

# L1. Probability review

## **Outline**

- 1. Probability**
- 2. Discrete distributions**
- 3. Continuous distributions**

## **Outline**

- 1. Probability**
2. Discrete distributions
3. Continuous distributions

## Probability as a measure of uncertainty

- Probability : A numerical measure of uncertainty

$Y$  : Random variable

$y$  : Data assigned to random variable  $Y$

$p_Y(Y = y) = p(y)$  :Short notation

$p(y)$     if  $y$  is continuous,  $p(y)$  is probability density function (PDF)  
              if  $y$  is discrete,  $p(y)$  is probability mass function (PMF)

We going to use  $p(y)$  to represent both PDF and PMF

$Y \sim \text{dist}(\theta)$ ,  $\theta$  is parameter for distribution

$p_Y(Y = y|\theta) = p(y|\theta) = \text{dist}(y|\theta)$     or     $p_Y(Y = y; \theta) = p(y; \theta) = \text{dist}(y; \theta)$

Example :

$$Y \sim N(\mu, \sigma^2) \rightarrow p(y|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(\mu-y)^2}{2\sigma^2}}$$

$$Y \sim \text{Bin}(n, \theta) \rightarrow p(y|\theta) = \binom{n}{y} \theta^y (1-\theta)^{n-y}$$

## Useful results from Probability Theory for Bayesian Statistics

$$p(u, v)$$

$$p(u|v) = \frac{p(u, v)}{p(v)}$$

$$p(v) = \int_u p(u, v) du$$

$$p(u|v) = \frac{p(v|u)p(u)}{\int_u p(u, v) du}$$

$$p(u, v) = p(u|v) p(v)$$

$$p(u, v, w) = p(u|v, w)p(v|w)p(w)$$

$$p(u) = \int_v p(u|v)p(v) dv, \quad p(u) = \sum_v p(u|v)p(v)$$

## Mean and Variance of conditional distributions

$$\mathbb{E}(U) = \int up(u)du$$

$$\mathbb{E}(g(U)) = \int g(u)p(u)du$$

$$\text{var}(U) = \int (u - \mathbb{E}(U))^2 p(u)du$$

- $\mathbb{E}(U) = \mathbb{E}_V(\mathbb{E}(U|V))$

$$\because \mathbb{E}(U) = \int \int up(u, v)dudv = \int \int up(u|v)du \color{blue}{p(v)dv} = \int \mathbb{E}(U|v)p(v)dv = \mathbb{E}_V(\mathbb{E}(U|V))$$

- $\text{var}(U) = \mathbb{E}(\text{var}(U|V)) + \text{var}(\mathbb{E}(U|V))$

$$\text{var}(x) = \mathbb{E}(x^2) - (\mathbb{E}(x))^2$$

$$\begin{aligned}\because \mathbb{E}_V(\text{var}(U|V)) + \text{var}(\mathbb{E}(U|V)) &= \mathbb{E}_V(\mathbb{E}(u^2|v) - (\mathbb{E}(u|v))^2) + \mathbb{E}_v((\mathbb{E}(u|v))^2) - (\mathbb{E}_v(\mathbb{E}(u|v)))^2 \\ &= \mathbb{E}_v(u^2) - \mathbb{E}_v((\mathbb{E}(u|v))^2) + \mathbb{E}_v((\mathbb{E}(u|v))^2) - (\mathbb{E}(u))^2 \\ &= \mathbb{E}(u^2) - (\mathbb{E}(u))^2 \\ &= \text{var}(u)\end{aligned}$$

## Mean and Variance of conditional distributions (scalar case)

$$\mathbb{E}(U) = \int up(u)du$$

$$\mathbb{E}(g(U)) = \int g(u)p(u)du$$

$$\text{var}(U) = \int (u - E(U))^2 p(u)du$$

- $\mathbb{E}(U) = \mathbb{E}_V(\mathbb{E}(U|V))$

$$\begin{aligned} \because \mathbb{E}(U) &= \int \int up(u, v)dudv = \int \int up(u|v)du p(v)dv = \int \mathbb{E}(U|v)p(v)dv = \mathbb{E}_V(\mathbb{E}(U|V)) \\ &\quad p(u, v) = p(u|v)p(v) \end{aligned}$$

- $\text{var}(U) = \mathbb{E}(\text{var}(U|V)) + \text{var}(\mathbb{E}(U|V))$

$$\text{var}(X) = \mathbb{E}(X^2) - (\mathbb{E}(X))^2$$

$$\begin{aligned} \because \mathbb{E}_V(\text{var}(U|V)) + \text{var}(\mathbb{E}(U|V)) &= \mathbb{E}_V(\mathbb{E}(U^2|V) - (\mathbb{E}(U|V))^2) + \mathbb{E}_V((\mathbb{E}(U|V))^2) - (\mathbb{E}_V(\mathbb{E}(U|V)))^2 \\ &= \mathbb{E}_V(U^2) - \mathbb{E}_V((\mathbb{E}(U|V))^2) + \mathbb{E}_V((\mathbb{E}(U|V))^2) - (\mathbb{E}(U))^2 \\ &= \mathbb{E}(U^2) - (\mathbb{E}(U))^2 \\ &= \text{var}(U) \end{aligned}$$

## Example of Conditional Probability

$$p(u|v) = \frac{p(u,v)}{p(v)} = \frac{p(v|u)p(u)}{\int_v p(u,v) dv}$$

### Given probability

$$p(C) = \frac{1}{100} \rightarrow p(NC) = \frac{99}{100}$$

$$p(+|C) = \frac{90}{100} \rightarrow p(-|C) = \frac{10}{100}$$

$$p(+|NC) = \frac{8}{100} \rightarrow p(-|NC) = \frac{92}{100}$$

$p(C)$  : Probability of having cancer

$p(+|C)$  : Positive result given cancer

$p(+|NC)$  : Positive result given no cancer

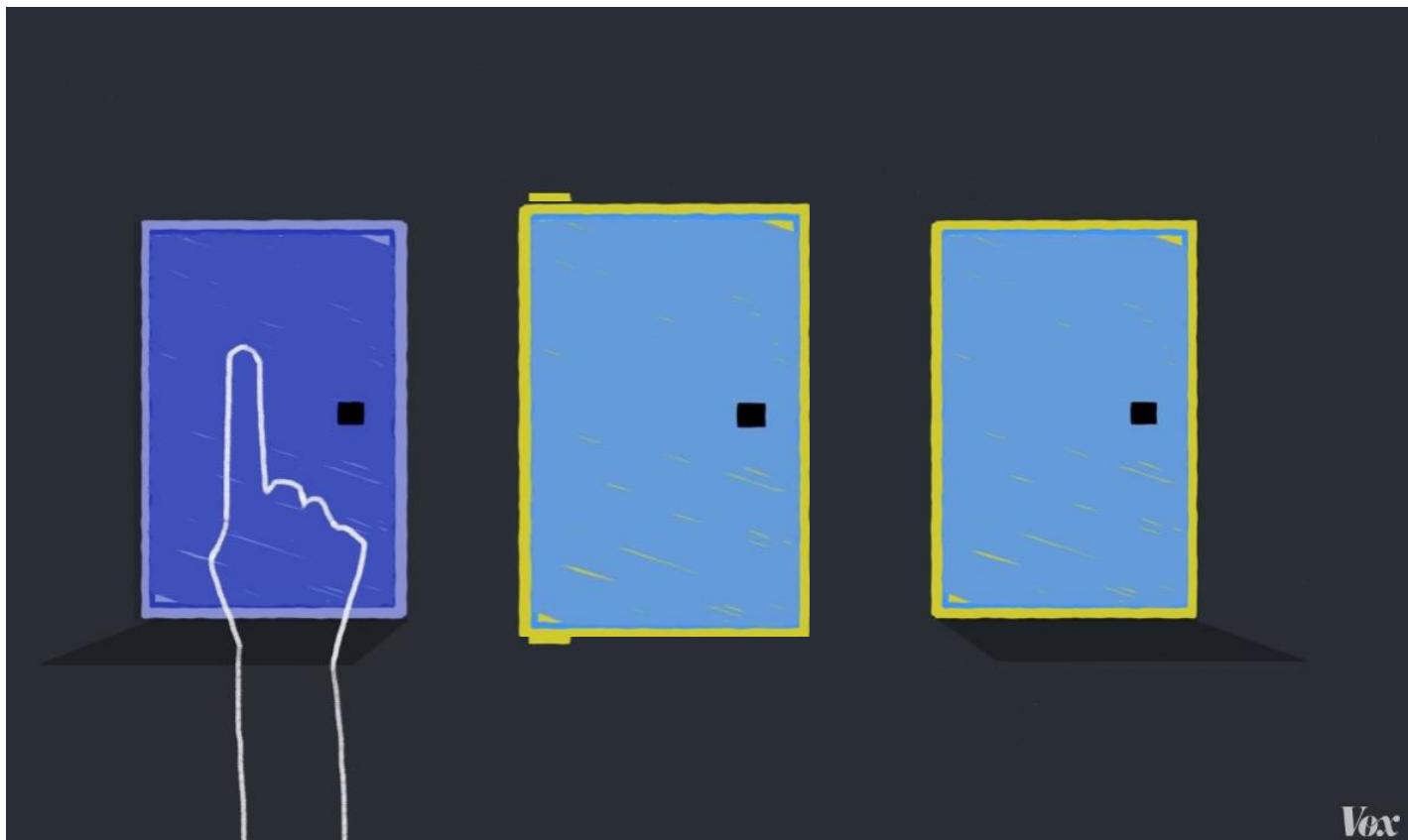
$$\begin{aligned} p(C|+) &= \frac{p(C,+)}{p(+)} = \frac{p(+|C)p(C)}{p(+)} \\ &= \frac{p(+|C)p(C)}{p(+|C)p(C) + p(+|NC)p(NC)} \\ &= \frac{\frac{90}{100} \times \frac{1}{100}}{\frac{90}{100} \times \frac{1}{100} + \frac{8}{100} \times \frac{99}{100}} \\ &\approx 0.1 \end{aligned}$$

Isn't it too low?

## Example of Conditional Probability



<https://www.youtube.com/watch?v=ggDQXlinbME>

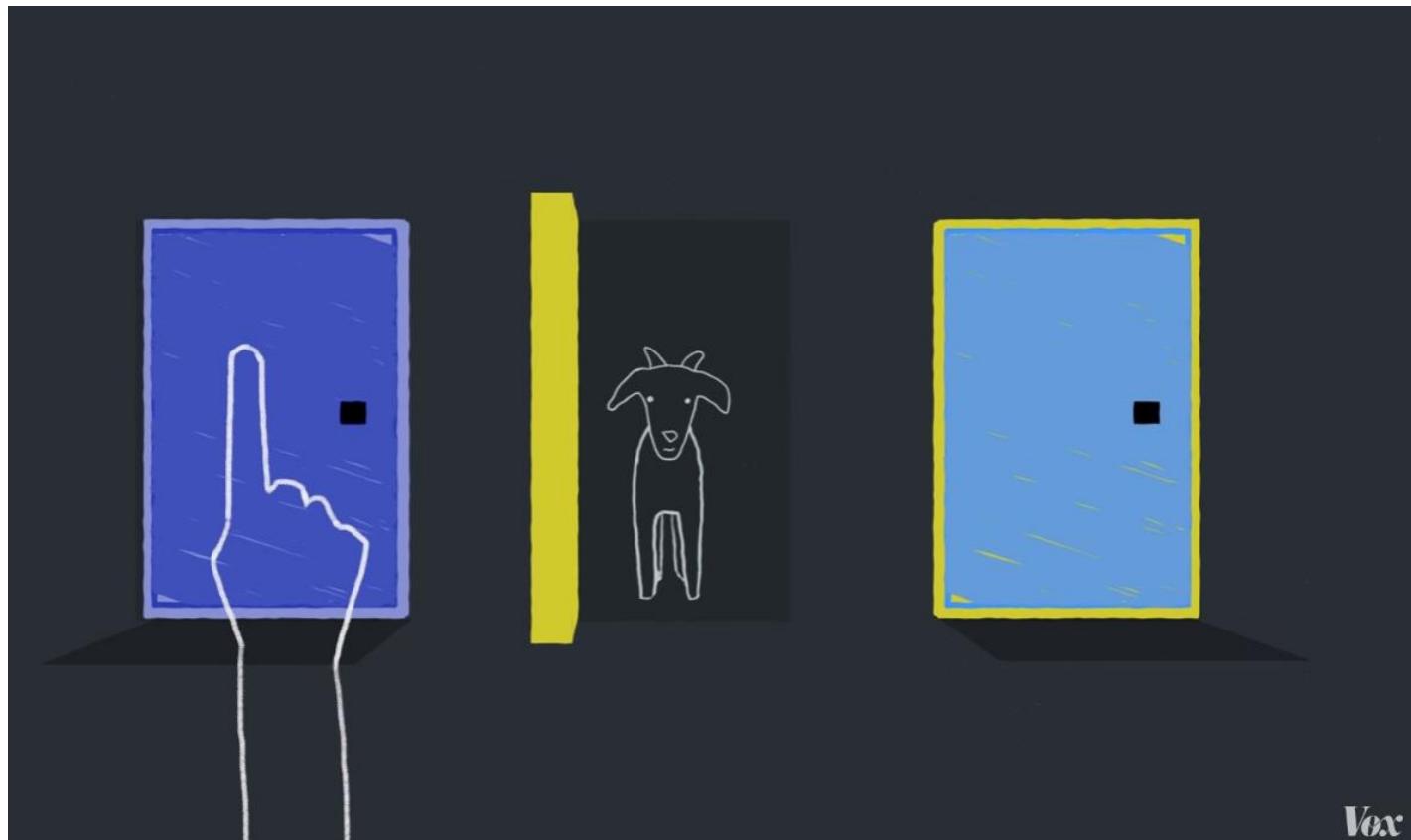


Vox

## Example of Conditional Probability



<https://www.youtube.com/watch?v=ggDQXlinbME>

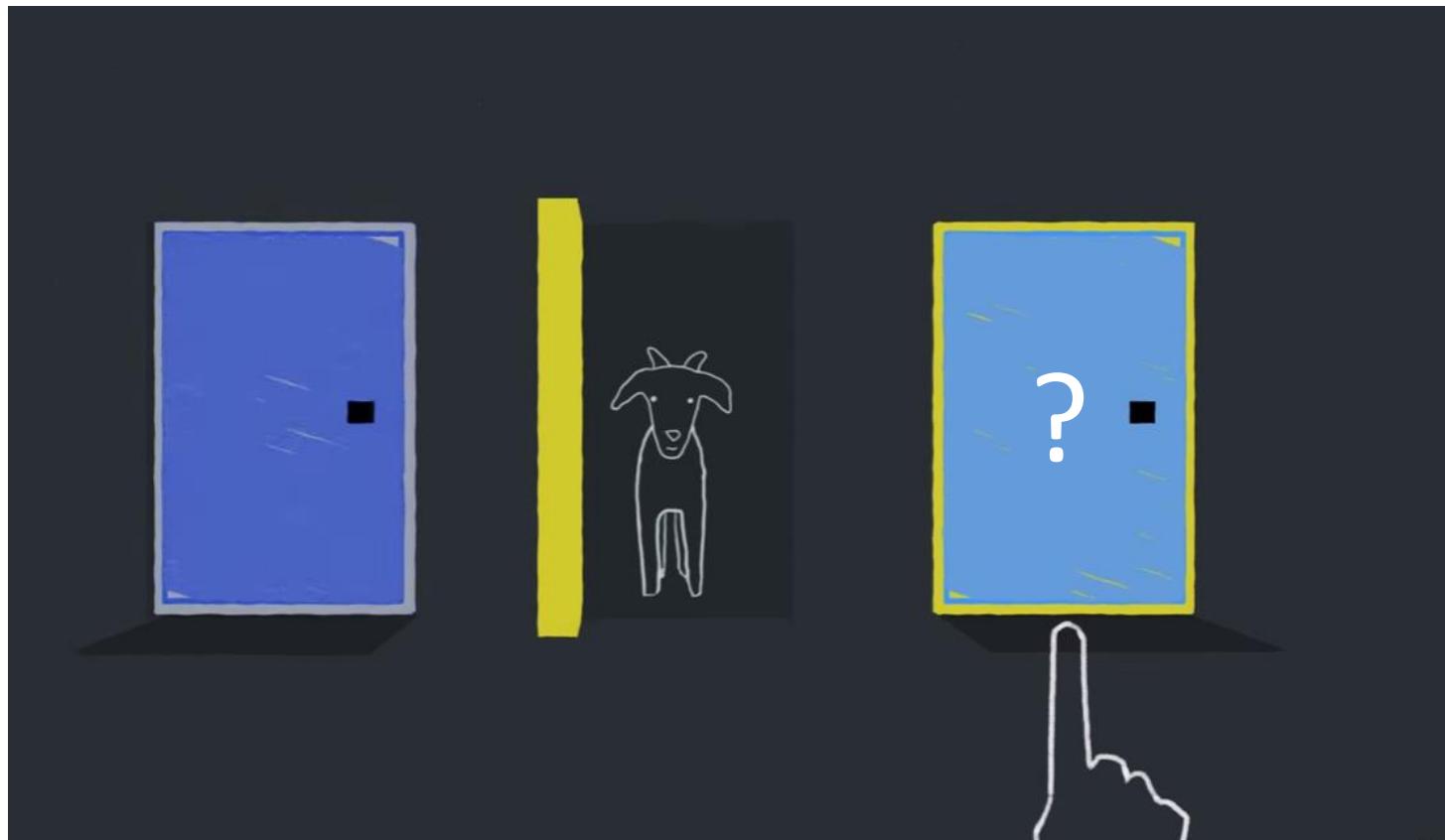


Vox

## Example of Conditional Probability



<https://www.youtube.com/watch?v=ggDQXlinbME>



## Example of Conditional Probability

Assume we picked up Door 1 and then Monty shows us a goat behind Door 2

Event A: The car is behind Door 1 and

Event B: The Monty shows us a goat behind Door 2 (observation)

## Example of Conditional Probability

Assume we picked up Door 1 and then Monty shows us a goat behind Door 2

Event A: The car is behind Door 1 and

Event B: The Monty shows us a goat behind Door 2

$$\begin{aligned} P(A|B) &= \frac{P(B|A)P(A)}{P(B)} \\ &= \frac{P(B|A)P(A)}{P(B|A)P(A) + P(B|\sim A)P(\sim A)} && \because P(B) = \sum_a P(B|A=a)P(A=a) \\ &= \frac{P(B|A)P(A)}{P(B|A)P(A) + P(B|\text{Car bh D2})P(\text{Car bh D2}) + P(B|\text{Car bh D3})P(\text{Car bh D3})} \\ &= \frac{\frac{1}{2} \times \frac{1}{3}}{\frac{1}{2} \times \frac{1}{3} + 0 \times \frac{1}{3} + 1 \times \frac{1}{3}} = \frac{1}{3} \end{aligned}$$

$$P(\sim A|B) = \frac{2}{3} \quad : \text{Probability that the goat is behind the door 3}$$

Changing the door will double the probability of getting the car!

## **Outline**

- 1. Probability**
- 2. Discrete distributions**
- 3. Continuous distributions**

## The Bernoulli distribution

The probability distribution of a random variable which takes value 1 with success probability and value 0 with failure probability .

- **Notation**

$$Y \sim B(p)$$

$$p_Y(Y = y) = p(y) = B(y|p) = p^y(1 - p)^{1-y} \quad \text{for } y \in \{0,1\}$$

- **Parameters**

$p \in [0,1]$  : Probability of success

- **Mean and variance**

$$\text{E}(Y) = p$$

$$\text{var}(Y) = p(1 - p)$$

## The Binomial Distribution

The distribution on the number of successes in a sequence of  $n$  independent yes/no experiments

- The total number of heads among  $n$  coin tossing

- Notation

$$Y \sim \text{Bin}(n, p)$$

$$p(y) = \text{Bin}(y|n, p) = \binom{n}{y} p^y (1-p)^{n-y} \quad y \in \{0, 1, \dots, n\}:$$

- Parameters

$n > 0$  : Sample size (number of trials)

$p \in [0, 1]$  : Probability of success

- Mean and variance

$$\text{E}(Y) = np$$

$$\text{var}(Y) = np(1-p)$$

$n = 1$ , the binomial distribution is a Bernoulli distribution

## The Poisson Distribution

Expresses the probability of a given number of events occurring in a fixed interval of time or space if these events occur with a known average rate (lambda)

- number of birth per hour on a given day in hospital
- disease cases within a given town

- Notation

$$Y \sim \text{Poisson}(\lambda)$$

$$P(y) = \text{Poisson}(y|\lambda) = \frac{1}{y!} \lambda^y \exp(-\lambda)$$

$$y \in \{0, 1, 2, \dots\},$$

- Parameters

$$\lambda > 0 : \text{rate}$$

- Mean and variance

$$\text{E}(Y) = \lambda$$

$$\text{var}(Y) = \lambda$$

## The Poisson Distribution

$$\sum_y p(y|\lambda) = \sum_y \frac{\lambda^y e^{-\lambda}}{y!} = e^{-\lambda} \left[ 1 + \lambda + \frac{\lambda^2}{2!} + \frac{\lambda^2}{2!} + \dots \right] = e^{-\lambda} e^\lambda = 1$$

$$E[Y] = \sum_y y \frac{\lambda^y e^{-\lambda}}{y!} = e^{-\lambda} \left[ \lambda + \lambda^2 + \frac{\lambda^3}{2!} + \frac{\lambda^4}{3!} + \dots \right] = e^{-\lambda} \lambda \left[ 1 + \lambda + \frac{\lambda^2}{2!} + \frac{\lambda^2}{2!} + \dots \right] = \lambda$$

## The Geometric Distribution

The probability distribution of the number X of Bernoulli trials needed to get one success

- **Notation**

$$Y \sim \text{Geometric}(\lambda)$$

$$p(y) = \text{Geometric}(y|\lambda) = (1 - \lambda)^{y-1} \lambda$$

$$y \in \{0, 1, 2, \dots\},$$

- **Parameters**

$\lambda > 0$  : Probability of success

- **Mean and variance**

$$E(Y) = \frac{1}{\lambda}$$

$$\text{var}(Y) = \frac{1 - \lambda}{\lambda^2}$$

## The Multinomial Distribution

- **Notation**

$$Y = (Y_1, \dots, Y_k) \sim \text{Multin}(n, p_1, \dots, p_k) \quad y = (y_1, \dots, y_k) : \text{Vector!}$$

$$p(y) = \text{Multin}(y|n, p_1, \dots, p_k) = \binom{n}{y_1 \ y_2 \ \dots \ y_k} p_1^{y_1} \cdots p_k^{y_k}$$

$$y_j \in \{0, 1, \dots, n\}, \quad \sum_j^k y_j = n$$

- **Parameters**

$n > 0$  : Sample size (number of trials)

