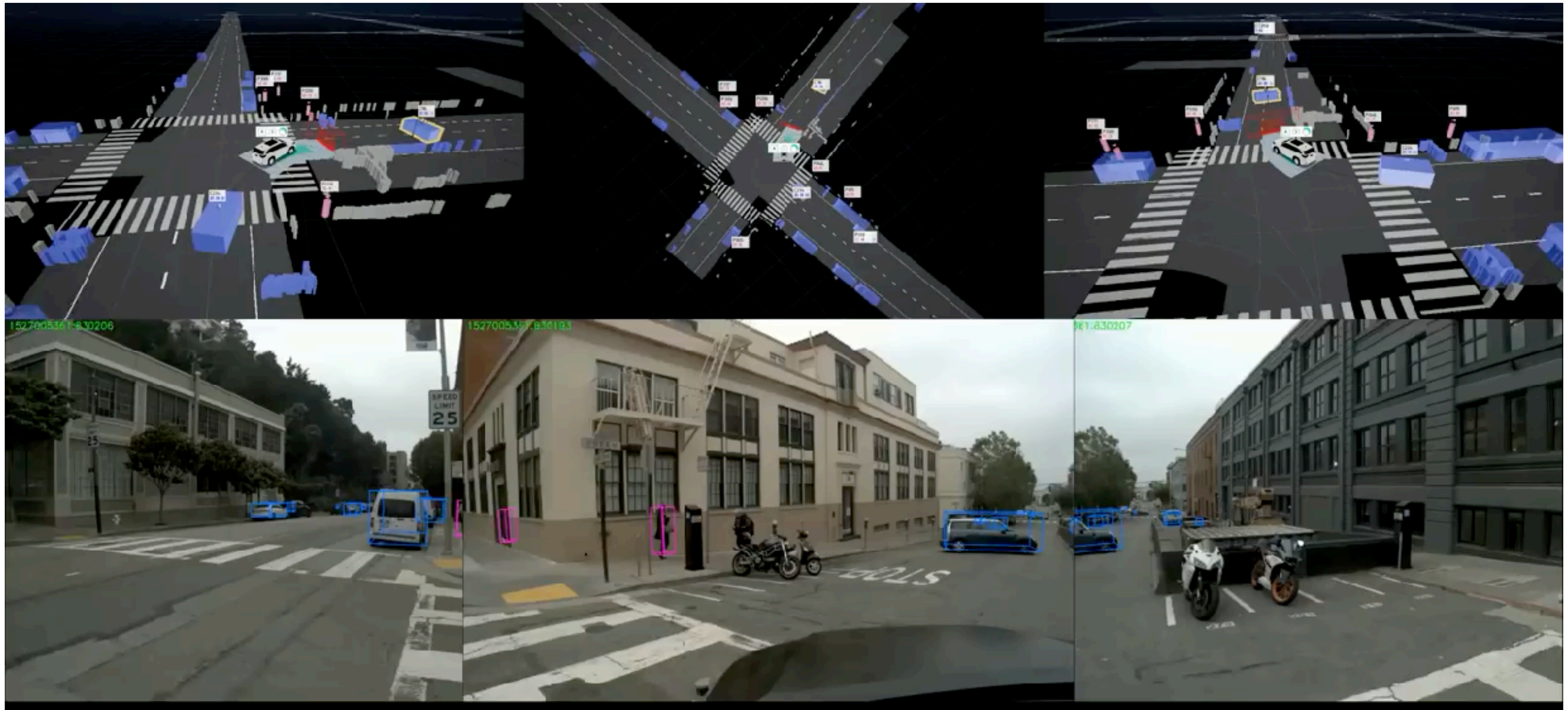
Goal: $\pi^* = \arg\max_\pi J_\pi$

- Autonomous car (z: RGB+lidar+IMU, u: wheel+throttle, r: reach goal+survive, x: ???)

Examples:

x:

z:



Problem definition

States: $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in \mathcal{R}^n$

Actions: $\mathbf{u}_1, \ldots, \mathbf{u}_t \in \mathcal{R}^m$

Measurements: $\mathbf{z}_1, \ldots, \mathbf{z}_t \in \mathcal{R}^k$

Algorithm: $\mathbf{u}_{t+1} = \pi(\mathbf{z}_{1:t}, \mathbf{u}_{1:t})$

Rewards: $r_t = r(\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{x}_t) \in \mathcal{R}$

Criterion: $J_\pi = \mathbb{E}_{\tau\sim\pi}\{\sum_{r_t\sim\tau} \gamma^t r_t\} \in \mathcal{R}$

Goal: $\pi^* = \arg\max_\pi J_\pi$

Examples:

- Balancing pendulum (z=x: joint angles/velocities/accelerations, u: base vel., r: height)
- RC helicopter tricks (z: body pose+angles/vel/acc, u: motor torques, r: ???)
- Agent playing Atari games (z: RGB, u: joystick, r: score/lives, x: ???)
- Autonomous car (z: RGB+lidar+IMU, u: wheel+throttle, r: reach goal+survive, x: ???)

Problem definition

States: $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in \mathcal{R}^n$

Actions: $\mathbf{u}_1, \ldots, \mathbf{u}_t \in \mathcal{R}^m$

Measurements: $\mathbf{z}_1, \ldots, \mathbf{z}_t \in \mathcal{R}^k$

Algorithm: $\mathbf{u}_{t+1} = \pi(\mathbf{z}_{1:t}, \mathbf{u}_{1:t})$

Rewards: $r_t = r(\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{x}_t) \in \mathcal{R}$

Criterion: $J_\pi = \mathbb{E}_{\tau\sim\pi}\{\sum_{r_t\sim\tau} \gamma^t r_t\} \in \mathcal{R}$

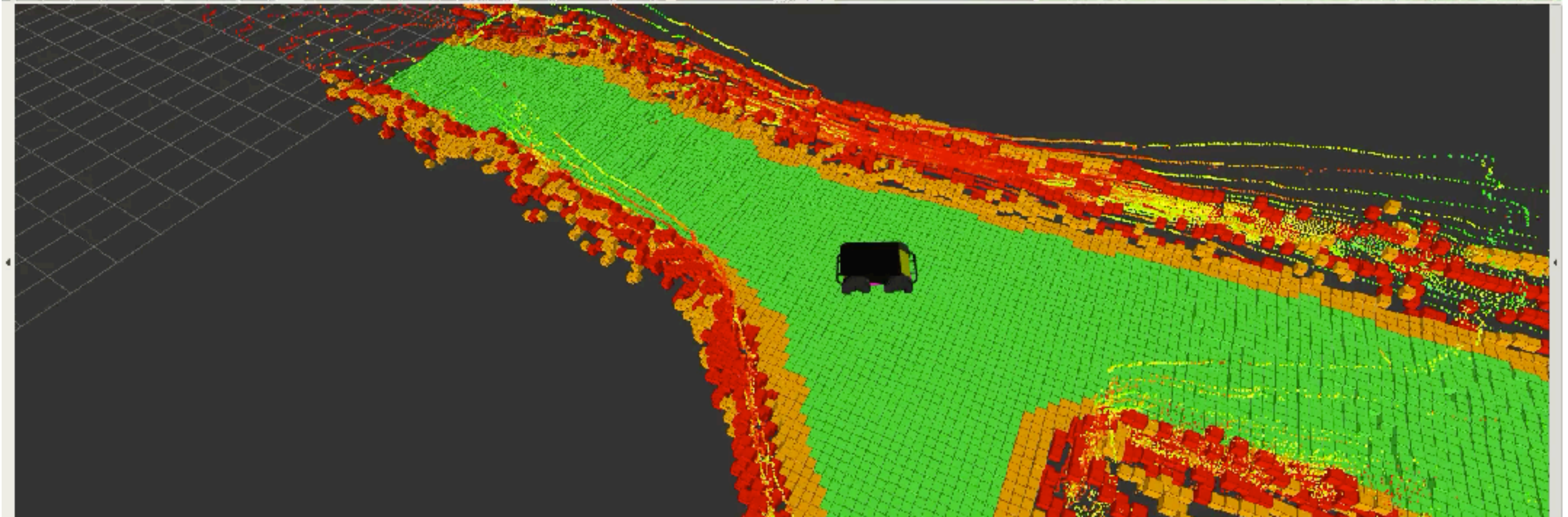
Goal: $\pi^* = \arg\max_\pi J_\pi$

- DARPA exploration scenario (x: successively constructed map+pose)