$p_j \in [0, 1], \sum_j^k p_j = 1$  : Probability of occurrence

- **Mean and variance**

$$\mathbb{E}(Y_j) = np_j$$

$$\text{var}(Y_j) = np_j(1 - p_j)$$

$$\text{cov}(Y_i, Y_j) = -np_i p_j$$

## **Outline**

- 1. Probability**
- 2. Discrete distributions**
- 3. Continuous distributions**

## The Uniform Distribution

- **Notation**

$$Y \sim U(\alpha, \beta)$$

$$p(y) = U(y|\alpha, \beta) = \frac{1}{\beta - \alpha}$$

$$y \in [\alpha, \beta]$$

- **Parameters**

$\alpha, \beta$  ( $\beta > \alpha$ ) : Boundaries

- **Mean and variance**

$$E(Y) = \frac{\alpha + \beta}{2}$$

$$\text{var}(Y) = \frac{(\beta - \alpha)^2}{12}$$

## The Univariate Normal Distribution

- **Notation**

$$Y \sim N(\mu, \sigma^2)$$

$$p(y) = N(y|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2\sigma^2}(y - \mu)^2\right)$$
$$y \in [-\infty, \infty]$$

- **Parameters**

$\mu$  : Mean (location)

$\sigma > 0$ : Standard deviation (scale)

- **Mean and variance**

$$\mathbb{E}(Y) = \mu$$

$$\text{var}(Y) = \sigma^2$$

## The Multivariate Normal Distribution

- **Notation**

$$Y = (Y_1, \dots, Y_k) \sim N(\mu, \Sigma)$$

$$p(y) = N(y|\mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^k |\Sigma|}} \exp\left(-\frac{1}{2}(y - \mu)^T \Sigma^{-1} (y - \mu)\right)$$

$y = (y_1, \dots, y_k)$  : Vector!  $y_i \in [-\infty, \infty]$

- **Parameters**

$\mu = (\mu_1, \dots, \mu_k)$ : Mean vector

$\Sigma \geq 0$ : Positive semi definite covariance matrix,  $\Sigma_{i,j} = \text{cov}(x_i, x_j)$

- **Mean and variance**

$$\mathbb{E}(Y) = \mu$$

$$\text{var}(Y) = \Sigma$$

## The Multivariate Normal Distribution

A **random vector**  $X = (X_1, \dots, X_k)$  is a Gaussian random vector (or  $X_1, \dots, X_k$  are jointly Gaussian r.v.s.) if the joint pdf is of the form

$$p_X(x) = \frac{1}{\sqrt{(2\pi)^k |\Sigma|}} \exp\left(-\frac{1}{2}(x - \mu)^T \Sigma^{-1} (x - \mu)\right)$$

**Property 1 :** For a GRV, uncorrelation implies independence

This can be verified by substituting  $\sigma_{ij} = 0$  for all  $i \neq j$  in the joint pdf. Then  $\Sigma$  becomes diagonal and so does  $\Sigma^{-1}$ , and the joint pdf reduces to the product of the marginal

$$X_i \sim N(\mu_i, \sigma_{ii})$$

## The Multivariate Normal Distribution

**Property 2 :** Linear transformation of a GRV yields a GRV, i.e., given any  $m \times n$  matrix  $A$ , where  $m \leq n$  and  $A$  has full rank  $m$ , then

$$Z = AY \sim N(A\mu, A\Sigma A^T)$$

Because

$$E(Z) = E(AY) = AE(Y) = A\mu$$

$$\begin{aligned}\Sigma_Z &= E[(Z - E(Z))(Z - E(Z))^T] \\ &= E[(AY - A\mu)(AY - A\mu)^T] \\ &= AE[(Y - \mu)(Y - \mu)^T]A^T \\ &= A\Sigma A^T\end{aligned}$$

(The proof is not complete; we need to show the joint PDF is in a right form

**Example:** Let

$$Y = N(\mathbf{0}, \begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix})$$

Find the joint pdf of

$$Z = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} Y$$

# The Multivariate Normal Distribution

## Rough proof of Property 2

1. Assume we have a random vector  $X = (X_1, \dots, X_k)$  is a Gaussian random vector (or  $X_1, \dots, X_k$  are jointly Gaussian r.v.s.) if the joint pdf is of the form

$$p_X(x) = \frac{1}{\sqrt{(2\pi)^k |\det(\Sigma)|}} \exp\left(-\frac{1}{2}(x - \mu)^T \Sigma^{-1} (x - \mu)\right)$$

2 To compute the probability of the transformed random vector  $y$  utilizing the existing pdf  $p_X(x)$ , there should be one-to-one relationship between  $X$  and  $Y$

→  $y = Ax$  have a solution only when  $m \leq n$  and  $A$  has full rank  $m$ ,  $x = (A^T A)^{-1} A^T y$

3. Assuming the left inverse  $(A^T A)^{-1} A^T$  is equivalent to  $A^{-1}$ , we can compute the pdf for  $y$  using  $p_X(x)$ :

$$p_Y(y) = \frac{1}{|\det(A)|} p_X(A^{-1}y)$$

4. Assuming the left inverse  $(A^T A)^{-1} A^T$  is equivalent to  $A^{-1}$ , we can compute the pdf for  $y$  using  $p_X(x)$ :

$$\begin{aligned} p_Y(y) &= \frac{1}{|\det(A)|} p_X(A^{-1}y) \\ &= \frac{1}{|\det(A)|} \frac{1}{\sqrt{(2\pi)^k |\det(\Sigma)|}} \exp\left[-\frac{1}{2}(A^{-1}y - \mu)^T \Sigma^{-1} (A^{-1}y - \mu)\right] \\ &= \frac{1}{\sqrt{(2\pi)^k |\det(A\Sigma A^T)|}} \exp\left[-\frac{1}{2}(y - A\mu)^T (A\Sigma A^T)^{-1} (y - A\mu)\right] \end{aligned}$$

$$\begin{aligned} \mu_Y &= A\mu \\ \Sigma_Y &= A\Sigma A^T \end{aligned}$$

## The Multivariate Normal Distribution

**Property 3 :** Marginal of GRV are Gaussian, i.e., if  $Y$  is GRV then for any subset  $\{i_1, i_2, \dots, i_k\} \subset \{1, 2, \dots, n\}$  of indexes, the RV

$$Z = \begin{bmatrix} Y_{i_1} \\ Y_{i_2} \\ \vdots \\ Y_{i_k} \end{bmatrix} \text{ is GRV}$$

Converse is not generally true

To show this we use Property 2. For example, let  $n = 3$  and  $Z = \begin{bmatrix} Y_1 \\ Y_2 \\ Y_3 \end{bmatrix}$ , we can express  $Z$  as a linear transformation of  $Y$ :

$$Z = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} Y_1 \\ Y_2 \\ Y_3 \end{bmatrix} = \begin{bmatrix} Y_1 \\ Y_2 \\ Y_3 \end{bmatrix}$$

Therefore,

$$Z = N \left( \begin{bmatrix} \mu_1 \\ \mu_3 \end{bmatrix}, \begin{bmatrix} \sigma_{11} & \sigma_{13} \\ \sigma_{31} & \sigma_{33} \end{bmatrix} \right)$$

## The Multivariate Normal Distribution

**Property 4 :** Conditionals of a GRV are Gaussians, more specifically, if

$$Z = \begin{bmatrix} Y_1 \\ - \\ Y_2 \end{bmatrix} \sim N \left( \begin{bmatrix} Y_1 \\ - \\ Y_2 \end{bmatrix}, \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \hline \Sigma_{21} & \Sigma_{22} \end{bmatrix} \right)$$

where  $Y_1$  is  $k$ -dim RV and  $Y_2$  is an  $n - k$  dim RV, then

$$Y_2 | \{Y_1 = y\} \sim N(\Sigma_{21}\Sigma_{11}^{-1}(y - \mu_1) + \mu_2, \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12})$$

**Example:**

$$\begin{bmatrix} Y_1 \\ Y_2 \\ Y_3 \end{bmatrix} \sim N \left( \begin{bmatrix} 1 \\ - \\ 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 1 & 2 & 1 \\ \hline 2 & 5 & 2 \\ \hline 1 & 2 & 9 \end{bmatrix} \right)$$

$$E \left( \begin{bmatrix} Y_2 \\ Y_3 \end{bmatrix} \middle| Y_1 = y \right) = \begin{bmatrix} 2 \\ 1 \end{bmatrix} (y - 1) + \begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 2y \\ y + 1 \end{bmatrix}$$

$$\Sigma \left( \begin{bmatrix} Y_2 \\ Y_3 \end{bmatrix} \middle| Y_1 = y \right) = \begin{bmatrix} 5 & 2 \\ 2 & 9 \end{bmatrix} - \begin{bmatrix} 2 \\ 1 \end{bmatrix} [2 \ 1] = \begin{bmatrix} 1 & 0 \\ 0 & 8 \end{bmatrix}$$

## The Gamma Distribution

- The gamma distribution is frequently used to model waiting times.
- The arrival times in the Poisson process have gamma distributions

- **Notation**

$$Y \sim \text{Gamma}(\alpha, \beta)$$

$$p(y) = \text{Gamma}(y|\alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} y^{\alpha-1} e^{-\beta y}$$
$$y > 0$$

$$\Gamma(\alpha) = \int_0^\infty x^{\alpha-1} e^{-x} dx$$

Gamma function  $\Gamma(\alpha)$  is a normalization constant to ensure that the total probability integrates to 1.

- **Parameters**

$\alpha > 0$  : Shape parameter

$\beta > 0$  : Rate parameter (Inverse scale)

- **Mean and variance**

$$\text{E}(Y) = \frac{\alpha}{\beta}$$

$$\text{var}(Y) = \frac{\alpha}{\beta^2}$$

## The Gamma Distribution

- The gamma distribution is frequently used to model waiting times.
- The arrival times in the Poisson process have gamma distributions

### Notation

$$Y \sim \text{Gamma}(\alpha, \beta)$$

$$p(y) = \text{Gamma}(y|\alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} y^{\alpha-1} e^{-\beta y}$$
$$y > 0$$

### Parameters

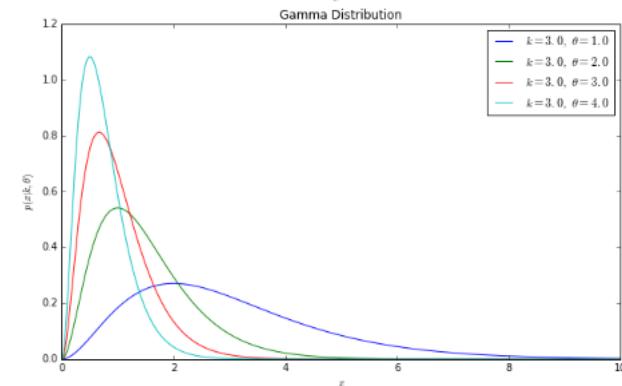
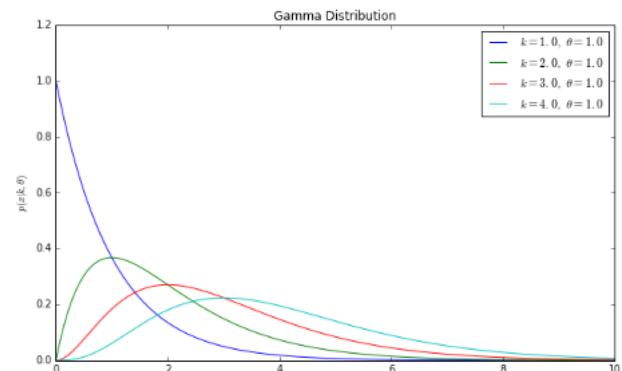
$\alpha > 0$  : Shape parameters

$\beta > 0$  : Inverse scale (rate) parameters

### Mean and variance

$$\text{E}(Y) = \frac{\alpha}{\beta}$$

$$\text{var}(Y) = \frac{\alpha}{\beta^2}$$



## The Gamma Distribution

- The gamma distribution is frequently used to model waiting times.
- The arrival times in the Poisson process have gamma distributions

$Y \sim \text{Gamma}(\alpha, \beta)$

$$p(y) = \text{Gamma}(y|\alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} y^{\alpha-1} e^{-\beta y}$$
$$y > 0$$

### Properties

- $\Gamma(\alpha) = \int_0^\infty x^{\alpha-1} e^{-x} dx$
- $\int_0^\infty x^{\alpha-1} e^{-\lambda x} dx = \frac{\Gamma(\alpha)}{\lambda^\alpha}$  for  $\lambda > 0$
- $\Gamma(\alpha + 1) = \alpha\Gamma(\alpha)$
- $\Gamma(n) = (n - 1)!$ , for  $n = 1, 2, 3, \dots$

$$\begin{aligned} \int_0^\infty \frac{\beta^\alpha}{\Gamma(\alpha)} y^{\alpha-1} e^{-\beta y} dy &= \frac{\beta^\alpha}{\Gamma(\alpha)} \int_0^\infty y^{\alpha-1} e^{-\beta y} dy \\ &= \frac{\beta^\alpha}{\Gamma(\alpha)} \frac{\Gamma(\alpha)}{\beta^\alpha} = 1 \end{aligned}$$

## The Gamma Distribution

$$\begin{aligned} E(Y) &= \int_0^{\infty} y \frac{\beta^\alpha}{\Gamma(\alpha)} y^{\alpha-1} e^{-\beta y} dy \\ &= \frac{\beta^\alpha}{\Gamma(\alpha)} \int_0^{\infty} y^\alpha e^{-\beta y} dy \\ &= \frac{\beta^\alpha}{\Gamma(\alpha)} \frac{\Gamma(\alpha+1)}{\beta^{\alpha+1}} \int_0^{\infty} \frac{\beta^{\alpha+1}}{\Gamma(\alpha+1)} y^\alpha e^{-\beta y} dy \\ &= \frac{\beta^\alpha}{\Gamma(\alpha)} \frac{\alpha \Gamma(\alpha)}{\beta^{\alpha+1}} \quad \therefore \Gamma(\alpha+1) = \alpha \Gamma(\alpha) \\ &= \frac{\alpha}{\beta} \end{aligned}$$

$$\begin{aligned} \text{Gamma}(y|\alpha, \beta) &= \frac{\beta^\alpha}{\Gamma(\alpha)} y^{\alpha-1} e^{-\beta y} \\ \text{Gamma}(y|\alpha+1, \beta) &= \frac{\beta^{\alpha+1}}{\Gamma(\alpha+1)} y^{\alpha+1-1} e^{-\beta y} \end{aligned}$$

## The Exponential Distribution

- Model **the time between the occurrence of events** in an interval of time, or the distance between events in space.
  - ✓ The duration of a phone call to a help center
  - ✓ The time between successive failures of a machine
- The exponential random variable can be viewed as a continuous analogue of the geometric distribution

- **Notation**

$$Y \sim \text{Exp}(\lambda)$$

$$p(y) = \text{Exp}(y|\lambda) = \begin{cases} \lambda e^{-\lambda y} & y \geq 0 \\ 0 & y < 0 \end{cases}$$

- **Parameters**

$\lambda > 0$  : the rate parameter

- **Mean, variance and mode**

$$\text{E}(Y) = \frac{1}{\lambda}$$

$$\text{var}(Y) = \frac{1}{\lambda^2}$$

$$\text{Gamma}(1, \lambda) = \text{Exp}(\lambda)$$

## The Beta Distribution

The beta distribution is a suitable model for the random behavior of percentages and proportions (i.e., a distribution *of probabilities*)

- **Notation**

$$Y \sim \text{Beta}(\alpha, \beta) \quad Y \in [0,1]$$

$$p(y) = \text{Beta}(y|\alpha, \beta) = \frac{1}{B(\alpha, \beta)} y^{\alpha-1} (1-y)^{\beta-1}$$

$$B(\alpha, \beta) = \int_0^1 x^{\alpha-1} (1-x)^{\beta-1} dx$$

Beta function  $B(\alpha, \beta)$  is a normalization constant to ensure that the total probability integrates to 1.

$$B(\alpha, \beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha + \beta)}$$

- **Parameters**

$\alpha, \beta > 0$  : Shape parameters

- **Mean and variance**

$$E(Y) = \frac{\alpha}{\alpha + \beta}$$

$$\text{var}(Y) = \frac{\alpha\beta}{(\alpha + \beta)^2(\alpha + \beta + 1)}$$

## The Inverse Gamma Distribution

- the distribution of the reciprocal of a variable distributed according to the gamma distribution
- Widely used for Bayesian statistics

- **Notation**

$$Y \sim \text{Inv-Gamma}(\alpha, \beta) \quad Y > 0$$

$$p(y) = \text{Inv-Gamma}(y|\alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} y^{-(\alpha+1)} e^{-\beta/y}$$

- **Parameters**

$\alpha > 0$  : Shape parameters

$\beta > 0$  : Inverse scale (rate) parameters

- **Mean and variance**

$$\mathbb{E}(Y) = \frac{\beta}{\alpha - 1} \quad \text{for } \alpha > 1$$

$$\text{var}(Y) = \frac{\beta^2}{(\alpha - 1)^2(\alpha - 2)} \quad \text{for } \alpha > 2$$

## The Dirichlet Distribution

- Multivariate generalization of the Beta distribution
- Probability distribution of probability distribution

$$\theta \sim \text{Beta}(\alpha, \beta) \quad 0 \leq \theta \leq 1$$

$$Y \sim \text{Bin}(n, \theta)$$

$$\theta = (\theta_1, \dots, \theta_k) \sim \text{Dirichlet}(\alpha_1, \dots, \alpha_k) \quad \sum_i^k \theta_i = 1$$

$$Y = (Y_1, \dots, Y_k) \sim \text{Multin}(n, \theta_1, \dots, \theta_k)$$

### Notation

$$Y = (Y_1, \dots, Y_k) \sim \text{Dirichlet}(\alpha_1, \dots, \alpha_k)$$

$$\begin{aligned} p(y) &= \text{Dirichlet}(y | \alpha_1, \dots, \alpha_k) = \frac{1}{B(\alpha)} y_1^{\alpha_1-1} \cdots y_k^{\alpha_k-1} \\ &= \frac{\Gamma(\alpha_1 + \cdots + \alpha_k)}{\Gamma(\alpha_1) \cdots \Gamma(\alpha_k)} y_1^{\alpha_1-1} \cdots y_k^{\alpha_k-1} \end{aligned}$$

$y = (y_1, \dots, y_k)$  is a probability simplex :  $y_j \in [0,1]$ ,  $\sum_j^k y_j = 1$

### Parameters

$$\alpha = (\alpha_1, \dots, \alpha_k), \alpha_i > 0, \quad \alpha_0 = \sum_{j=1}^k \alpha_j$$

### Mean and variance

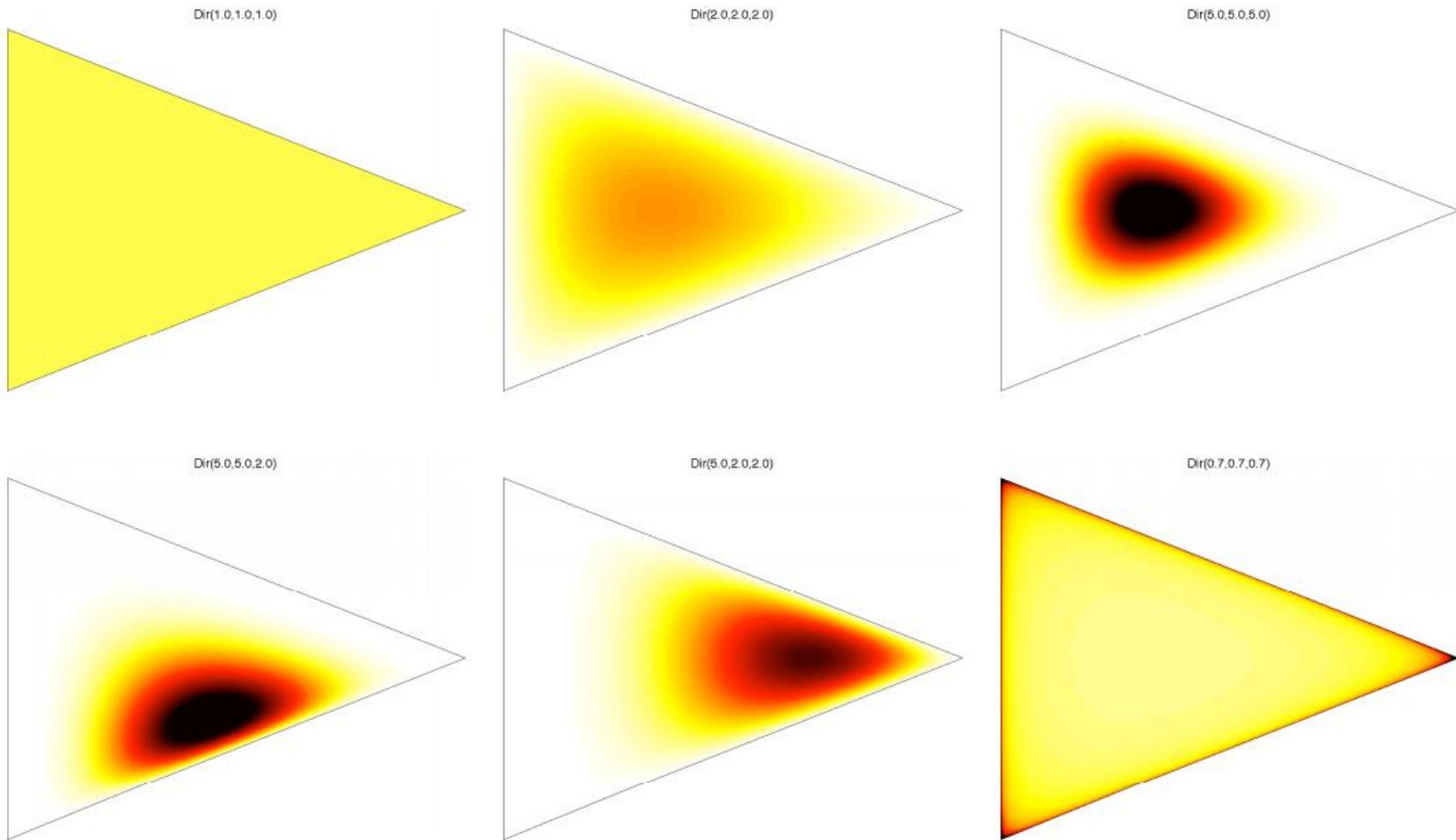
$$E(Y_j) = \frac{\alpha_j}{\alpha_0}, \quad \text{var}(Y_j) = \frac{\alpha_j(\alpha_0 - \alpha_j)}{\alpha_0^2(\alpha_0 + 1)}$$

$$B(\alpha, \beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha + \beta)}$$

$$B(\alpha) = \frac{\Gamma(\alpha_1) \cdots \Gamma(\alpha_k)}{\Gamma(\alpha_1 + \cdots + \alpha_k)}$$

$$B(\alpha) = \frac{\Gamma(\alpha_1) \cdots \Gamma(\alpha_k)}{\Gamma(\alpha_0)}$$

# The Dirichlet Distribution





Home work

## L2. Fundamentals of Bayesian Statistics

What is the probability of getting head?



**Statistics infer the causes that generated the observed data.**

Model



data

$\theta$  (parameters) :  
characteristics of a model

$\theta$  : Probability of having a head for each coin tossing

$y = (y_1, \dots, y_n)$  :  
Observed consequence

(Head, Head, Tail, ... )

## [View on Probability](#)

### **Frequentists :**

- probability only has meaning in terms of a limiting case of repeated measurements
- probabilities are fundamentally related to frequencies of events.

### **Bayesian :**

- degrees of certainty about statements
- probabilities are fundamentally related to our own knowledge about an event.

## Approaches to Statistics

### Frequentists :

- Data are a repeatable random sample → there is a frequency of occurrence
- Underlying parameters remain constant during this repeatable process
- Parameters are fixed and unchanging under all realistic circumstances

### Bayesian :

- Data are observed from the realized sample
- Parameters are unknown and described probabilistically  
(View the world probabilistically)
- Data are fixed

## Frequentist vs Bayesian : Coin Tossing

### Frequentist

- $\theta$  = relative frequency of head in a “large number” of “identical flip”
- Statistical results assume that data were from a controlled experiment
- Nothing is more important than repeatability

Try

1. Estimate the parameter  $\theta$  by conducting experiments
2. Give me estimates

## Frequentist vs Bayesian : Coin Tossing

### Frequentist

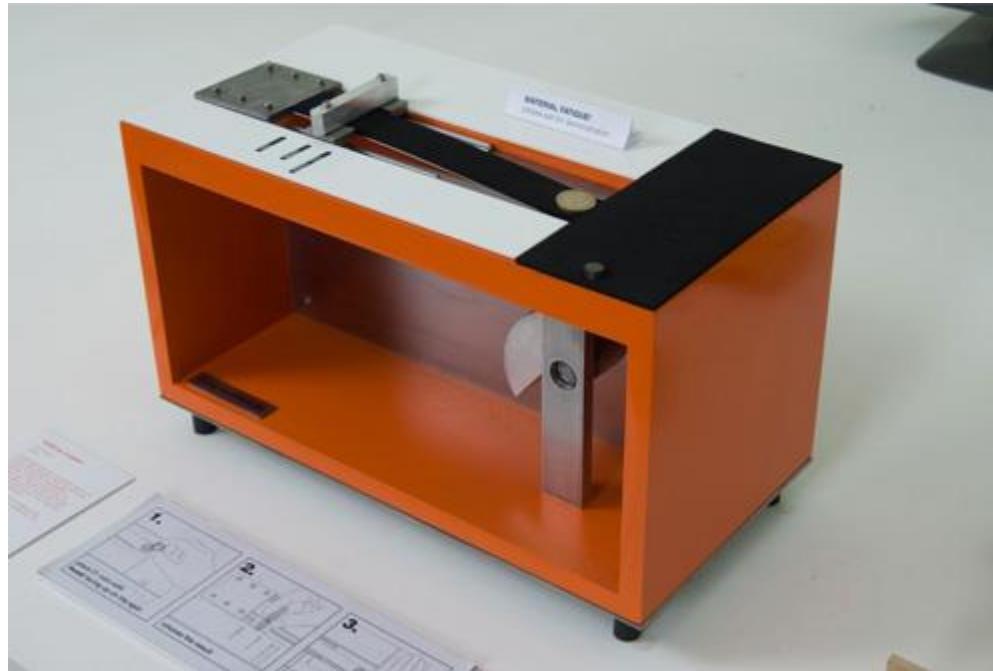
- $\theta$  = relative frequency of head in a “large number” of “identical flip”
- Statistical results assume that data were from a controlled experiment
- Nothing is more important than repeatability

### Issues

- When the number of trials  $n$  is small, estimation is biased       $\frac{\text{#Success}}{\text{\#Trials}}: \frac{1}{3}, \frac{5}{6}, \frac{5}{13}, \frac{129}{313}, \frac{61423}{123400}$
- Identical flip (controlled experiment) is unrealistic

## Identical Coin Tossing

The Artist might be a frequentist



<http://www.dotmancando.info/index.php?/projects/coin-flipper/>

## Frequentist vs Bayesian : Coin Tossing

### Frequentist

- $\theta$  = relative frequency of head in a “large number” of “identical flip”
- Statistical results assume that data were from a controlled experiment
- Nothing is more important than repeatability

### Issues

- When the number of trials  $n$  is small, estimation is biased       $\frac{\text{#Success}}{\text{#Trials}}: \frac{1}{3}, \frac{5}{6}, \frac{5}{13}, \frac{129}{313}, \frac{61423}{123400}$
- Identical flip (controlled experiment) is unrealistic

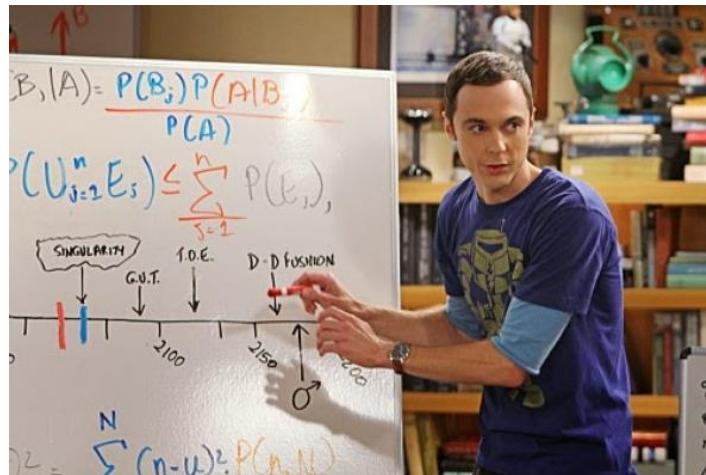
### Bayesian

- Parameters  $\theta$  are varied (uncertain) ← Core part of Bayesian approach
- Use probability concept to provide our belief on  $\theta$  :  $p(\theta)$
- Each parameter  $\theta$  can be associated with different conditions, i.e., orientation, force, etc.
- Data fixed

### Issues

- Subjective on  $p(\theta)$
- How to specify  $p(\theta)$

## Bayes' rule



$$p(A|B) = \frac{p(B|A)p(A)}{p(B)}$$

Bayes' rule will play fundamental role in proceeding Bayesian statistical analysis

Derivation of Bayes' rule :

$$P(A|B) = \frac{P(A \wedge B)}{P(B)} \quad P(B|A) = \frac{P(B \wedge A)}{P(A)}$$

Since  $P(A \wedge B) = P(B \wedge A)$ ,

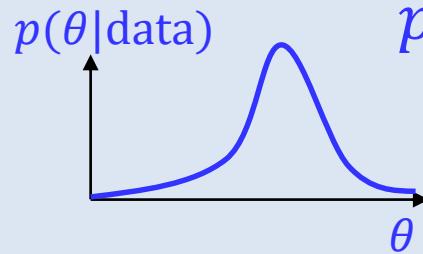
$$\Rightarrow P(A|B)P(B) = P(B|A)P(A)$$

$$\Rightarrow P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

## Bayes' rule in Statistics

### The posterior :

The probability on the parameter  $\theta$  given the evidence (data)



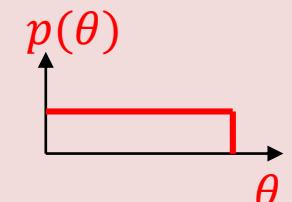
### The evidence

The probability of getting this evidence given the parameter

$$p(\theta|data) = \frac{p(data|\theta)p(\theta)}{p(data)}$$

### The Prior

The belief (uncertainty) about the parameter  $\theta$



### The Marginal probability

The probability of the evidence, Probability of data over all possibilities

- Parameter instantiates one model among a model class

$$p(\theta|data, \text{model}) = \frac{p(data|\theta, \text{model})p(\theta|\text{model})}{p(data|\text{model})}$$

### Approaches

1. Select a model for data (structure)
2. Specify a prior on model parameters
3. Likelihood
4. Construct posterior
5. If necessary, predict unobserved value given the updated information on the parameters

## Bayesian Approach Example : Is sea water contaminated with cholera

### Prior

$$\theta = \begin{cases} 0 & \text{Sea without cholera} \\ 1 & \text{Sea with cholera} \end{cases}$$

$$p(\theta) = \begin{cases} 1/2 & \theta = 0 \\ 1/2 & \theta = 1 \end{cases}$$

Uniform prior  
Discretized  $\theta$

### Likelihood

$$y_i = \begin{cases} 1 & \text{fish with cholera} \\ 0 & \text{fish without cholera} \end{cases}$$

$$p(y_i = 1|\theta = 0) = 0 \rightarrow p(y_i = 0|\theta = 0) = 1$$

$$p(y_i = 1|\theta = 1) = \frac{1}{2} \rightarrow p(y_i = 0|\theta = 1) = \frac{1}{2}$$

**data**  $y = \{y_1 = 0, y_2 = 0, y_3 = 0\}$      $y_i$  are i.i.d.

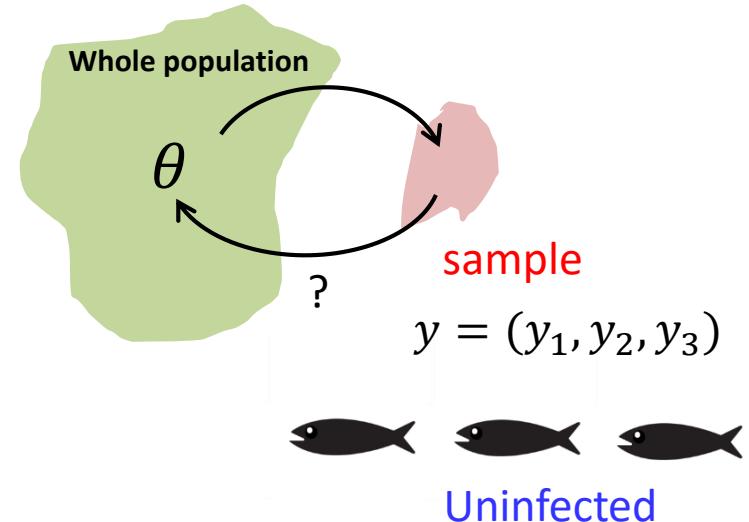
$$P(y|\theta = 0) = (P(y = 0|\theta = 0))^3 = (1)^3 = 1$$

$$P(y|\theta = 1) = (P(y = 0|\theta = 1))^3 = (1/2)^3 = 1/8$$

### Denominator (Marginal likelihood)

$$P(y) = \sum_{\theta} P(y|\theta)P(\theta)$$

$$= P(y|\theta = 0)P(\theta = 0) + P(y|\theta = 1)P(\theta = 1) = 1 \times \frac{1}{2} + \frac{1}{8} \times \frac{1}{2} = \frac{9}{16}$$



## Bayesian Approach Example : Is sea water contaminated with cholera

Posterior distribution:  $p(\theta|y) = \frac{p(y|\theta)p(\theta)}{p(y)}$

$$P(\theta = 0|y) = \frac{P(y|\theta = 0,)P(\theta = 0)}{P(y)} = \frac{1 \times 1/2}{9/16} = \frac{8}{9}$$

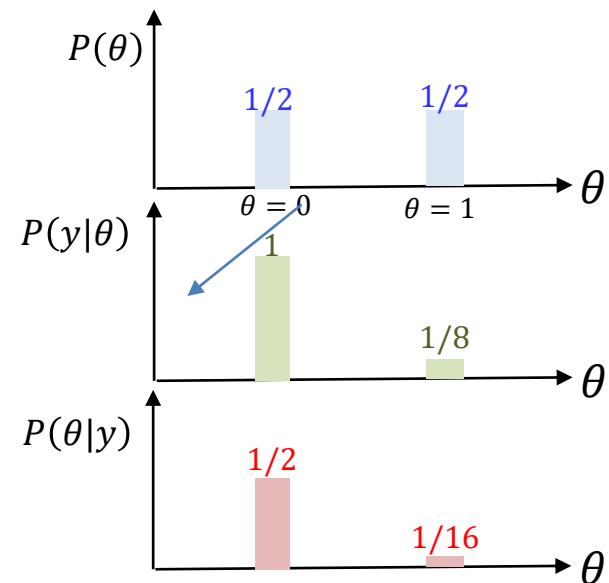
$$P(\theta = 1|y) = \frac{P(y|\theta = 1,)P(\theta = 1)}{P(y)} = \frac{1/8 \times 1/2}{9/16} = \frac{1}{9}$$

Prior  
 $P(\theta = 1) = \frac{1}{2}$

$$P(\theta = 0|y) \propto P(y|\theta = 0,)P(\theta = 0) = 1 \times 1/2$$

$$P(\theta = 1|y) \propto P(y|\theta = 1,)P(\theta = 1) = 1/8 \times 1/2$$

$$\text{Bayes' factor} = \frac{P(\theta = 0|y)}{P(\theta = 1|y)} = \frac{8}{1}$$



## Estimating Model Parameters

### Frequentist approach

$$\begin{aligned} p(\theta|y) &= \frac{p(y|\theta)p(\theta)}{p(y)} & \left( k(y) = \frac{p(\theta)}{p(y)} \right) \\ &= p(y|\theta)k(y) \\ &\propto p(y|\theta) \end{aligned}$$

$$L(\theta) = p(y|\theta)$$

- $\theta$  is fixed
- Maximize likelihood to estimate  $\theta$  (single point estimation)

$$\theta^* = \operatorname{argmax}_{\theta} L(\theta)$$

Maximum Likelihood estimation (MLE)

### Bayesian approach

$$\begin{aligned} p(\theta|y) &= \frac{p(y|\theta)p(\theta)}{p(y)} \\ &= \frac{p(y|\theta)p(\theta)}{\int_{\theta} p(y|\theta)p(\theta)d\theta} \\ &\propto p(y|\theta)p(\theta) \end{aligned}$$

- $\theta$  is random
- Estimation is expressed as prob. dist.
- Posterior is a ***balance*** between our belief and observation

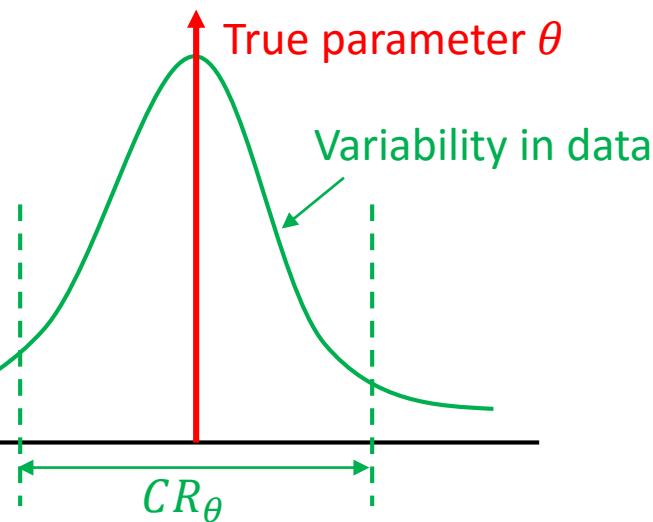
$$\theta^* = \operatorname{argmax}_{\theta} p(\theta|y)$$

Maximum a posteriori estimation (MAP)

## Confidence Interval vs. Credible region

### Frequentist approach

- Describe variability in data given the fixed parameter  $\theta$

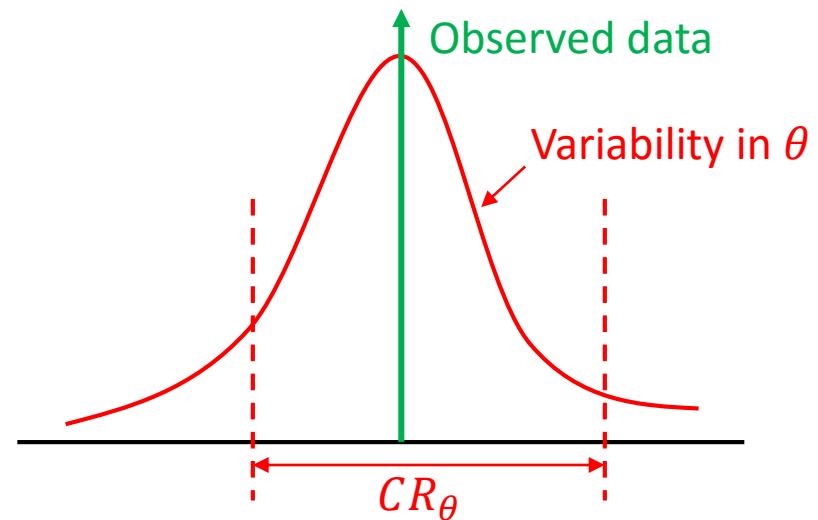


"There is a 95% probability that when I compute  $CI_{\theta}$  from a current data, the computed CI contain  $\theta_{true}$

→ From current data set,  
We can only say that  $\theta \in CI$  or  $\theta \notin CI$

### Bayesian approach

- Describe variability in  $\theta$  for fixed data



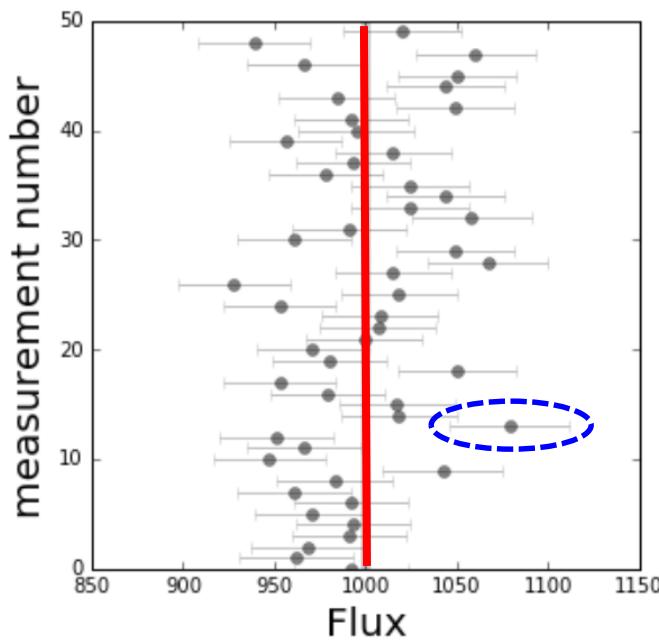
"Given our observed data, there is a 95% probability that the true value of  $\theta$  falls within Credible region  $CR_{\theta}$ "

→ Form current data set,  
We can make Probabilistic statement  
 $\Pr(\theta \in CR) = 0.95$

## Confidence Interval vs. Credible region

### Frequentist approach

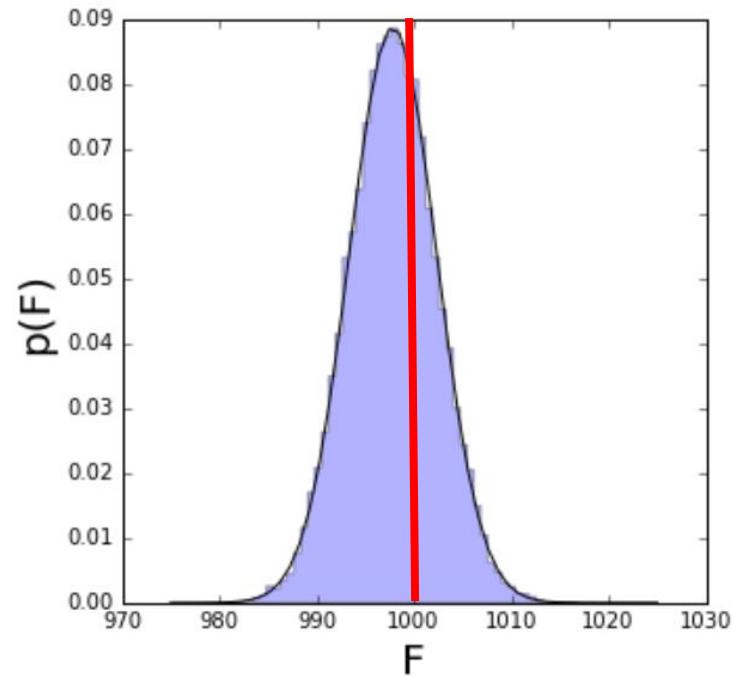
- Describe variability in data given the fixed parameter  $\theta$



$\theta \in \text{CI}$  or  $\theta \notin \text{CI}$

### Bayesian approach

- Describe variability in  $\theta$  for fixed data



$\Pr(\theta \in \text{CR}) = 0.95$

## Confidence Interval

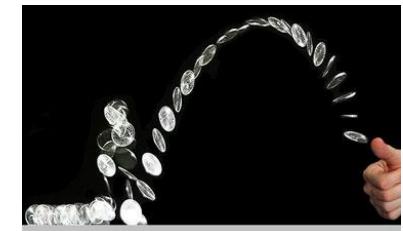
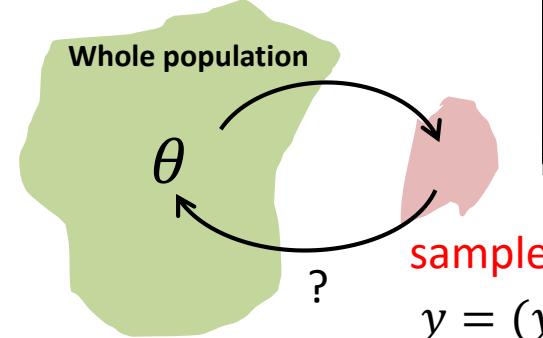
Jupyter Demo Simulation

## Maximum Likelihood Estimation for Coin Flipping Probability

$$Y_i \sim B(\theta) \quad Y_i = \begin{cases} 1 & \text{if Head} \\ 0 & \text{if Tail} \end{cases}$$

$$p(y_i) = B(y_i|\theta) = \theta^{y_i}(1-\theta)^{1-y_i}$$

$\theta \in [0,1]$  : Probability of having a head



Likelihood of a single observation :

$$L(\theta) = P(y_i|\theta) = \theta^{y_i}(1-\theta)^{1-y_i}$$

Likelihood of a three-observations  $y = (y_1, y_2, y_3) = (1, 0, 1)$ :