Examples:

z:

x:



Problem definition

States: $\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_t \in \mathcal{R}^n$

Actions: $\mathbf{u}_1, \dots, \mathbf{u}_t \in \mathcal{R}^m$

Measurements: $\mathbf{z}_1, \dots, \mathbf{z}_t \in \mathcal{R}^k$

Algorithm: $\mathbf{u}_{t+1} = \pi(\mathbf{z}_{1:t}, \mathbf{u}_{1:t})$

Rewards: $r_t = r(\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{x}_t) \in \mathcal{R}$

Criterion: $J_\pi = \mathbb{E}_{\tau\sim\pi}\{\sum_{r_t\sim\tau} \gamma^t r_t\} \in \mathcal{R}$

Goal: $\pi^* = \arg\max_\pi \, J_\pi$

Examples:

- Balancing pendulum (z=x: joint angles/velocities/accelerations, u: base vel., r: height)
- RC helicopter tricks (z: body pose+angles/vel/acc, u: motor torques, r: ???)
- Agent playing Atari games (z: RGB, u: joystick, r: score/lives, x: ???)
- Autonomous car (z: RGB+lidar+IMU, u: wheel+throttle, r: reach goal+survive, x: ???)
- DARPA exploration scenario (x: successively constructed map+pose)

Problem definition

States: $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in \mathcal{R}^n$

Actions: $\mathbf{u}_1, \ldots, \mathbf{u}_t \in \mathcal{R}^m$

Measurements: $\mathbf{z}_1, \ldots, \mathbf{z}_t \in \mathcal{R}^k$

Algorithm: $\mathbf{u}_{t+1} = \pi(\mathbf{z}_{1:t}, \mathbf{u}_{1:t})$

Rewards: $r_t = r(\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{x}_t) \in \mathcal{R}$

Criterion: $J_\pi = \mathbb{E}_{\tau\sim\pi}\{\sum_{r_t\sim\tau} \gamma^t r_t\} \in \mathcal{R}$

Goal: $\pi^* = \arg\max_\pi J_\pi$

- Autonomous robotic warehouse (x=z: all robots and packages, r: -avg delivery time)

Examples:

Problem definition

States: $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t \in \mathcal{R}^n$

Actions: $\mathbf{u}_1, \ldots, \mathbf{u}_t \in \mathcal{R}^m$

Measurements: $\mathbf{z}_1, \ldots, \mathbf{z}_t \in \mathcal{R}^k$

Algorithm: $\mathbf{u}_{t+1} = \pi(\mathbf{z}_{1:t}, \mathbf{u}_{1:t})$

Rewards: $r_t = r(\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{x}_t) \in \mathcal{R}$

Criterion: $J_\pi = \mathbb{E}_{\tau\sim\pi}\{\sum_{r_t\sim\tau} \gamma^t r_t\} \in \mathcal{R}$

Goal: $\pi^* = \arg\max_\pi J_\pi$

Examples:

- Balancing pendulum (z=x: joint angles/velocities/accelerations, u: base vel., r: height)
- RC helicopter tricks (z: body pose+angles/vel/acc, u: motor torques, r: ???)
- Agent playing Atari games (z: RGB, u: joystick, r: score/lives, x: ???)
- Autonomous car (z: RGB+lidar+IMU, u: wheel+throttle, r: reach goal+survive, x: ???)
- DARPA exploration scenario (x: successively constructed map+pose)
- Autonomous robotic warehouse (x=z: all robots and packages, r: -avg delivery time)
- Exotic tasks also covered: e.g. Active SLAM (r: accurate state estimate)

Summary

- State, action, measurements, reward, cost, criterion,
- Goal is policy/algorithm/pipeline/regulator/controller that optimise criterion
- Two subproblems:
  - (1) Perception (approx. 6 KZ lectures)
  - (2) motion planning/control (approx 6 VV lectures)
- Further reading Probabilistic Robotics book (section 1.1 - 2.3) https://docs.ufpr.br/~danielsantos/ProbabilisticRobotics.pdf
- Next lecture: Localization