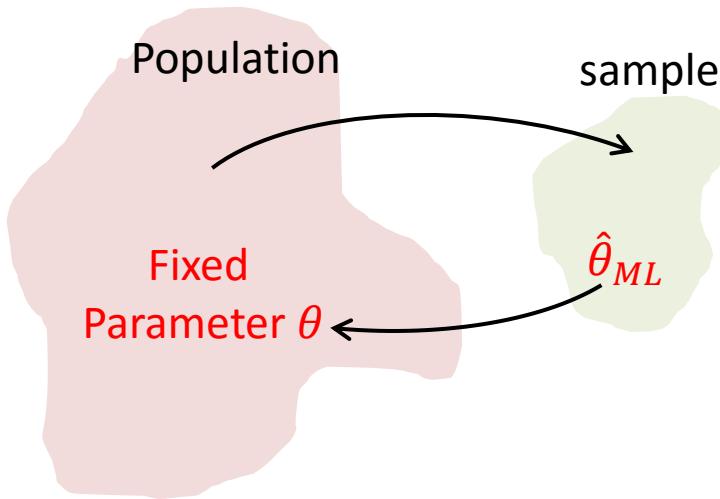
$$\begin{aligned} L(\theta) &= p(y_1 = 1, y_2 = 0, y_3 = 1 | \theta) \\ &= p(1|\theta)p(0|\theta)p(1|\theta) = \theta^2(1-\theta)^1 \quad \text{i.i.d.} \rightarrow \text{Exchangeability} \end{aligned}$$

Likelihood of  $n$  – observations :

$$L(\theta) = P(y_1, y_2, \dots, y_n | \theta) = \theta^{\Sigma y_i}(1-\theta)^{n-\Sigma y_i}$$

$\Sigma_1^n y_i$  : Number of Heads in  $n$  trials (sufficient statistics)

## Maximum Likelihood Estimation for Coin Flipping Probability



Maximum likelihood estimation :

$$\hat{\theta}_{ML} = \underset{\theta}{\operatorname{argmax}} L(\theta) = p(y_1, y_2, \dots, y_n | \theta) = \theta^{\sum y_i} (1 - \theta)^{n - \sum y_i}$$

$$\frac{\partial L(\theta)}{\partial \theta} = \sum y_i \theta^{\sum y_i - 1} (1 - \theta)^{n - \sum y_i} - \theta^{\sum y_i} (n - \sum y_i) (1 - \theta)^{n - \sum y_i - 1} = 0$$

$$\Rightarrow \hat{\theta}_{ML} = \frac{\sum y_i}{n} \quad \text{MLE estimation gives a relative frequency}$$

## Bayesian Approach for Estimating Model Parameters

The essential characteristics of Bayesian methods

= explicit use of probability for **quantifying uncertainty** in the statistical models

**Bayes' rule:**

$$\begin{aligned} p(\theta|y) &= \frac{p(y|\theta)p(\theta)}{p(y)} \\ &= \frac{p(y|\theta)p(\theta)}{\int_{\theta} p(y|\theta)p(\theta)d\theta} \quad \left( \because p(y) = \int_{\theta} p(y|\theta)p(\theta)d\theta \right) \\ &\propto p(y|\theta)p(\theta) \end{aligned}$$

$$\text{Posterior} \propto \text{Likelihood} \times \text{Prior}$$

$p(\theta)$  : Prior - subjective belief about  $\theta$

$p(y|\theta)$  : Likelihood – observation (data) regarding  $\theta$

$p(\theta|y)$  : Posterior - Updated belief about  $\theta$  with the data

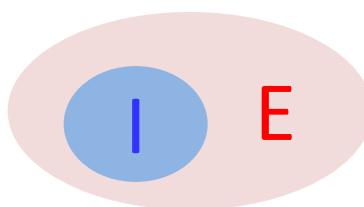
## Exchangeability

$$P(\text{Sequence of 5 coins}) = P(\text{Sequence of 5 coins})$$

**Definition (Infinite exchangeability).** We say that  $(y_1, y_2, \dots)$  is an infinitely exchangeable sequence of random variables if, for any  $n$ , the joint probability  $p(y_1, y_2, \dots, y_n)$  is invariant to permutation of the indices. That is, for any permutation  $\pi$ ,

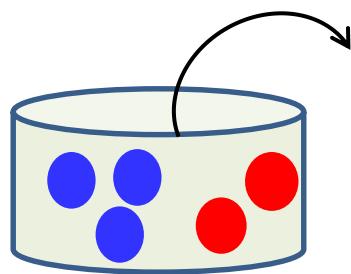
$$p_Y(y_1, y_2, \dots, y_n) = p_Y(y_{\pi_1}, y_{\pi_2}, \dots, y_{\pi_n})$$

R.V.s are independent and identically distributed (i.i.d)



Random variables are infinitely exchangeability

E



$$P(R, R, B, B, B) = P(B, R, B, B, R)$$

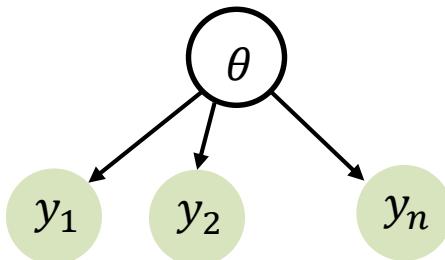
Exchangeable

Check!!

$$P(t_2 = R | t_1 = R) \neq P(t_2 = R | t_1 = B)$$

Not independent

## Exchangeability



**Theorem** (De Finetti, 1930s). A sequence of random variables  $(y_1, y_2, \dots)$  is infinitely exchangeable iff, for all  $n$ ,

$$p(y_1, y_2, \dots, y_n) = \int \prod_{i=1}^n p(y_i|\theta)p(\theta)d\theta$$

$$\text{For coin tossing } p(y_1, y_2, \dots, y_n) = \int \theta^{\sum y_i} [1 - \theta]^{N - \sum y_i} p(\theta)d\theta$$

The theorem says that if we have exchangeable data,

- There must exist a parameter  $\theta$
- There must exist a likelihood  $p(y|\theta)$
- There must exist a distribution  $p$  on  $\theta$
- The above quantities must exist so as to render the data  $y = (y_1, y_2, \dots, y_n)$  conditionally independent

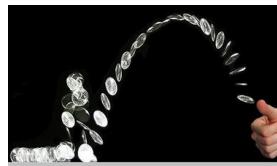
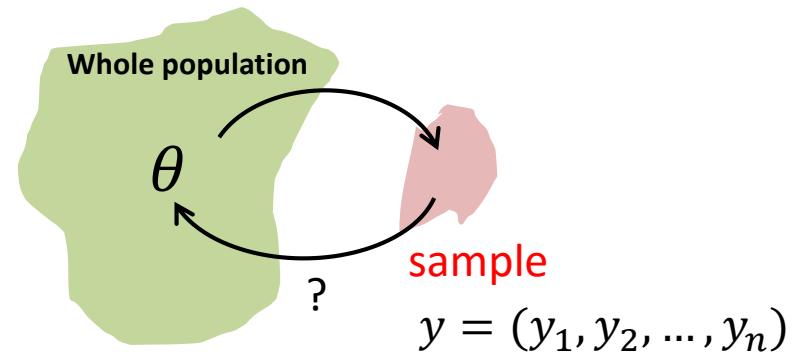
Prior (Bayesian approach) is suggested by the data being exchangeable

## Maximum a posteriori estimation (Bayesian Estimation) for Coin Flipping Probability

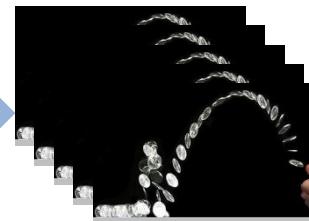
$$Y_i \sim B(\theta) \quad Y_i = \begin{cases} 1 & \text{if Head} \\ 0 & \text{if Tail} \end{cases}$$

$$p(y_i) = B(y_i|\theta) = \theta^{y_i}(1-\theta)^{1-y_i}$$

$\theta \in [0,1]$  : Probability of having a head



$n$  times



### Bernoulli distribution

$$Y_i \sim B(\theta) \quad Y_i = \begin{cases} 1 & \text{if Head} \\ 0 & \text{if Tail} \end{cases}$$

$$p(y_i) = B(y_i|\theta) = \theta^{y_i}(1-\theta)^{1-y_i}$$

$\theta \in [0,1]$  : Probability of having a head

### Binomial distribution

$$Y \sim \text{Bin}(n, \theta)$$

$$p(y|\theta) = \text{Bin}(y|n, \theta) = \binom{n}{y} \theta^y (1-\theta)^{n-y}$$

$$y \in \{0, 1, \dots, n\}$$

The number of successes in a sequence of  $n$  independent yes/no experiments

## Maximum a posteriori estimation (Bayesian Estimation) for Coin Flipping Probability

$$\begin{aligned} p(\theta|y) &= \frac{p(y|\theta)p(\theta)}{p(y)} \\ &= \frac{p(y|\theta)p(\theta)}{\int_0^1 p(y,\theta)d\theta} \\ &= \frac{p(y|\theta)p(\theta)}{\int_0^1 p(y|\theta)p(\theta)d\theta} \end{aligned}$$

Likelihood :

$$Y \sim \text{Bin}(n, \theta) \rightarrow p(y|\theta) = \binom{n}{y} \theta^y (1-\theta)^{n-y}$$

$y = \# \text{of success among } N \text{ trial}$

Prior:

$$\theta \sim \text{Beta}(\alpha, \beta) \rightarrow p(\theta) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta^{\alpha-1} (1-\theta)^{\beta-1}$$

### Numerator

$$\begin{aligned} p(y|\theta)p(\theta) &= \text{Binomial}(y|n, \theta) \times \text{Beta}(\theta|\alpha, \beta) \\ &= \binom{n}{y} \theta^y (1-\theta)^{n-y} \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta^{\alpha-1} (1-\theta)^{\beta-1} \\ &= \frac{\Gamma(n+1)}{\Gamma(y+1)\Gamma(n-y+1)} \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta^y (1-\theta)^{n-y} \theta^{\alpha-1} (1-\theta)^{\beta-1} \\ &= \frac{\Gamma(N+1)\Gamma(\alpha+\beta)}{\Gamma(y+1)\Gamma(n-y+1)\Gamma(\alpha)\Gamma(\beta)} \theta^{\alpha+y-1} (1-\theta)^{\beta+n-y-1} \end{aligned}$$

## Maximum a posteriori estimation (Bayesian Estimation) for Coin Flipping Probability

denominator

$$\begin{aligned}
 p(y) &= \int_0^1 p(y|\theta)p(\theta)d\theta \\
 &= \int_0^1 \binom{n}{y} \theta^y (1-\theta)^{n-y} \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta^{\alpha-1} (1-\theta)^{\beta-1} d\theta \\
 &= \frac{\Gamma(n+1)}{\Gamma(y+1)\Gamma(n-y+1)} \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \int_0^1 \theta^y (1-\theta)^{n-y} \theta^{\alpha-1} (1-\theta)^{\beta-1} d\theta \\
 &= \frac{\Gamma(n+1)\Gamma(\alpha + \beta)}{\Gamma(y+1)\Gamma(n-y+1)\Gamma(\alpha)\Gamma(\beta)} \int_0^1 \theta^{y+\alpha-1} (1-\theta)^{n-y+\beta-1} d\theta \\
 &= \frac{\Gamma(N+1)\Gamma(\alpha + \beta)\Gamma(y+\alpha)\Gamma(n-y+\beta)}{\Gamma(y+1)\Gamma(n-y+1)\Gamma(\alpha)\Gamma(\beta)\Gamma(n+\alpha+\beta)} \int_0^1 \frac{\Gamma(n+\alpha+\beta)}{\Gamma(y+\alpha)\Gamma(n-y+\beta)} \theta^{y+\alpha-1} (1-\theta)^{n-y+\beta-1} d\theta \\
 &= \frac{\Gamma(N+1)\Gamma(\alpha + b)\Gamma(X+\alpha)\Gamma(n-y+\beta)}{\Gamma(y+1)\Gamma(n-y+1)\Gamma(\alpha)\Gamma(\beta)\Gamma(n+\alpha+\beta)} \int_0^1 \text{Beta}(\theta|y+\alpha, n-y+\beta) d\theta \\
 &= \frac{\Gamma(n+1)\Gamma(\alpha + \beta)\Gamma(n+\alpha)\Gamma(n-y+\beta)}{\Gamma(n+1)\Gamma(n-y+1)\Gamma(\alpha)\Gamma(\beta)\Gamma(n+\alpha+\beta)} \\
 &= \text{Beta-Binomial}(y|n, \alpha, \beta)
 \end{aligned}$$

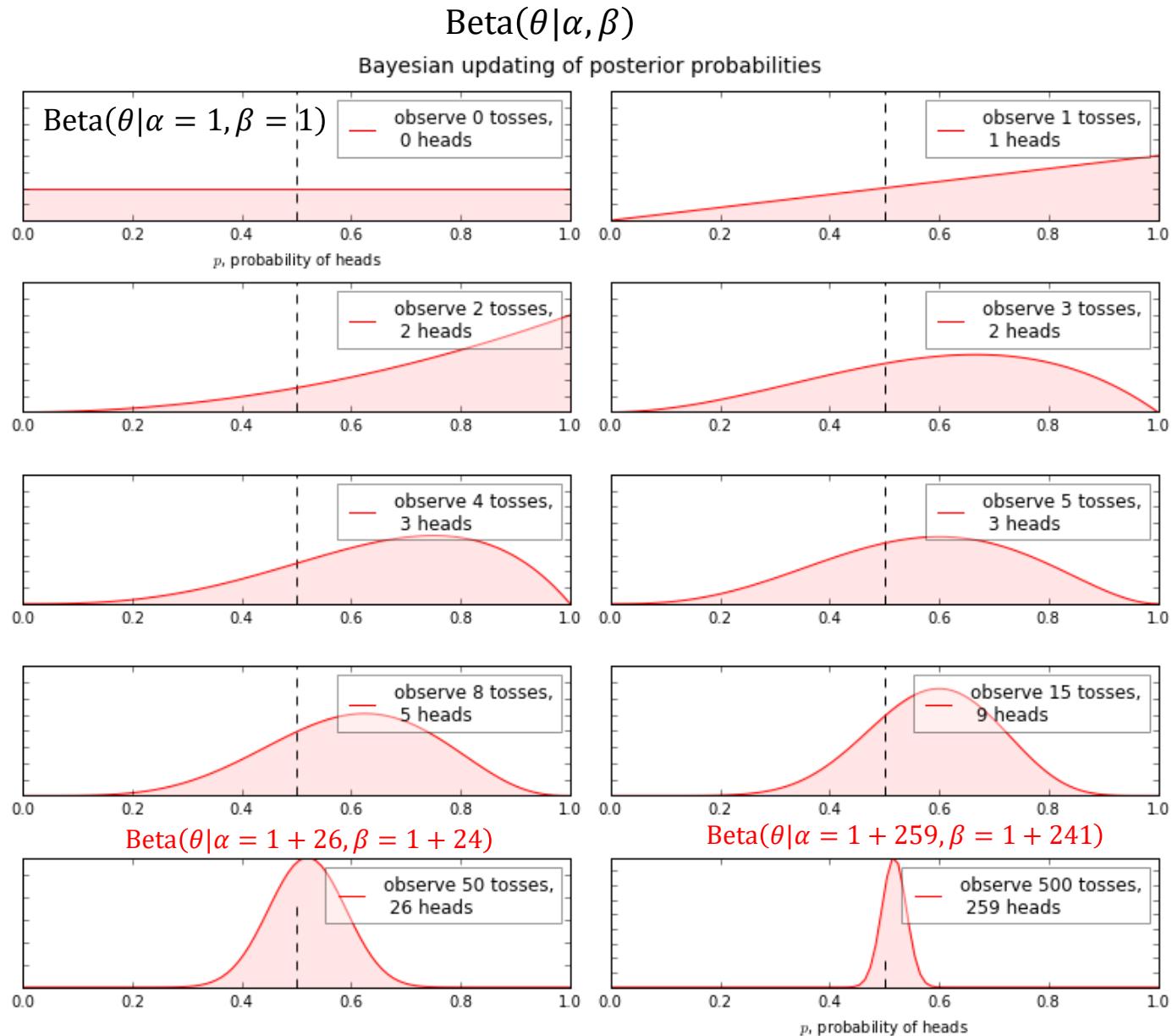
$$\binom{n}{y} = \frac{n!}{y!(n-y)!} = \frac{\Gamma(n+1)}{\Gamma(y+1)\Gamma(n-y+1)}$$

## Maximum a posteriori estimation (Bayesian Estimation) for Coin Flipping Probability

$$\begin{aligned} p(\theta|y) &= \frac{p(y|\theta)p(\theta)}{\int_0^1 p(y|\theta)p(\theta)d\theta} = \frac{\text{Numerator}}{\text{Denominator}} \\ &= \frac{\frac{\Gamma(n+1)\Gamma(\alpha+\beta)}{\Gamma(x+1)\Gamma(n-x+1)\Gamma(\alpha)\Gamma(\beta)}}{\frac{\Gamma(n+1)\Gamma(\alpha+\beta)\Gamma(y+\alpha)\Gamma(n-y+\beta)}{\Gamma(y+1)\Gamma(n-y+1)\Gamma(\alpha)\Gamma(\beta)\Gamma(n+\alpha+\beta)}} \theta^{\alpha+y-1}(1-\theta)^{\beta+n-y-1} \\ &= \frac{\Gamma(n+\alpha+\beta)}{\Gamma(y+\alpha)\Gamma(n-y+\beta)} \theta^{\alpha+y-1}(1-\theta)^{\beta+n-y-1} \\ &= \text{Beta}(\theta|\alpha+y, \beta+n-y) \end{aligned}$$

$$p(\theta) = \text{Beta}(\theta|\alpha, \beta) \xrightarrow{\text{data}} p(\theta|y) = \text{Beta}(\theta|\alpha+y, \beta+n-y)$$

# Maximum a posteriori estimation (Bayesian Estimation) for Coin Flipping Probability



# Maximum a posteriori estimation (Bayesian Estimation) for Coin Flipping Probability

Jupyter Demo Simulation

## Maximum a posteriori estimation (Bayesian Estimation) for Coin Flipping Probability

- From the previous result  $p(\theta|y) = \text{Beta}(\theta|\alpha + y, \beta + n - y)$ ,

$$\mathbb{E}(\theta) = \frac{\alpha}{\alpha + \beta}$$

$$\begin{aligned}\mathbb{E}(\theta|y) &= \frac{\alpha + y}{\alpha + \beta + n} = \frac{\alpha}{\alpha + \beta} \frac{\alpha + \beta}{(\alpha + \beta + n)} + \frac{y}{n} \frac{n}{(\alpha + \beta + n)} \\ &= \mathbb{E}(\theta) \frac{\alpha + \beta}{\alpha + \beta + n} + \hat{\theta}_{ML} \frac{n}{\alpha + \beta + n}\end{aligned}$$

- As  $n \rightarrow \infty$ ,  $\mathbb{E}(\theta|X) \rightarrow \frac{y}{n} = \hat{\theta}_{ML}$
- Large value of  $\alpha + \beta$  signifies Posterior

→ In the limit, the prior does not influence the results. That is, the results are dominated by the data (observation).

$$\text{var}(\theta|y) = \frac{(\alpha + y)(\beta + n - y)}{(\alpha + \beta + n)^2(\alpha + \beta + n + 1)} = \frac{\mathbb{E}(\theta|y)(1 - \mathbb{E}(\theta|y))}{\alpha + \beta + n + 1}$$

- As  $n$  and  $(n - y) \rightarrow \infty$ ,  $\text{var}(\theta|y) \rightarrow \frac{1}{n} \frac{y}{n} \left(1 - \frac{y}{n}\right) = \frac{p(1-p)}{n}$

## Posterior as compromise between data and prior information

- $E(u) = E(E(u|v))$

$$E(\theta) = E(E(\theta|y))$$

*The prior mean of  $\theta$  is the average of all possible posterior means over the distribution of possible data*

- $\text{var}(u) = E(\text{var}(u|v)) + \text{var}(E(u|v))$

$$\text{var}(\theta) = E(\text{var}(\theta|y)) + \text{var}(E(\theta|y))$$

*The posterior variance is on average smaller than the prior variance by an amount that depends on the variation in posterior means over the distribution of all the possible data*

The posterior distribution is centered at a point that represents a compromise between the prior information and the data, and the compromise is controlled to a greater extent by the data as the sample size increases

## The Role of Prior

### Example (BDA Ch.2.4)

Probability of girl birth given placenta Previa

- Among 980 births with placenta Previa, 437 are females
- How much evidence does this provide for the claim that the proportion of female birth in the population of placenta Previa birth is less than 0.485?

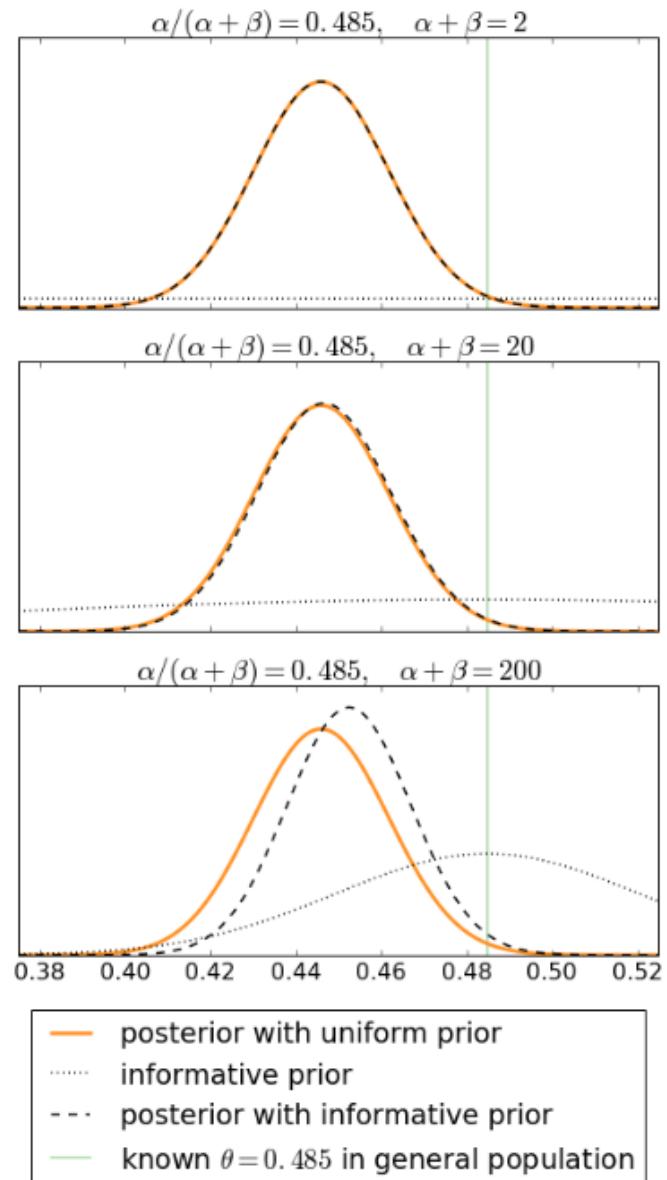
$$P(\theta) = \text{Beta}(\theta|\alpha, \beta)$$

$$P(\theta|y) = \text{Beta}(\theta|\alpha + 437, \beta + 543)$$

$$\mathbb{E}(\theta) = \frac{\alpha + 437}{\alpha + 437 + \beta + 543}$$

Parameters of the prior distribution

$\frac{\alpha}{\alpha+\beta}$	$\alpha + \beta$	Posterior median of $\theta$	95% posterior interval for $\theta$
0.500	2	0.446	[0.415, 0.477]
0.485	2	0.446	[0.415, 0.477]
0.485	5	0.446	[0.415, 0.477]
0.485	10	0.446	[0.415, 0.477]
0.485	20	0.447	[0.416, 0.478]
0.485	100	0.450	[0.420, 0.479]
0.485	200	0.453	[0.424, 0.481]



## Beyond Parameter Estimation: Prediction based on Bayesian Approach

- **Posterior unpredictable distribution**

the distribution of the unknown and an observable parameter  $\theta$  given observed  $y$

$$p(\theta|y) = \frac{p(y|\theta)p(\theta)}{p(y)} = \frac{p(y|\theta)p(\theta)}{\int_{\theta} p(y|\theta)p(\theta)d\theta} \propto p(y|\theta)p(\theta)$$

- **Prior predictive distribution**

Before the data  $y$  are considered, the distribution of the unknown but observable  $y$  is

$$p(y) = \int_{\theta} p(y, \theta) d\theta = \int_{\theta} p(y|\theta)p(\theta)$$

- **Posterior predictive distribution**

Prediction for an observable  $\hat{y}$  conditional on the observed  $y$

$$p(\hat{y}|y) = \int_{\theta} p(\hat{y}, \theta|y) d\theta = \int_{\theta} p(\hat{y}|\theta, y)p(\theta|y) d\theta = \int_{\theta} p(\hat{y}|\theta)p(\theta|y) d\theta$$

- Posterior predictive distribution=an average of conditional predictions over the posterior dist. on  $\theta$

## Prior predictive distribution: Coin tossing example

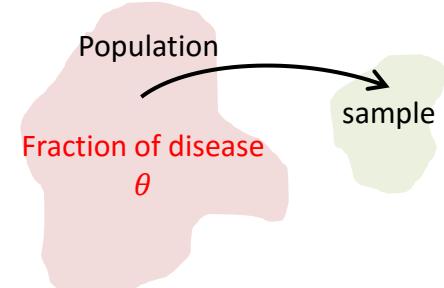
- Prior predictive distribution

Before the data  $y$  are considered, the distribution of the unknown but observable  $y$  is

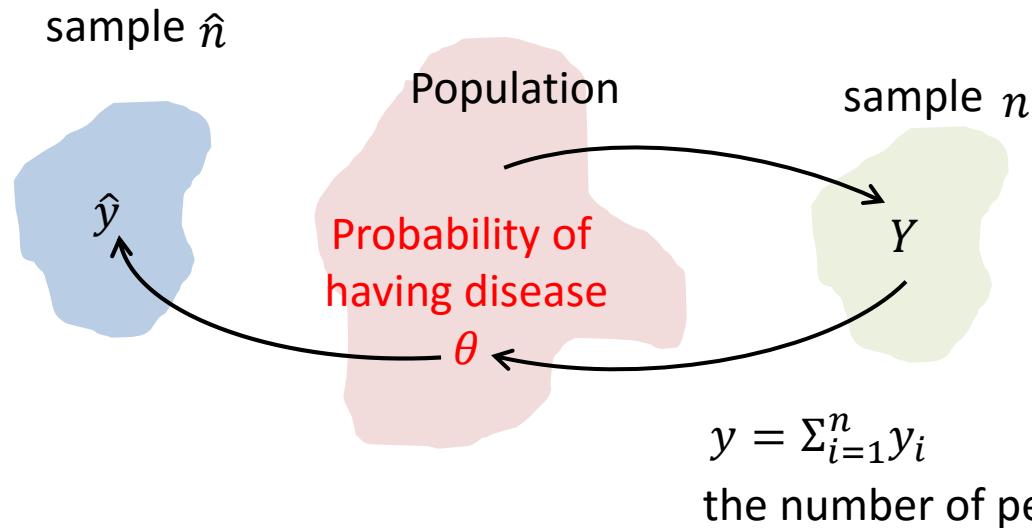
$y = \sum_{i=1}^n y_i$  : the number of persons with diseases among  $n$

$$\begin{aligned}
 P(y) &= \int_0^1 P(y|\theta)d\theta \\
 &= \int_0^1 P(y|\theta)p(\theta)d\theta \quad P(y|\theta) = \text{Bin}(y|n, \theta), \quad p(\theta) = \text{Beta}(\alpha, \beta) \\
 &= \int_0^1 \binom{n}{y} \theta^y (1-\theta)^{n-y} \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta^{\alpha-1} (1-\theta)^{\beta-1} d\theta \quad \binom{n}{y} = \frac{y!}{Y!(n-y)!} = \frac{\Gamma(n+1)}{\Gamma(Y+1)\Gamma(n-Y+1)} \\
 &= \frac{\Gamma(n+1)}{\Gamma(y+1)\Gamma(n-y+1)} \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \int_0^1 \theta^y (1-\theta)^{n-y} \theta^{\alpha-1} (1-\theta)^{\beta-1} d\theta \\
 &= \frac{\Gamma(n+1)\Gamma(\alpha+\beta)}{\Gamma(y+1)\Gamma(n-y+1)\Gamma(\alpha)\Gamma(\beta)} \int_0^1 \theta^{y+\alpha-1} (1-\theta)^{n-y+\beta-1} d\theta \\
 &= \frac{\Gamma(n+1)\Gamma(\alpha+\beta)\Gamma(y+\alpha)\Gamma(n-y+\beta)}{\Gamma(y+1)\Gamma(n-y+1)\Gamma(\alpha)\Gamma(\beta)\Gamma(n+\alpha+\beta)} \int_0^1 \frac{\Gamma(n+\alpha+\beta)}{\Gamma(y+\alpha)\Gamma(n-y+\beta)} \theta^{y+\alpha-1} (1-\theta)^{n-y+\beta-1} d\theta \\
 &= \frac{\Gamma(n+1)\Gamma(\alpha+\beta)\Gamma(y+\alpha)\Gamma(n-y+\beta)}{\Gamma(y+1)\Gamma(n-y+1)\Gamma(\alpha)\Gamma(\beta)\Gamma(n+\alpha+\beta)} \int_0^1 \text{Beta}(\theta|y+\alpha, n-y+\beta) d\theta \\
 &= \frac{\Gamma(n+1)\Gamma(\alpha+\beta)\Gamma(y+\alpha)\Gamma(n-y+\beta)}{\Gamma(y+1)\Gamma(n-y+1)\Gamma(\alpha)\Gamma(\beta)\Gamma(y+\alpha+\beta)} 1 \\
 &= \text{Beta-Binomial}(y|n, \alpha, \beta)
 \end{aligned}$$

For  $\alpha = 1, \beta = 1, P(y) = \frac{1}{n+1}$  (check!)



## Posterior-Predictive Distribution : Coin tossing example



- **Posterior predictive distribution**

Prediction for an observable  $\hat{y}$  conditional on the observed  $y$

$$p(\hat{y}|y) = \int_{\theta} p(\hat{y}, \theta|y) d\theta = \int_{\theta} p(\hat{y}|\theta, y)p(\theta|y) d\theta = \int_{\theta} p(\hat{y}|\theta)p(\theta|y) d\theta$$

- Posterior predictive distribution=an average of conditional predictions over the posterior dist. on  $\theta$

## Posterior-Predictive Distribution : Coin tossing example

Remember Prior Predictive distribution

$$\begin{aligned} p(\hat{y}) &= \int_0^1 p(\hat{y}, \theta) d\theta = \int_0^1 p(\hat{y}|\theta) p(\theta) d\theta \\ &= \text{Beta-Binomial}(\hat{Y}|n, \alpha, \beta) \end{aligned}$$

$$p(\theta) = \text{Beta}(a, b)$$



Using this result, the posterior predictive distribution is

$$\begin{aligned} p(\hat{y}|y) &= \int_0^1 p(\hat{y}, \theta|y) d\theta = \int_0^1 p(\hat{y}|\theta) p(\theta|y) d\theta \\ &= \text{Beta-Binomial}(\hat{Y}|N, \alpha + y, \beta + n - y) \end{aligned}$$

$$p(\theta|y) = \text{Beta}(\alpha + y, \beta + n - y)$$

When  $\hat{y} = 1$ ,  $p(\hat{y} = 1|y) = \int_0^1 \theta p(\theta|y) d\theta \quad \because p(\hat{y} = 1|\theta) = \theta$

$$\begin{aligned} &= \mathbb{E}[\theta|y] \\ &= \frac{\alpha + y}{\alpha + y + \beta + n - y} \\ &= \frac{\alpha + y}{\alpha + \beta + n} \quad \leftarrow \text{When } N \text{ is small prior has strong influence} \end{aligned}$$

For  $\alpha = 1, \beta = 1, P(\hat{y}|y) = \frac{y+1}{n+2}$  (check!)

## Beyond Parameter Estimation: Prediction based on Bayesian Approach

Jupyter Demo Simulation

## Three Steps in Bayesian Approaches

### Step 1: Modeling

setting up a full probability model, a joint probability distribution for all observable and unobservable quantities in a target problem

### Step 2: Inferencing

calculate and interpret the appropriate posterior distribution, the conditional probability distribution of the unobserved quantities of interests

### Step 3: Checking

Evaluate the fit of the model and the sensitiveness of the assumption in step 1

## L3. Conjugate Models

## **Questions from last lecture**

1. From Bernoulli to Binomial? Does the order matter?
2. How to find the credible region?
3. Are the hyper parameters fixed?
4. How prior can be specified?
5. The meaning of  $p(\hat{y}|y)$ ?

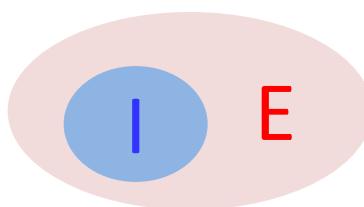
## Exchangeability

$$P(\text{Sequence of 5 coins}) = P(\text{Sequence of 5 coins})$$

**Definition (Infinite exchangeability).** We say that  $(y_1, y_2, \dots)$  is an infinitely exchangeable sequence of random variables if, for any  $n$ , the joint probability  $p(y_1, y_2, \dots, y_n)$  is invariant to permutation of the indices. That is, for any permutation  $\pi$ ,

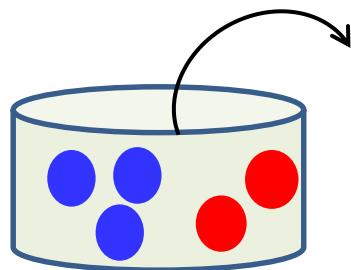
$$p_Y(y_1, y_2, \dots, y_n) = p_Y(y_{\pi_1}, y_{\pi_2}, \dots, y_{\pi_n})$$

R.V.s are independent and identically distributed (i.i.d)



Random variables are infinitely exchangeability

E



$$P(R, R, B, B, B) = P(B, R, B, B, R)$$

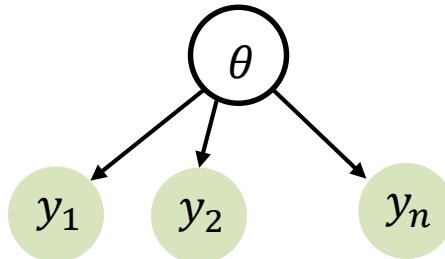
Exchangeable

Check!!

$$P(t_2 = R | t_1 = R) \neq P(t_2 = R | t_1 = B)$$

Not independent

## Exchangeability



**Theorem** (De Finetti, 1930s). A sequence of random variables  $(y_1, y_2, \dots)$  is infinitely exchangeable iff, for all  $n$ ,

$$p(y_1, y_2, \dots, y_n) = \int \prod_{i=1}^n p(y_i|\theta)p(\theta)d\theta$$

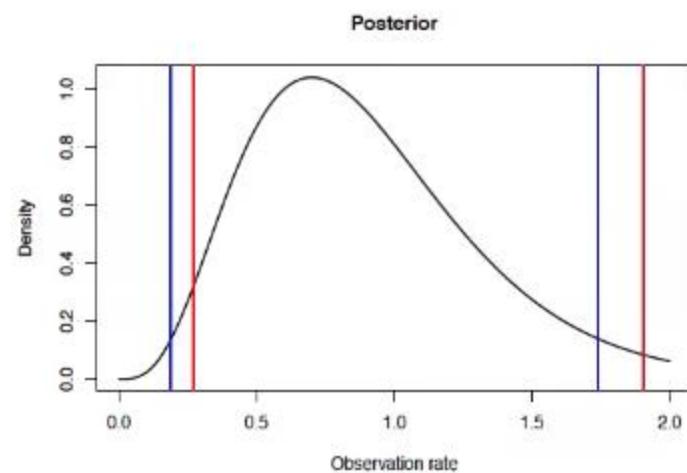
$$\text{For coin tossing } p(y_1, y_2, \dots, y_n) = \int \binom{n}{\sum y_i} \theta^{\sum y_i} [1 - \theta]^{N - \sum y_i} p(\theta)d\theta$$

The theorem says that if we have exchangeable data,

- There must exist a parameter  $\theta$
- There must exist a likelihood  $p(y|\theta)$
- There must exist a distribution  $p$  on  $\theta$
- The above quantities must exist so as to render the data  $y = (y_1, y_2, \dots, y_n)$  conditionally independent

Prior (Bayesian approach) is suggested by the data being exchangeable

## Credible region



Equal-tail 95% credible interval (red): (0.27, 1.90)

Highest posterior density (HPD) 95% credible interval (blue): (0.19, 1.74)

## Priors

$$p(\theta|y) = \frac{p(y|\theta)p(\theta)}{p(y)} = \frac{p(y|\theta)p(\theta)}{\int_{\theta} p(y|\theta)p(\theta)d\theta}$$

1. Non-informative prior (objective prior, vague prior, reference prior...)
2. Weakly informative prior
3. Informative prior (subjective prior)

## Conjugacy

From the Bayes rule,  $p(\theta|y) = \frac{p(y|\theta)p(\theta)}{p(y)} = \frac{p(y|\theta)p(\theta)}{\int_{\theta} p(y|\theta)p(\theta)d\theta}$

$$\begin{array}{ccc} \mathcal{P} & \mathcal{F} & \mathcal{P} \\ \text{Prior} & + & \text{Likelihood} = \text{Posterior} \\ \text{Same form of dist.} & \curvearrowright & \end{array}$$

$p(\theta|y) \in \mathcal{P}$  for all  $p(\cdot|\theta) \in \mathcal{F}$  and  $p(\cdot) \in \mathcal{P}$

$\rightarrow \mathcal{P}$  is conjugate for  $\mathcal{F}$

If the **posterior** is a distribution that is of the same family as our **prior**  
 $\rightarrow$  the prior is conjugate to the **likelihood**.

## Advantages

- By using conjugate prior, we know the form of the resultant posterior (no math!)  
 $\rightarrow$  we can easily summarize the results using mean, mode, variance, etc.
- Easy to understand the meaning of the prior used in the analysis (insight). For example, Beta prior is just adding pseudo counts to the data.
- Due to the analytical form available, we can carry out the integration

$$p(y) = \int_{\theta} p(y|\theta)p(\theta)d\theta$$

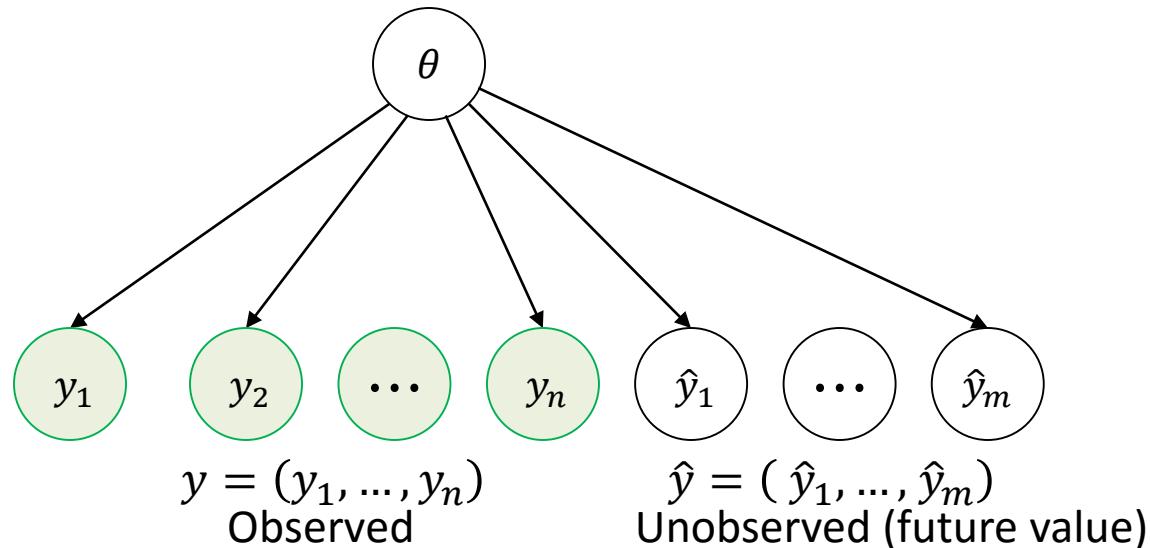
## Conjugate pairs

	Likelihood	Prior	Posterior
Single parameter model	Binomial	Beta	Beta
	Negative Binomial	Beta	Beta
	Geometric	Beta	Beta
	Poisson	Gamma	Gamma
	Exponential	Gamma	Gamma
	Normal (mean unknown)	Normal	Normal
	Normal (variance unknown)	Inverse Gamma	Inverse Gamma
Multi parameters model	Normal (mean and variance unknown)	Normal-Gamma	Normal-Gamma
	Multinomial	Dirichlet	Dirichlet

## Conjugate pairs

	Likelihood	Prior	Posterior
Single parameter model	Binomial	Beta	Beta
	Negative Binomial	Beta	Beta
	Geometric	Beta	Beta
	Poisson	Gamma	Gamma
	Exponential	Gamma	Gamma
	Normal (mean unknown)	Normal	Normal
Multi parameters model	Normal (variance unknown)	Inverse Gamma	Inverse Gamma
	Normal (mean and variance unknown)	Normal-Gamma	Normal-Gamma
	Multinomial	Dirichlet	Dirichlet

## Bayesian Inference Problems



### Objectives

- **Prior predictive distribution**

$$p(y) = \int_{\theta} p(y|\theta)p(\theta)d\theta = \int_{\theta} p(y|\theta)p(\theta)$$

- **Posterior distribution**

$$p(\theta|y) = \frac{p(y|\theta)p(\theta)}{p(y)} = \frac{p(y|\theta)p(\theta)}{\int_{\theta} p(y|\theta)p(\theta)d\theta} \propto p(y|\theta)p(\theta)$$

- **Posterior predictive distribution**

$$p(\hat{y}|y) = \int_{\theta} p(\hat{y}, \theta|y)d\theta = \int_{\theta} p(\hat{y}|\theta, y)p(\theta|y)d\theta = \int_{\theta} p(\hat{y}|\theta)p(\theta|y)d\theta$$

$$p(\hat{y}|\theta, y) = p(\hat{y}|\theta) \text{ because } \hat{y} \perp y | \theta$$

## **Binomial Likelihood and Beta Prior Distribution (Recap)**

## Binomial Likelihood and Beta Prior Distribution (Recap)

Likelihood :

$$Y \sim \text{Bin}(n, \theta) \rightarrow p(y|\theta) = \binom{n}{y} \theta^y (1-\theta)^{n-y}$$

Prior:

$$\theta \sim \text{Beta}(\alpha, \beta) \rightarrow p(\theta) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta^{\alpha-1} (1-\theta)^{\beta-1}$$

Prior predictive distribution :

$$p(y) = \int_{\theta} p(y, \theta) d\theta = \int_{\theta} p(y|\theta)p(\theta) d\theta$$

$$\begin{aligned}
 p(y) &= \int_0^1 p(y, \theta) d\theta \\
 &= \int_0^1 p(y|\theta)p(\theta) d\theta \quad P(y|\theta) = \text{Bin}(y|n, \theta), \quad p(\theta) = \text{Beta}(\alpha, \beta) \\
 &= \int_0^1 \binom{n}{y} \theta^y (1-\theta)^{n-y} \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta^{\alpha-1} (1-\theta)^{\beta-1} d\theta \quad \binom{n}{y} = \frac{n!}{y!(n-y)!} = \frac{\Gamma(n+1)}{\Gamma(y+1)\Gamma(n-y+1)} \\
 &= \frac{\Gamma(n+1)}{\Gamma(y+1)\Gamma(n-y+1)} \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \int_0^1 \theta^y (1-\theta)^{n-y} \theta^{\alpha-1} (1-\theta)^{\beta-1} d\theta \\
 &= \frac{\Gamma(n+1)\Gamma(\alpha + \beta)}{\Gamma(y+1)\Gamma(n-y+1)\Gamma(\alpha)\Gamma(\beta)} \int_0^1 \theta^{y+\alpha-1} (1-\theta)^{n-y+\beta-1} d\theta \\
 &= \frac{\Gamma(n+1)\Gamma(\alpha + b)\Gamma(y+\alpha)\Gamma(n-y+\beta)}{\Gamma(y+1)\Gamma(n-y+1)\Gamma(\alpha)\Gamma(\beta)\Gamma(n+\alpha+\beta)} \int_0^1 \frac{\Gamma(n+\alpha+\beta)}{\Gamma(y+\alpha)\Gamma(n-y+\beta)} \theta^{y+\alpha-1} (1-\theta)^{n-y+\beta-1} d\theta \\
 &= \frac{\Gamma(n+1)\Gamma(\alpha + \beta)\Gamma(y+\alpha)\Gamma(n-y+\beta)}{\Gamma(y+1)\Gamma(n-y+1)\Gamma(\alpha)\Gamma(\beta)\Gamma(n+\alpha+\beta)} \int_0^1 \text{Beta}(\theta|y+\alpha, n-y+\beta) d\theta \\
 &= \frac{\Gamma(n+1)\Gamma(\alpha + \beta)\Gamma(y+\alpha)\Gamma(n-y+\beta)}{\Gamma(y+1)\Gamma(n-y+1)\Gamma(\alpha)\Gamma(\beta)\Gamma(y+\alpha+\beta)} 1 \\
 &= \text{Beta-Binomial}(y|n, \alpha, \beta)
 \end{aligned}$$

The beta-binomial distribution is the binomial distribution in which the probability of success at each trial is not fixed but random and follows the beta distribution

$$\text{Beta-bin}(y|n, \alpha, \beta) = \int \text{Bin}(y|n, \theta) \text{Beta}(\theta|\alpha, \beta) d\theta$$

## Binomial Likelihood and Beta Prior Distribution (Recap)

Likelihood :

$$Y \sim \text{Bin}(n, \theta) \rightarrow p(y|\theta) = \binom{n}{y} \theta^y (1-\theta)^{n-y} \quad Y \text{ is the number of success among } n \text{ Bernoulli trials}$$

Prior:

$$\theta \sim \text{Beta}(\alpha, \beta) \rightarrow p(\theta) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta^{\alpha-1} (1-\theta)^{\beta-1}$$

Posterior :

$$P(\theta|y) \propto P(y|\theta)p(\theta)$$

$$= \binom{n}{y} \theta^y (1-\theta)^{n-y} \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta^{\alpha-1} (1-\theta)^{\beta-1}$$

$$\propto \theta^y (1-\theta)^{n-y} \theta^{\alpha-1} (1-\theta)^{\beta-1}$$

$$= \theta^{\alpha+y-1} (1-\theta)^{\beta+n-y-1}$$

$$= \text{Beta}(\theta|\alpha + y, \beta + n - y)$$

( $\alpha$  is pseudo counts for the success while  $\beta$  is a pseudo counts for the failure)

## Binomial Likelihood and Beta Prior Distribution (Recap)

Likelihood :

$$Y \sim \text{Bin}(n, \theta) \rightarrow p(y|\theta) = \binom{n}{y} \theta^y (1-\theta)^{n-y}$$

$Y$  is the number of success among  $n$  Bernoulli trial

Prior:

$$\theta \sim \text{Beta}(\alpha, \beta) \rightarrow p(\theta) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta^{\alpha-1} (1-\theta)^{\beta-1}$$

Posterior :

$$P(\theta|y) = \text{Beta}(\theta|\alpha + y, \beta + n - y)$$

Posterior predictive distribution :

$$p(\hat{y}|y) = \int_0^1 p(\hat{y}, \theta|y) d\theta = \int_0^1 p(\hat{y}|\theta)p(\theta|y) d\theta$$

$$p(y) = \int_0^1 p(y, \theta) d\theta = \int_0^1 p(y|\theta) p(\theta) d\theta$$

$$= \text{Beta-Binomial}(y|n, \alpha, \beta)$$

Using this result, the posterior predictive distribution is

$$p(\hat{y}|y) = \int_0^1 p(\hat{y}, \theta|y) d\theta = \int_0^1 p(\hat{y}|\theta) p(\theta|y) d\theta$$

$$= \text{Beta-Binomial}(\hat{y}|n, \alpha + y, \beta + n - y)$$

$$p(\theta) = \text{Beta}(\alpha, \beta)$$

$$p(\theta|y) = \text{Beta}(\alpha + y, \beta + n - y)$$

## Poisson Likelihood-Gamma Prior

## Poisson Likelihood-Gamma Prior

Likelihood :

$$Y_i \sim \text{Poisson}(\lambda) \rightarrow p(y_i|\lambda) = \frac{\lambda^{y_i} e^{-\lambda}}{y_i!}$$

Prior:

$$\lambda \sim \text{Gamma}(\alpha, \beta) \rightarrow p(\lambda) = \frac{\beta^\alpha}{\Gamma(\alpha)} \lambda^{\alpha-1} e^{-\beta\lambda}$$

Prior predictive distribution :

$$p(y) = \int_{\theta} p(y, \lambda) d\theta = \int_{\theta} p(y|\lambda)p(\lambda)d\theta$$

$$\begin{aligned} p(y) &= \int_0^{\infty} \frac{\lambda^y e^{-\lambda}}{y!} \frac{\beta^\alpha}{\Gamma(\alpha)} \lambda^{\alpha-1} e^{-\beta\lambda} d\lambda \\ &= \frac{\beta^\alpha}{y! \Gamma(\alpha)} \int_0^{\infty} \lambda^y e^{-\lambda} \lambda^{\alpha-1} e^{-\beta\lambda} d\lambda \\ &= \frac{\beta^\alpha}{y! \Gamma(\alpha)} \int_0^{\infty} \lambda^{y+\alpha-1} e^{-(\beta+1)\lambda} d\lambda \\ &= \frac{\beta^\alpha}{y! \Gamma(\alpha)} \frac{\Gamma(y+\alpha)}{(\beta+1)^{y+\alpha}} \int_0^{\infty} \frac{(\beta+1)^{y+\alpha}}{\Gamma(y+\alpha)} \lambda^{y+\alpha-1} e^{-(\beta+1)\lambda} d\lambda \\ &= \frac{\beta^\alpha}{y! \Gamma(\alpha)} \frac{\Gamma(y+\alpha)}{(\beta+1)^{y+\alpha}} \quad \text{Gamma}(\lambda|\alpha+y, \beta+1) \\ &= \binom{y+\alpha-1}{y} \left( \frac{1}{\beta+1} \right)^y \left( \frac{\beta}{\beta+1} \right)^\alpha = \text{Neg-bin}(y|\alpha, \beta) \end{aligned}$$

**Negative Binomial distribution (Neg – bin)** is a discrete probability distribution of the number of successes  $y$  in a sequence of independent and identically distributed Bernoulli trials before a specified (non-random) number of failures (denoted  $\alpha$ ) occurs.

$$\text{Neg-bin}(y|\alpha, \beta) = \int \text{Poisson}(y|\lambda) \text{Gamma}(\lambda|\alpha, \beta) d\lambda$$

## Poisson Likelihood-Gamma Prior

Likelihood :

$$Y_i \sim \text{Poisson}(\lambda) \rightarrow p(y_i|\lambda) = \frac{\lambda^{y_i} e^{-\lambda}}{y_i!} \quad E(Y_i) = \lambda$$

Prior:

$$\lambda \sim \text{Gamma}(\alpha, \beta) \rightarrow p(\lambda) = \frac{\beta^\alpha}{\Gamma(\alpha)} \lambda^{\alpha-1} e^{-\beta\lambda} \quad E(\lambda) = \frac{\alpha}{\beta}$$

Posterior :

$$P(\lambda|y) \propto P(y|\lambda)p(\lambda) \quad y = (y_1, \dots, y_n) \text{ is a sequence of i.i.d. observation}$$

$$= \prod_{i=1}^n P(y_i|\lambda)p(\lambda)$$

$$= \prod_{i=1}^n \frac{\lambda^{y_i} e^{-\lambda}}{y_i!} \frac{\beta^\alpha}{\Gamma(\alpha)} \lambda^{\alpha-1} e^{-\beta\lambda}$$

$$= \frac{\lambda^{\sum_i y_i} e^{-n\lambda}}{\prod_i^n y_i!} \frac{\beta^\alpha}{\Gamma(\alpha)} \lambda^{\alpha-1} e^{-\beta\lambda}$$

$$\propto \lambda^{n\bar{y}} e^{-n\lambda} \lambda^{\alpha-1} e^{-\beta\lambda}$$

$$\propto \lambda^{\alpha+n\bar{y}-1} e^{-(\beta+n)\lambda}$$

$$= \text{Gamma}(\alpha + n\bar{y}, \beta + n)$$

$$\because \bar{y} = \frac{1}{n} \sum_{i=1}^n y_i$$

$$E(\lambda) = \frac{\alpha + n\bar{y}}{\beta + n} = \frac{\alpha + \sum_i y_i}{\beta + n}$$

( $\alpha$  is a pseudo count for #events while  $\beta$  is a pseudo count for #observations)

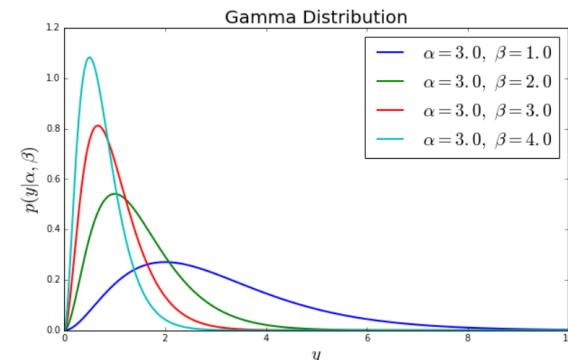
## Poisson Likelihood-Gamma Prior

The posterior mean is

$$\begin{aligned} E(\lambda|y) &= \frac{\alpha + \sum_i y_i}{\beta + n} \\ &= \frac{\alpha}{\beta + n} + \frac{\sum_i y_i}{\beta + n} \\ &= \frac{\beta}{\beta + n} \left( \frac{\alpha}{\beta} \right) + \frac{n}{\beta + n} \left( \frac{\sum_i y_i}{n} \right) \\ &= \frac{\beta}{\beta + n} E(\lambda) + \frac{n}{\beta + n} \hat{\theta}_{ML} \end{aligned}$$

Again, the data get weighted more heavily as  $n \rightarrow \infty$

$\beta$  control strength of prior



## Example

Counts the numbers of PocketMons in each district of SF. The number are

14 13 7 10 15 15 2 13 13 11 10 13 5 13 9 12 9 12 8 7



- It is assumed that the numbers are independent and drawn from a Poisson distribution with mean  $\lambda$

$$Y_i \sim \text{Poisson}(\lambda) \rightarrow p(y_i | \lambda) = \frac{\lambda^{y_i} e^{-\lambda}}{y_i!}$$

- The prior distribution for  $\lambda$  is a Gamma distribution with mean 20 and standard deviation 10

$$\lambda \sim \text{Gamma}(\alpha, \beta) \rightarrow p(\lambda) = \frac{\beta^\alpha}{\Gamma(\alpha)} \lambda^{\alpha-1} e^{-\beta\lambda}$$

$$E(\lambda) = \frac{\alpha}{\beta} = 20, \text{Var}(\lambda) = \frac{\alpha}{\beta^2} = 10^2 \rightarrow \alpha = 4, \beta = 0.2$$

- The posterior is

$$P(\lambda | y) = \text{Gamma}(4 + 211, 0.2 + 20) \quad P(\lambda | y) = \text{Gamma}(\alpha + n\bar{y}, \beta + n)$$

$$E(\lambda | y) = \frac{215}{20.2} = 10.64, \text{Var}(\lambda) = \frac{215}{20.2^2} = 0.5269$$

## Poisson Likelihood-Gamma Prior

Likelihood :

$$Y_i \sim \text{Poisson}(\lambda) \rightarrow p(y_i|\lambda) = \frac{\lambda^{y_i} e^{-\lambda}}{y_i!}$$

Prior:

$$\lambda \sim \text{Gamma}(\alpha, \beta) \rightarrow p(\lambda) = \frac{\beta^\alpha}{\Gamma(\alpha)} \lambda^{\alpha-1} e^{-\beta\lambda}$$

Posterior :

$$P(\lambda|y) = \text{Gamma}(\alpha + n\bar{y}, \beta + n)$$

$y = (y_1, \dots, y_n)$  is a sequence of i.i.d. observation

**Posterior predictive distribution (Using Bayes' rule)**

$$p(\hat{y}|y) = \int_{\lambda} p(\hat{y}, \lambda|y) d\theta = \int_{\lambda} p(\hat{y}|\lambda)p(\lambda|y) d\theta$$

$$p(y) = \int_0^{\infty} p(y, \lambda) d\theta = \int_0^{\infty} p(y|\lambda) p(\lambda) d\theta$$

$$p(\theta) = \text{Gamma}(\alpha, \beta)$$

$$= NB(y|\alpha, \beta) = \binom{y + \alpha - 1}{y} \left( \frac{1}{\beta + 1} \right)^y \left( \frac{\beta}{\beta + 1} \right)^{\alpha}$$

Using this result, the posterior predictive distribution is

$$p(\hat{y}|y) = \int_0^{\infty} p(\hat{y}, \theta|y) d\theta = \int_0^{\infty} p(\hat{y}|\theta) p(\theta|y) d\theta$$

$$p(\theta|y) = \text{Gamma}(\alpha + n\bar{y}, \beta + n)$$

$$= NB(\hat{y}|\alpha + n\bar{y}, \beta + n) = \binom{\hat{y} + \alpha + n\bar{y} - 1}{\hat{y}} \left( \frac{1}{\beta + n + 1} \right)^{\hat{y}} \left( \frac{\beta + n}{\beta + n + 1} \right)^{\alpha + n\bar{y}}$$

## Poisson model parameterized in terms of rate and exposure

Likelihood :

$$Y_i \sim \text{Poisson}(x_i \lambda) \rightarrow p(y_i | \lambda, x_i) = \frac{(x_i \lambda)^{y_i} e^{-(x_i \lambda)}}{(x_i \lambda)!}$$

Where the value  $x_i$  is called the exposure of the  $i$ th unit

$$p(y | \lambda, x_i) = \prod_{i=1}^n \frac{(x_i \lambda)^{y_i} e^{-(x_i \lambda)}}{(x_i \lambda)!} \quad y = (y_1, \dots, y_n) \text{ is a sequence of i.i.d. observation}$$

$$\propto \lambda^{(\sum_i^n y_i)} e^{-(\sum_i^n x_i) \lambda}$$

It is more flexible model in that we can control the unit time or area

Prior:

$$\lambda \sim \text{Gamma}(\alpha, \beta) \rightarrow p(\lambda) = \frac{\beta^\alpha}{\Gamma(\alpha)} \lambda^{\alpha-1} e^{-\beta \lambda}$$

Posterior :

$$P(\lambda | y) = \text{Gamma}\left(\alpha + \sum_{i=1}^n y_i, \beta + \sum_{i=1}^n x_i\right)$$

## Estimating a rate from Poisson data: an idealized example

- A Poisson sampling model is often used for epidemiological data of this form
- 3 persons, out of a population of 200,000 died of asthma  
→ 1.5 cases per 100,000 persons per year : exposure  $x = 2.0$
- Under the Poisson model, the sampling distribution of  $y$ , the number of deaths in a city of 200,000 in one year, can be expressed as

$$y \sim \text{Poisson}(2.0\lambda)$$

Where  $\lambda$  represents the true underlying long-term asthma mortality rate in our city (measured in cases per 100,000 persons per year)

- We can use knowledge about asthma mortality rates around the world to construct a prior distribution for  $\lambda$  and then combine the datum  $y = 3$  to obtain a posterior distribution

## Estimating a rate from Poisson data: an idealized example

Prior:

$$\lambda \sim \text{Gamma}(\alpha = 3, \beta = 5)$$

$$\lambda = \#\text{death per } 100,000$$

Posterior :

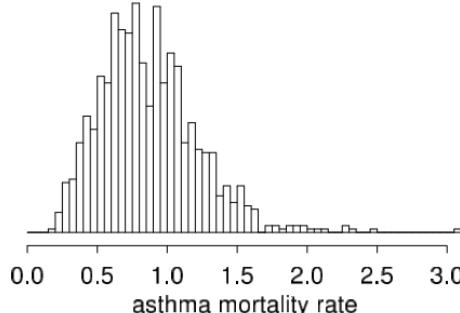
$$P(\lambda|y) = \text{Gamma}\left(\alpha + \sum_{i=1}^n y_i, \beta + \sum_{i=1}^n x_i\right)$$

**Case 1:** 3 persons, out of a population of 200,000 died of asthma for years

$$\sum_{i=1}^n x_i = 2 : \text{exposure}$$

$$\sum_{i=1}^n y_i = 3 \text{ (average)}$$

$$P(\lambda|y) = \text{Gamma}(3 + 3, 5 + 2)$$

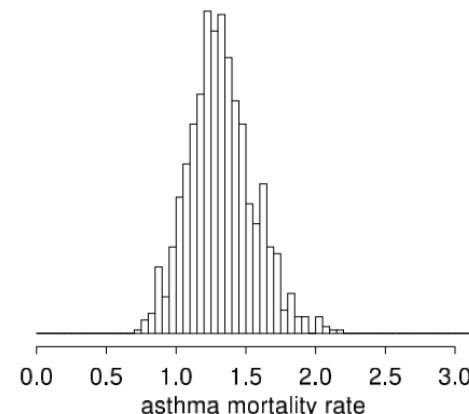


**Case 1:** 30 persons, out of a population of 200,000 died of asthma for 10 years

$$\sum_{i=1}^{n=10} x_i = 20 : \text{exposure}$$

$$\sum_{i=1}^n y_i = 30 \text{ (average)}$$

$$P(\lambda|y) = \text{Gamma}(3 + 30, 5 + 20)$$



## Estimating a rate from Poisson data: an idealized example

**Jupyter Demo Simulation**

## **Exponential Likelihood-Gamma Prior**

## Exponential Likelihood-Gamma Prior

Likelihood :

$$Y_i \sim \text{Exp}(\lambda) \rightarrow p(y_i|\lambda) = \lambda e^{-\lambda y_i}$$

$$\mathbb{E}(Y_i) = \frac{1}{\lambda}$$

Prior:

$$\lambda \sim \text{Gamma}(\alpha, \beta) \rightarrow p(\lambda) = \frac{\beta^\alpha}{\Gamma(\alpha)} \lambda^{\alpha-1} e^{-\beta\lambda}$$

Posterior :

$$P(\lambda|y) \propto P(y|\lambda)p(\lambda) \quad y = (y_1, \dots, y_n) \text{ is a sequence of i.i.d. observation}$$

$$= \prod_{i=1}^n P(y_i|\lambda)p(\lambda)$$

$$= \prod_{i=1}^n \lambda e^{-\lambda y_i} \frac{\beta^\alpha}{\Gamma(\alpha)} \lambda^{\alpha-1} e^{-\beta\lambda}$$

$$= \lambda^n e^{-\lambda \sum_i y_i} \frac{\beta^\alpha}{\Gamma(\alpha)} \lambda^{\alpha-1} e^{-\beta\lambda}$$

$$\propto \lambda^n e^{-\lambda n \bar{y}} \lambda^{\alpha-1} e^{-\beta\lambda} \quad \because \bar{y} = \frac{1}{n} \sum_{i=1}^n y_i$$

$$= \lambda^{\alpha+n-1} e^{-(\beta+n\bar{y})\lambda}$$

$$= \text{Gamma}(\alpha + n, \beta + n\bar{y})$$

## Exponential Likelihood-Gamma Prior (Example)

A machine continuously produces nylon filament. From time to time the filament snaps. Suppose that the time intervals, in minutes, between snaps are random, independent and have an exponential distribution.

- Time interval between two successive failures :  $Y_i \sim \text{Exp}(\lambda) \rightarrow p(y_i|\lambda) = \lambda e^{-\lambda y_i}$   
The mean time =  $E(Y_i) = \frac{1}{\lambda}$
- Prior  $\lambda \sim \text{Gamma}(\alpha = 6, \beta = 1800)$   
 $E(\lambda) = \frac{6}{1800} = 0.0033, \text{Var}(\lambda) = \frac{6}{1800^2} = 1.85 \times 10^{-6}$
- The mean time  $\frac{1}{\lambda}$  follows the *inverse-Gamma* distribution, since its inverse follow Gamma

$$\begin{aligned} E\left(\frac{1}{\lambda}\right) &= \int_0^\infty \frac{1}{\lambda} \frac{\beta^\alpha}{\Gamma(\alpha)} \lambda^{\alpha-1} e^{-\beta\lambda} d\theta & \lambda \sim \text{Gamma}(\alpha, \beta) \rightarrow p(\lambda) = \frac{\beta^\alpha}{\Gamma(\alpha)} \lambda^{\alpha-1} e^{-\beta\lambda} \\ &= \frac{\beta^\alpha \Gamma(\alpha - 1)}{\beta^{\alpha-1} \Gamma(\alpha)} \int_0^\infty \frac{\beta^{\alpha-1}}{\Gamma(\alpha - 1)} \lambda^{\alpha-1-1} e^{-\beta\lambda} d\theta = \frac{\beta}{\alpha - 1} \end{aligned}$$

$$E\left(\frac{1}{\lambda^2}\right) = \frac{\beta^2}{(\alpha - 1)(\alpha - 2)}$$

$$\text{Var}\left(\frac{1}{\lambda}\right) = \frac{\beta^2}{(\alpha - 1)(\alpha - 2)} - \left(\frac{\beta}{\alpha - 1}\right)^2 = \frac{\beta^2}{(\alpha - 1)^2(\alpha - 2)}$$

## Exponential Likelihood-Gamma Prior (Example)

$$Y_i \sim \text{Exp}(\lambda) \rightarrow p(y_i|\lambda) = \lambda e^{-\lambda y_i}$$

$$\lambda \sim \text{Gamma}(\alpha = 6, \beta = 1800)$$

- The prior mean for the time interval :  $E\left(\frac{1}{\lambda}\right) = \frac{\beta}{\alpha-1} = \frac{1800}{6-1} = 360$ (5 hours)
- The prior variance for the time interval :  $\text{var}\left(\frac{1}{\lambda}\right) = \frac{\beta^2}{(\alpha-1)^2(\alpha-2)} = \frac{1800^2}{5^2 \times 4} = 32,400$
- The prior std. for the time interval :  $\text{std}\left(\frac{1}{\lambda}\right) = \sqrt{32,400} = 180$ (3hours)

### Observations:

55 30 231 592 141 139 695 56 803 642 1890 208 246 183 38 486 264 1091 368 222 662 150  
2 133 417 418 743 216 138 306 201 145 804 193 66 577 773 268 388 861

$$P(\lambda|y) = \text{Gamma}(\alpha + n, \beta + \sum_i y_i) = \text{Gamma}(6 + 40, 1,800 + 15,841)$$

- The posterior mean for the time interval :  $E\left(\frac{1}{\lambda}|y\right) = \frac{\beta}{\alpha-1} = \frac{1800+15,841}{46-1} = 392.0$ (minutes)
- The posterior variance for the time interval :  $\text{var}\left(\frac{1}{\lambda}|y\right) = \frac{\beta^2}{(\alpha-1)^2(\alpha-2)} = \frac{17641^2}{45^2 \times 44} = 3492.76$
- The posterior std. for the time interval :  $\text{std}\left(\frac{1}{\lambda}|y\right) = \sqrt{3492.76} = 59.1$ (~1hours)

**Normal Likelihood-Normal Prior (unknown mean and known variance)**

## Normal Likelihood-Normal Prior (unknown mean and known variance)

Likelihood :

$$Y_i \sim N(\theta, \sigma_Y^2) \rightarrow p(y_i | \theta, \sigma_Y^2) = \frac{1}{\sqrt{2\pi\sigma_Y^2}} \exp\left(-\frac{(y_i - \theta)^2}{2\sigma_Y^2}\right)$$

Note that  $\theta = \mu_Y$ , ( $\sigma_Y^2$  is known)

Prior:

$$\theta \sim N(\mu_0, \tau_0^2) \rightarrow p(\theta) = \frac{1}{\sqrt{2\pi\tau_0^2}} \exp\left(-\frac{(\theta - \mu_0)^2}{2\tau_0^2}\right)$$

### Prior predictive distribution :

Before the data  $y$  are considered, the distribution of the unknown but observable  $y$  is

$$p(y) = \int_{\theta} p(y, \theta) d\theta = \int_{-\infty}^{\infty} p(y|\theta)p(\theta)d\theta$$

(Without integration)

$$E(y) = E[E(y|\theta)] = E[\theta] = \mu_0$$

$$\because E(y|\theta) = \theta$$

$$\begin{aligned} \text{var}(y) &= E[\text{var}(y|\theta)] + \text{var}(E(y|\theta)) \\ &= E(\sigma_Y^2) + \text{var}(\theta) \\ &= \sigma_Y^2 + \tau_0^2 \end{aligned}$$

$$\because \text{var}(y|\theta) = \sigma_Y^2$$

$$p(y) = N(y|\mu_0, \sigma_Y^2 + \tau_0^2)$$

## Normal Likelihood-Normal Prior (unknown mean and known variance)

Likelihood :

$$Y_i \sim N(\theta, \sigma_Y^2) \rightarrow p(y_i | \theta, \sigma_Y^2) = \frac{1}{\sqrt{2\pi\sigma_Y^2}} \exp\left(-\frac{(y_i - \theta)^2}{2\sigma_Y^2}\right)$$

Note that  $\theta = \mu_Y$ , ( $\sigma_Y^2$  is known)

Prior:

$$\theta \sim N(\mu_0, \tau_0^2) \rightarrow p(\theta) = \frac{1}{\sqrt{2\pi\tau_0^2}} \exp\left(-\frac{(\theta - \mu_0)^2}{2\tau_0^2}\right)$$

Posterior :

$$P(\theta | y) \propto P(y | \theta, \sigma_Y^2) p(\theta) \quad y = (y_1, \dots, y_n) \text{ is a sequence of i.i.d. observation}$$

$$= \prod_{i=1}^n \frac{1}{\sqrt{2\pi\sigma_Y^2}} \exp\left(-\frac{(y_i - \theta)^2}{2\sigma_Y^2}\right) \times \frac{1}{\sqrt{2\pi\tau_0^2}} \exp\left(-\frac{(\theta - \mu_0)^2}{2\tau_0^2}\right)$$

$$\propto \exp\left(-\sum_{i=1}^n \frac{(y_i - \theta)^2}{2\sigma_Y^2} + \frac{(\theta - \mu_0)^2}{2\tau_0^2}\right)$$

$$= \exp\left[-\frac{1}{2} \left( \sum_{i=1}^n \frac{(y_i - \theta)^2}{\sigma_Y^2} + \frac{(\theta - \mu_0)^2}{\tau_0^2} \right)\right]$$

$$= \exp\left[-\frac{1}{2\sigma_Y^2\tau_0^2} \left( \tau_0^2 \sum_{i=1}^n (y_i - \theta)^2 + \sigma_Y^2 (\theta - \mu_0)^2 \right)\right]$$

$$= \exp\left[-\frac{1}{2\sigma_Y^2\tau_0^2} \left( \tau_0^2 \sum_{i=1}^n (y_i^2 - 2\theta y_i + \theta^2) + \sigma_Y^2 (\theta^2 - 2\theta\mu_0 + \mu_0^2) \right)\right]$$

## Normal Likelihood-Normal Prior (unknown mean and known variance)

Posterior :

$$\begin{aligned}
 P(\theta|y) &\propto \exp \left[ -\frac{1}{2\sigma_Y^2\tau_0^2} \left( \tau_0^2 \sum_{i=1}^n (y_i^2 - 2\theta y_i + \theta^2) + \sigma_Y^2(\theta^2 - 2\theta\mu_0 + \mu_0^2) \right) \right] \\
 &= \exp \left[ -\frac{1}{2\sigma_Y^2\tau_0^2} \left( \tau_0^2 \sum_{i=1}^n y_i^2 - 2\tau_0^2\theta n\bar{y} + \tau_0^2 n\theta^2 + \sigma_Y^2\theta^2 - 2\sigma_Y^2\theta\mu_0 + \sigma_Y^2\mu_0^2 \right) \right] \\
 &= \exp \left[ -\frac{1}{2\sigma_Y^2\tau_0^2} \left( \tau_0^2 \sum_{i=1}^n y_i^2 - 2\tau_0^2\theta n\bar{y} + \tau_0^2 n\theta^2 + \sigma_Y^2\theta^2 - 2\sigma_Y^2\theta\mu_0 + \sigma_Y^2\mu_0^2 \right) \right] \\
 &\propto \exp \left[ -\frac{1}{2\sigma_Y^2\tau_0^2} (\theta^2(\sigma_Y^2 + n\tau_0^2) - 2\theta(\mu_0\sigma_Y^2 + n\bar{y}\tau_0^2) + \text{const}) \right] \\
 &= \exp \left[ -\frac{1}{2} \left( \theta^2 \left( \frac{\sigma_Y^2 + n\tau_0^2}{\sigma_Y^2\tau_0^2} \right) - 2\theta \left( \frac{\mu_0\sigma_Y^2 + n\bar{y}\tau_0^2}{\sigma_Y^2\tau_0^2} \right) + \text{const}' \right) \right] \\
 &= \exp \left[ -\frac{1}{2} \left( \theta^2 \left( \frac{1}{\tau_0^2} + \frac{n}{\sigma_Y^2} \right) - 2\theta \left( \frac{\mu_0}{\tau_0^2} + \frac{n\bar{y}}{\sigma_Y^2} \right) + \text{const}' \right) \right] \\
 &= \exp \left[ -\frac{1}{2} \left( \frac{1}{\tau_0^2} + \frac{n}{\sigma_Y^2} \right) \left( \theta^2 - 2\theta \left( \frac{\frac{\mu_0}{\tau_0^2} + \frac{n\bar{y}}{\sigma_Y^2}}{\frac{1}{\tau_0^2} + \frac{n}{\sigma_Y^2}} \right) + \text{const}' \right) \right] \\
 &= \exp \left[ -\frac{1}{2} \left( \frac{1}{\tau_0^2} + \frac{n}{\sigma_Y^2} \right) \left( \theta - \left( \frac{\frac{\mu_0}{\tau_0^2} + \frac{n\bar{y}}{\sigma_Y^2}}{\frac{1}{\tau_0^2} + \frac{n}{\sigma_Y^2}} \right) \right)^2 \right]
 \end{aligned}$$

$\left( \because \bar{y} = \frac{1}{n} \sum_{i=1}^n y_i \right)$

## Normal Likelihood-Normal Prior (unknown mean and known variance)

Likelihood :

$$Y_i \sim N(\theta, \sigma_Y^2) \rightarrow p(y_i | \theta, \sigma_Y^2) = \frac{1}{\sqrt{2\pi\sigma_Y^2}} \exp\left(-\frac{(y_i - \theta)^2}{2\sigma_Y^2}\right)$$

Prior:

$$\theta \sim N(\mu_0, \tau_0^2) \rightarrow p(\theta) = \frac{1}{\sqrt{2\pi\tau_0^2}} \exp\left(-\frac{(\theta - \mu_0)^2}{2\tau_0^2}\right)$$

Posterior on  $\theta = \mu_Y$

$$P(\theta|y) = N\left(\theta \left| \frac{\frac{\mu_0}{\tau_0^2} + \frac{n\bar{y}}{\sigma_Y^2}}{\frac{1}{\tau_0^2} + \frac{n}{\sigma_Y^2}}, \left( \frac{1}{\tau_0^2} + \frac{n}{\sigma_Y^2} \right)^{-1} \right.\right)$$

- Posterior mean  $\mu_1$  :

$$\mu_1 = \frac{\frac{\mu_0}{\tau_0^2} + \frac{n\bar{y}}{\sigma_Y^2}}{\frac{1}{\tau_0^2} + \frac{n}{\sigma_Y^2}} = \frac{\sigma_Y^2 \mu_0}{\sigma_Y^2 + n\tau_0^2} + \frac{n\tau_0^2 \bar{y}}{\sigma^2 + n\tau_0^2}$$

$\mu_0$ : Prior mean  
 $\bar{y} = \frac{1}{n} \sum_{i=1}^n y_i$  : Data mean

- $\tau_0^2 \downarrow \Rightarrow$  Prior mean  $\mu_0$  becomes accurate and influence more to  $\mu_1$
  - $\sigma_Y^2 \downarrow \Rightarrow$  the data become precise, making  $\bar{y}$  stronger
  - $n \uparrow \Rightarrow$   $\bar{y}$  stronger
- Posterior variance  $\tau_1^2$  :

$$\tau_1^2 = \left( \frac{1}{\tau_0^2} + \frac{n}{\sigma_Y^2} \right)^{-1}$$

- Posterior precision :

$$\frac{1}{\tau_1^2} = \frac{1}{\tau_0^2} + \frac{n}{\sigma_Y^2}$$

$\frac{1}{\tau_0^2}$ : Prior precision  
 $\frac{n}{\sigma^2}$ : data precision

## Normal Likelihood-Normal Prior (unknown mean and known variance)

Likelihood :

$$Y_i \sim N(\theta, \sigma_Y^2) \rightarrow p(y_i | \theta, \sigma_Y^2) = \frac{1}{\sqrt{2\pi\sigma_Y^2}} \exp\left(-\frac{(y_i - \theta)^2}{2\sigma_Y^2}\right)$$

Note that  $\theta = \mu_Y$ , ( $\sigma_Y^2$  is known)

Posterior :  $\theta = \mu_Y$

$$P(\theta|y) = N(\theta|\mu_1, \tau_1^2)$$

$$\mu_1 = \frac{\frac{\mu_0}{\tau_0^2} + \frac{n\bar{y}}{\sigma_Y^2}}{\frac{1}{\tau_0^2} + \frac{n}{\sigma_Y^2}} \quad \tau_1^2 = \left( \frac{1}{\tau_0^2} + \frac{n}{\sigma_Y^2} \right)^{-1}$$

$$E[\theta|y] = \mu_1 \quad \text{var}(\theta|y) = \tau_1^2$$

Posterior predictive distribution

$$p(\hat{y}|y) = \int_{\theta} p(\hat{y}, \theta|y) d\theta = \int_{\theta} p(\hat{y}|\theta, y) p(\theta|y) d\theta = \int_{\theta} p(\hat{y}|\theta) p(\theta|y) d\theta$$

(Without integration)

$$E(\hat{y}|y) = E[E(\hat{y}|\theta, y)|y] = E[\theta|y] = \mu_1 \quad \because E(\hat{y}|\theta, y) = E(\tilde{y}|\theta) = \theta$$

$$\begin{aligned} \text{var}(\hat{y}|y) &= E[\text{var}(\hat{y}|\theta, y)|y] + \text{var}(E(\hat{y}|\theta, y)|y) \\ &= E(\sigma_Y^2|y) + \text{var}(\theta|y) \\ &= \sigma_Y^2 + \tau_1^2 \end{aligned} \quad \because \text{var}(\hat{y}|\theta, y) = \sigma_Y^2$$

$$p(\hat{y}|y) = N(\hat{y}|\mu_1, \sigma_Y^2 + \tau_1^2)$$

## Normal Likelihood-Normal Prior (unknown mean and known variance)

$$E(y) = E[E(y|\theta)] = E[\theta] = \mu_0 \quad \because E(y|\theta) = \theta$$

$$\begin{aligned} \text{var}(y) &= E[\text{var}(y|\theta)] + \text{var}(E(y|\theta)) \\ &= E(\sigma_Y^2) + \text{var}(\theta) \\ &= \sigma_Y^2 + \tau_0^2 \end{aligned} \quad \because \text{var}(y|\theta) = \sigma_Y^2$$

$$p(y) = N(y|\mu_0, \sigma_Y^2 + \tau_0^2)$$

$$E(\hat{y}|y) = E[E(\hat{y}|\theta, y)|y] = E[\theta|y] = \mu_1 \quad \because E(\hat{y}|\theta, y) = E(\tilde{y}|\theta) = \theta$$

$$\begin{aligned} \text{var}(\hat{y}|y) &= E[\text{var}(\hat{y}|\theta, y)|y] + \text{var}(E(\hat{y}|\theta, y)|y) \\ &= E(\sigma_Y^2|y) + \text{var}(\theta|y) \\ &= \sigma_Y^2 + \tau_1^2 \end{aligned} \quad \because \text{var}(\hat{y}|\theta, y) = \sigma_Y^2$$

$$p(\hat{y}|y) = N(\hat{y}|\mu_1, \sigma_Y^2 + \tau_1^2)$$

## Normal Likelihood-Normal Prior (unknown mean and known variance) : Example

**Jupyter Demo Simulation**

**Normal Likelihood-Normal Prior (known mean and unknown variance)**

## Normal Likelihood-Normal Prior (known mean and unknown variance)

Likelihood :

$$Y_i \sim N(\mu, \theta) \rightarrow p(y_i | \mu, \theta) = \frac{1}{\sqrt{2\pi\theta}} \exp\left(-\frac{(y_i - \mu)^2}{2\theta}\right)$$

Note that  $\theta = \sigma^2$ , ( $\mu$  is known)

Prior:

$$\theta \sim \text{Inv-Gamma}(\theta | \alpha_0, \beta_0) \rightarrow p(\theta) = \frac{\beta_0^{\alpha_0}}{\Gamma(\alpha_0)} \theta^{-(\alpha_0+1)} \exp\left(-\frac{\beta_0}{\theta}\right)$$

Posterior :

$$P(\theta | y) \propto p(y|\theta)p(\theta) \quad y = (y_1, \dots, y_n) \text{ is a sequence of i.i.d. observation}$$

$$= \prod_{i=1}^n \frac{1}{\sqrt{2\pi\theta}} \exp\left(-\frac{(y_i - \mu)^2}{2\theta}\right) \times \frac{\beta_0^{\alpha_0}}{\Gamma(\alpha_0)} \theta^{-(\alpha_0+1)} \exp\left(-\frac{\beta_0}{\theta}\right)$$

$$\propto \prod_{i=1}^n \theta^{-\frac{1}{2}} \exp\left(-\frac{(y_i - \mu)^2}{2\theta}\right) \times \theta^{-(\alpha_0+1)} \exp\left(-\frac{\beta_0}{\theta}\right)$$

$$= \theta^{-(\alpha_0+1+\frac{n}{2})} \exp\left(-\left(\frac{\beta_0}{\theta} + \frac{\sum_{i=1}^n (y_i - \mu)^2}{2\theta}\right)\right)$$

$$= \theta^{-(\alpha_0+\frac{n}{2}+1)} \exp\left(-\left(\frac{\beta_0 + \frac{1}{2}\sum_{i=1}^n (y_i - \mu)^2}{\theta}\right)\right)$$

$$= \text{Inv-Gamma}\left(\theta | \alpha_0 + \frac{n}{2}, \beta_0 + \frac{1}{2}\sum_{i=1}^n (y_i - \mu)^2\right)$$

## Normal Likelihood-Normal Prior (known mean and unknown variance) : Example

Population mean test score : normal

**Jupyter Demo Simulation**

## **Multi-Parameters Model**

## Multinomial Likelihood-Dirichlet Prior (unknown mean and known variance)

Likelihood :

$$\begin{aligned}
 p(y|\theta) &= \text{Multin}(y|n, \theta_1, \dots, \theta_k) = \binom{n}{y_1 \ y_2 \ \dots \ y_k} \theta_1^{y_1} \dots \theta_k^{y_k} \\
 &= \frac{\Gamma(n+1)}{\prod_{j=1}^K \Gamma(x_j + 1)} \prod_{i=j}^K \theta_j^{y_j}
 \end{aligned}$$

$y = (y_1, \dots, y_j, \dots, y_k)$   
 $y_j \in \{0, 1, \dots, n\}, \sum_{j=1}^k y_j = n$

Prior:

$$\begin{aligned}
 \theta \sim \text{Dirichlet}(\alpha_1, \dots, \alpha_k) \rightarrow p(\theta) &= \frac{\Gamma(\alpha_1 + \dots + \alpha_k)}{\Gamma(\alpha_1) \dots \Gamma(\alpha_k)} \theta_1^{\alpha_1-1} \dots \theta_k^{\alpha_k-1} \\
 &= \frac{\Gamma(\sum_{j=1}^K \alpha_j)}{\prod_{j=1}^K \Gamma(\alpha_j)} \theta_1^{\alpha_1-1} \dots \theta_k^{\alpha_k-1}
 \end{aligned}$$

Posterior :

$$\begin{aligned}
 P(\theta|y) &\propto P(y|\theta)p(\theta) \\
 &= \frac{\Gamma(n+1)}{\prod_{j=1}^K \Gamma(x_j + 1)} \prod_{j=1}^K \theta_j^{y_j} \frac{\Gamma(\sum_{j=1}^K \alpha_j)}{\prod_{j=1}^K \Gamma(\alpha_j)} \theta_1^{\alpha_1-1} \dots \theta_k^{\alpha_k-1} \\
 &\propto \prod_{j=1}^K \theta_j^{y_j} \prod_{j=1}^K \theta_j^{\alpha_j-1} \\
 &\propto \prod_{j=1}^K \theta_j^{\alpha_j + y_j - 1} \\
 &= \text{Dirichlet}(\alpha_1 + y_1, \dots, \alpha_k + y_k)
 \end{aligned}$$

## Multinomial Likelihood-Dirichlet Prior (unknown mean and known variance)

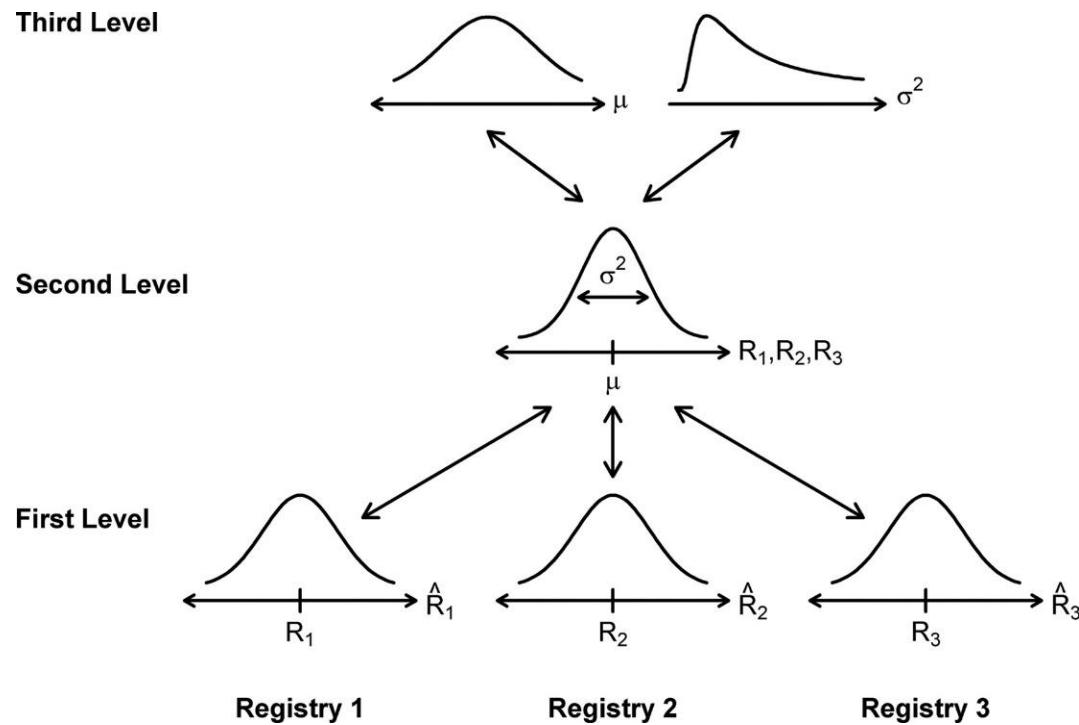
### Posterior predictive distribution

$$p(\hat{y}|y) = \int_{\theta} p(\hat{y}, \theta|y) d\theta = \int_{\theta} p(\hat{y}|\theta, y) p(\theta|y) d\theta = \int_{\theta} p(\hat{y}|\theta) p(\theta|y) d\theta$$

$$\begin{aligned} p(\hat{y}|y) &= \int_{\theta} p(\hat{y}|\theta_1, \dots, \theta_k) p(\theta_1, \dots, \theta_k|y) d\theta \\ &= \int_{\theta} \frac{\Gamma(n+1)}{\prod_{j=1}^K \Gamma(y_j+1)} \prod_{j=1}^K \theta_j^{y_j} \frac{\Gamma(\sum_{j=1}^K \alpha_j)}{\prod_{j=1}^K \Gamma(\alpha_j)} \prod_{j=1}^K \theta_j^{\alpha_j-1} d\theta \\ &= \frac{\Gamma(n+1)}{\prod_{j=1}^K \Gamma(y_j+1)} \frac{\Gamma(\sum_{j=1}^K \alpha_j)}{\prod_{j=1}^K \Gamma(\alpha_j)} \int_{\theta} \prod_{j=1}^K \theta_j^{y_j + \alpha_j - 1} d\theta \\ &= \frac{\Gamma(n+1)}{\prod_{j=1}^K \Gamma(y_j+1)} \frac{\Gamma(\sum_{j=1}^K \alpha_j)}{\prod_{j=1}^K \Gamma(\alpha_j)} \frac{\prod_{j=1}^K \Gamma(y_j + \alpha_j)}{\Gamma(n + \sum_{j=1}^K \alpha_j)} \int_{\theta} \frac{\Gamma(n + \sum_{j=1}^K \alpha_j)}{\prod_{j=1}^K \Gamma(y_j + \alpha_j)} \prod_{j=1}^K \theta_j^{y_j + \alpha_j - 1} d\theta \\ &= \frac{\Gamma(n+1)}{\prod_{j=1}^K \Gamma(y_j+1)} \frac{\Gamma(\sum_{j=1}^K \alpha_j)}{\prod_{j=1}^K \Gamma(\alpha_j)} \frac{\prod_{j=1}^K \Gamma(y_j + \alpha_j)}{\Gamma(n + \sum_{j=1}^K \alpha_j)} \end{aligned}$$

## Flexible Conjugate Prior : Mixture of Priors

## L4. Hierarchical Bayesian models



## Why Hierarchical models?



Seasons	Made	Attempts
2012-2013	25	46
2013-2014	41	93
2014-2015	93	176
2015-2016	79	120

Is his free-throw percentage higher this year than past years?

## Why Hierarchical models?

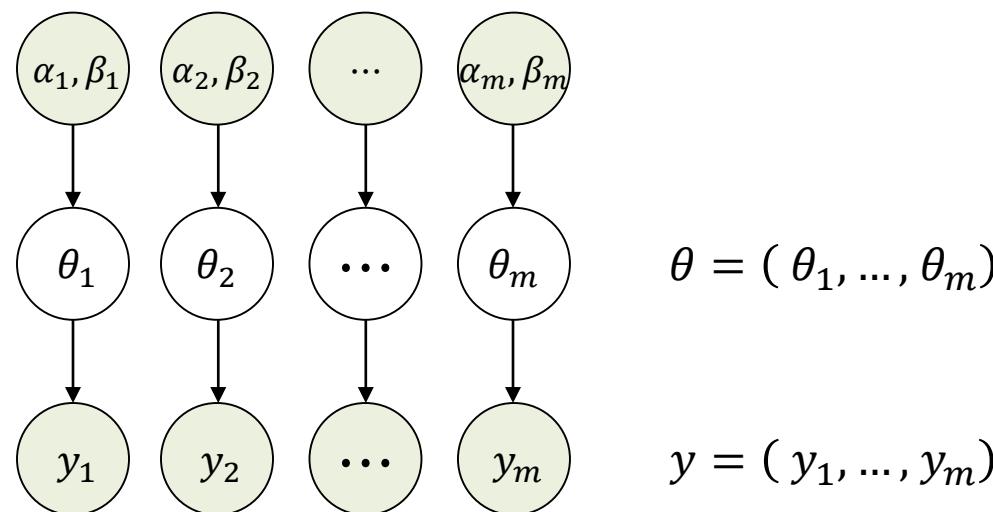
Likelihood :

$$Y_i \sim \text{Bin}(n_i, \theta_i) \rightarrow p(y_i | \theta_i) = \binom{n_i}{y_i} \theta_i^{y_i} (1 - \theta_i)^{n_i - y_i}$$

$$p(y | \theta) = \prod_{i=1}^m p(y_i | \theta_i) = \prod_{i=1}^m \binom{n_i}{y_i} \theta_i^{y_i} (1 - \theta_i)^{n_i - y_i}$$

Prior:

$$p(\theta) = \prod_{i=1}^m p(\theta_i) = \prod_{i=1}^m \text{Beta}(\alpha_i, \beta_i) = \prod_{i=1}^m \frac{\Gamma(\alpha_i + \beta_i)}{\Gamma(\alpha_i)\Gamma(\beta_i)} \theta_i^{\alpha_i-1} (1 - \theta_i)^{\beta_i-1}$$



## Why Hierarchical models?

Likelihood :

$$Y_i \sim \text{Bin}(n_i, \theta_i) \rightarrow p(y_i | \theta_i) = \binom{n_i}{y_i} \theta_i^{y_i} (1 - \theta_i)^{n_i - y_i}$$

$$p(y | \theta) = \prod_{i=1}^m p(y_i | \theta_i) = \prod_{i=1}^m \binom{n_i}{y_i} \theta_i^{y_i} (1 - \theta_i)^{n_i - y_i}$$

Prior:

$$p(\theta) = \prod_{i=1}^m p(\theta_i) = \prod_{i=1}^m \text{Beta}(\alpha_i, \beta_i) = \prod_{i=1}^m \frac{\Gamma(\alpha_i + \beta_i)}{\Gamma(\alpha_i)\Gamma(\beta_i)} \theta_i^{\alpha_i-1} (1 - \theta_i)^{\beta_i-1}$$

Posterior :

$$\begin{aligned} P(\theta | y) &\propto P(y | \theta)p(\theta) \\ &= \prod_{i=1}^m p(y_i | \theta_i) \prod_{i=1}^m p(\theta_i) \\ &= \prod_{i=1}^m p(y_i | \theta_i) p(\theta_i) \\ &= \prod_{i=1}^m \text{Beta}(\theta_i | \alpha_i + y_i, \beta_i + n_i - y_i) \end{aligned}$$

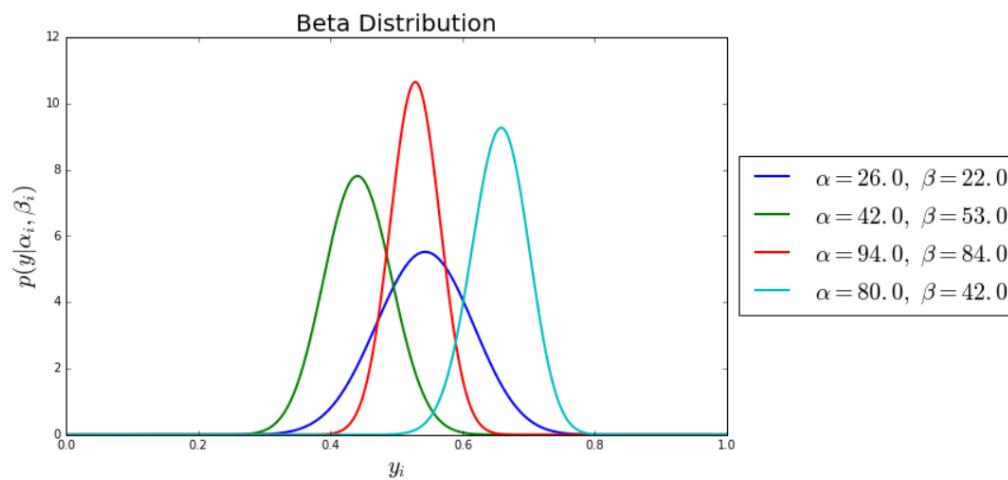
So the posterior for each  $\theta_i$  is exactly the same as if we treated each season independently

## Why Hierarchical models?

Assuming  $\theta_i \sim \text{Beta}(\alpha_i = 1, \beta_i = 1)$  for all  $i$ , results in independent Beta posterior

Seasons	Made	Attempts
2012-2013	25	46
2013-2014	41	93
2014-2015	93	176
2015-2016	79	120

Seasons	$\alpha_i$	$\beta_i$
2012-2013	26	22
2013-2014	42	53
2014-2015	94	84
2015-2016	80	42



Is there any way to use the data from the previous seasons for estimating the success probability for the current season?

## Why Hierarchical models?

Likelihood :

$$Y_i \sim \text{Bin}(n_i, \theta_i) \rightarrow p(y_i | \theta_i) = \binom{n_i}{y_i} \theta_i^{y_i} (1 - \theta_i)^{n_i - y_i}$$

$$p(y | \theta) = \prod_{i=1}^m p(y_i | \theta_i) = \prod_{i=1}^m \binom{n_i}{y_i} \theta_i^{y_i} (1 - \theta_i)^{n_i - y_i}$$

Prior:

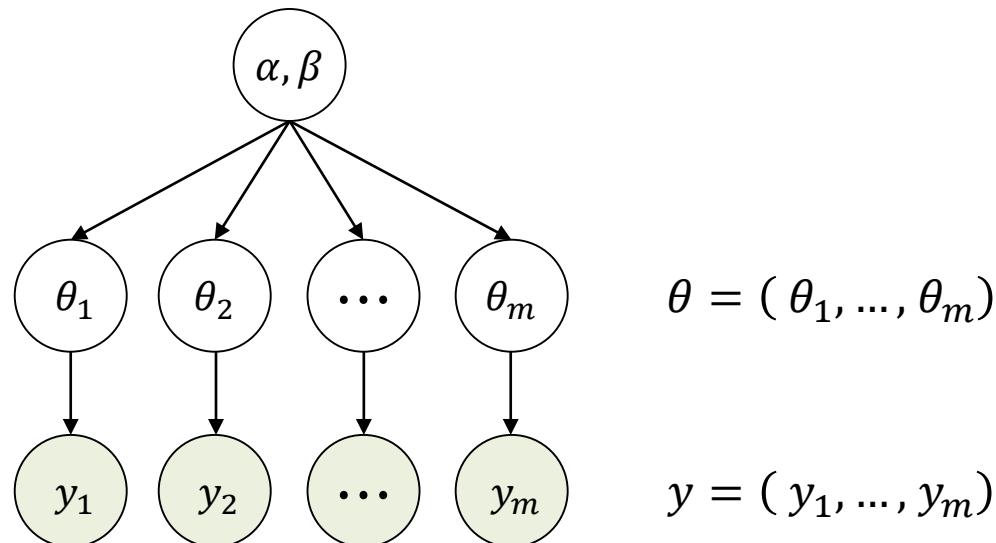
$$\theta_i \sim p(\theta | \alpha, \beta)$$

$$p(\theta_i | \alpha, \beta) = \text{Beta}(\theta_i | \alpha, \beta)$$

Hyper prior:

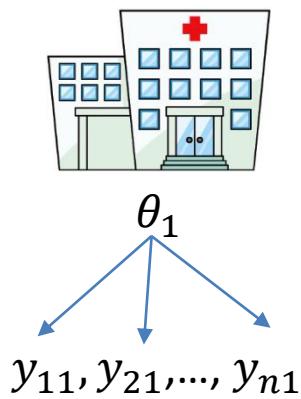
$$(\alpha, \beta) \sim p(\alpha, \beta)$$

$\alpha, \beta$  are random and describes the variability in free-throw percentage across seasons



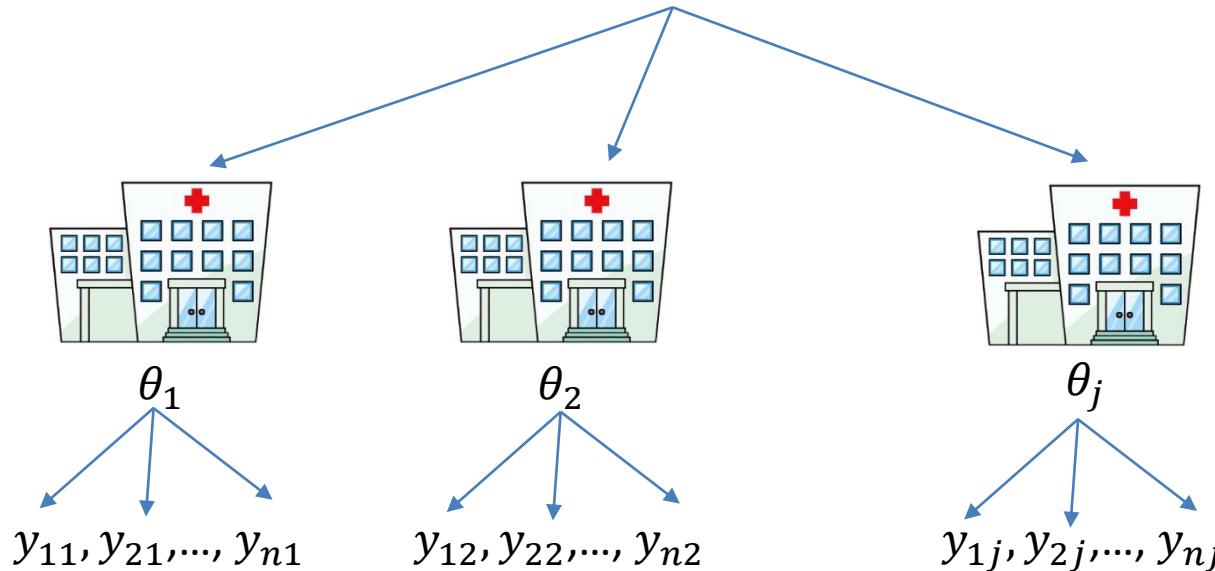
## Why Hierarchical models?

Survival probability of cardiac patients  $\theta_1$



## Why Hierarchical models?

Survival probability of cardiac patients  $\theta_j \sim$  population distribution



- Population distribution is used to structure some dependence into the parameters, thereby avoiding problems of overfitting
- It is natural to model such a problem hierarchically, with observable outcomes modeled conditionally on certain parameters, which themselves are given a probabilistic specification in terms of further parameters, known as hyper-parameters
- Such hierarchical thinking helps in understanding multi-parameter problems and also plays an important role in developing computational strategies

## Why Hierarchical models?

### Nonhierarchical models

- With too small parameters, a model cannot fit large data set  
→ Large Bias
- With too many parameters, a model over fit data set  
→ Poor generalization

### Hierarchical models:

- Can have enough parameters to fit the data well
- Uses population distribution to structure some dependencies into the parameters
  - ✓ Prior knowledge can be encoded into hierarchical structure
  - ✓ Advantageous when only a small data set is available
- Avoid problems of overfitting

## Motivating example : New drug test

- Current experimental result  
4 success from 14 tests
- What is the probability of success?  
$$\frac{4}{14} = 28.6\%$$
- It seems that we only have very small data set

## Motivating example : New drug test

- Current experimental result

4 success from 14 tests

- Historical experimental results

0/20	0/20	0/20	0/20	0/20	0/20	0/20	0/19	0/19	0/19
0/19	0/18	0/18	0/17	1/20	1/20	1/20	1/20	1/19	1/19
1/18	1/18	2/25	2/24	2/23	2/20	2/10	2/20	2/20	2/20
2/20	1/10	5/49	2/19	5/46	3/27	2/17	7/49	7/47	3/20
3/20	2/13	9/48	10/50	4/20	4/20	4/20	4/20	4/20	4/20
4/20	10/48	4/19	4/19	4/19	5/22	11/46	12/49	5/20	5/20
6/23	5/19	6/22	6/20	6/20	6/20	16/52	15/47	15/46	9/24

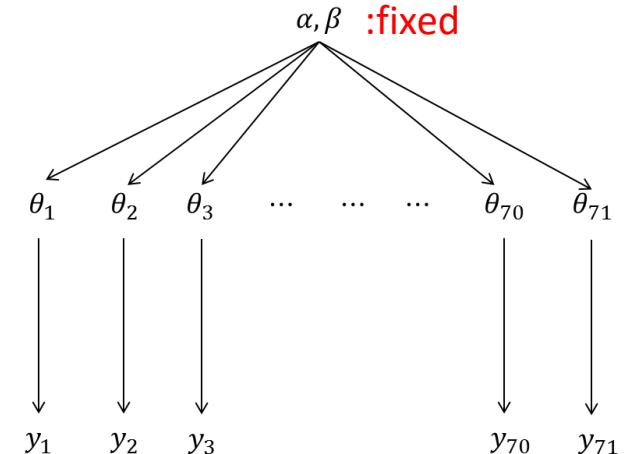
- ✓ The observed sample mean of 70 values  $\frac{y_j}{n_j} = 0.136$
- ✓ The observed sample standard deviation : 0.103

## Motivating example : Drug test

- Historical experimental results

0/20	0/20	0/20	0/20	0/20	0/20	0/19	0/19	0/19
0/19	0/18	0/18	0/17	1/20	1/20	1/20	1/19	1/19
1/18	1/18	2/25	2/24	2/23	2/20	2/20	2/20	2/20
2/20	1/10	5/49	2/19	5/46	3/27	2/17	7/49	7/47
3/20	2/13	9/48	10/50	4/20	4/20	4/20	4/20	4/20
4/20	10/48	4/19	4/19	4/19	5/22	11/46	12/49	5/20
6/23	5/19	6/22	6/20	6/20	6/20	16/52	15/47	15/46
						9/24		

- The observed sample mean of 70 values  $\frac{y_j}{n_j} = 0.136$
- The observed sample standard deviation : 0.103



- Assume a success rate  $\theta$  for each experiment follows Beta distribution

$$\theta \sim \text{Beta}(\alpha, \beta)$$

$$E(\theta) = \frac{\alpha}{\alpha + \beta}$$

$$\text{var}(\theta) = \frac{\alpha\beta}{(\alpha + \beta)^2(\alpha + \beta + 1)}$$

Estimated parameters  
 $(\alpha, \beta) = (1.4, 8.6)$

- Now, we have prior distribution that is empirically estimated from data

$$p(\theta) = \text{Beta}(\theta | 1.4, 8.6)$$

## Motivating example : Drug test

- Historical experimental results

0/20	0/20	0/20	0/20	0/20	0/20	0/19	0/19	0/19
0/19	0/18	0/18	0/17	1/20	1/20	1/20	1/19	1/19
1/18	1/18	2/25	2/24	2/23	2/20	2/20	2/20	2/20
2/20	1/10	5/49	2/19	5/46	3/27	2/17	7/49	7/47
3/20	2/13	9/48	10/50	4/20	4/20	4/20	4/20	4/20
4/20	10/48	4/19	4/19	4/19	5/22	11/46	12/49	5/20
6/23	5/19	6/22	6/20	6/20	6/20	16/52	15/47	15/46
						9/24		

- The observed sample mean of 70 values  $\frac{y_j}{n_j} = 0.136$
- The observed sample standard deviation : 0.103

- Now, we have prior distribution that is empirically estimated from data

$$p(\theta) = \text{Beta}(\theta|1.4, 8.6)$$

- Likelihood of the current observation (4 success from 14 tests)

$$p(y|\theta) = \text{Bin}(4, 14)$$

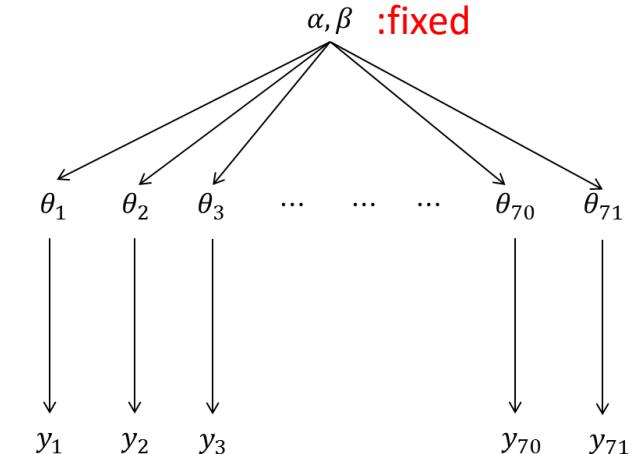
- Posterior distribution of the current experiment results

$$p(\theta|y) = \text{Beta}(5.4, 18.6)$$

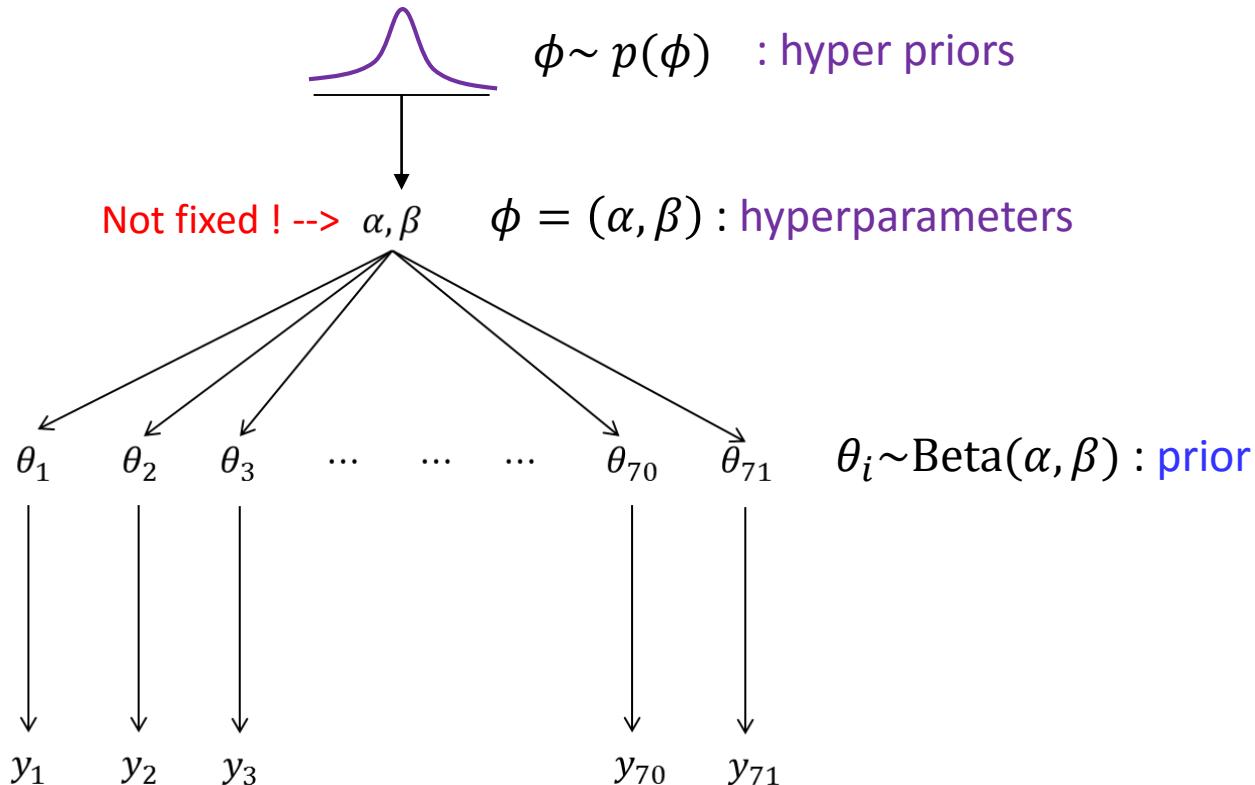
$$\begin{aligned} E(\theta|y) &= 0.223 < \frac{y_{71}}{n_{71}} = \frac{4}{14} = 0.286 \\ \text{var}(\theta|y) &= 0.083 \end{aligned}$$

- This is an empirical Bayes analysis

→ The point estimation on  $\alpha, \beta$  is arbitrary, and the point estimates ignore some uncertainties



## The full Bayesian treatment of the hierarchical model



The key characteristics of hierarchical Bayesian model is that  $\phi$  is not known and thus has its own prior distribution  $p(\phi)$

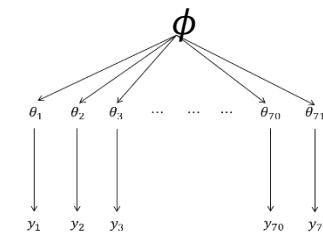
$$\begin{aligned} p(\phi, \theta | y) &\propto p(y|\phi, \theta)p(\phi, \theta) \\ &= p(y|\phi, \theta)p(\theta|\phi)p(\phi) \\ &= p(y|\theta)p(\theta|\phi)p(\phi) \end{aligned}$$

- This model include the **uncertainty** in hyperparameters  $\phi$
- The hyper parameter  $\phi$  affects  $y$  only through parameters

## Exchangeability and hierarchical models

- In order to create a joint probability model for all the parameters  $\theta = (\theta_1, \dots, \theta_J)$ , we use the crucial idea of **exchangeability**
- The parameters  $(\theta_1, \dots, \theta_J)$  are **exchangeable** in their joint distribution if  $p(\theta_1, \dots, \theta_J)$  is invariant to permutations of the indexes  $(1, \dots, J)$
- The simplest form of an exchangeable distribution has each of the parameters  $\theta_j$  as an independent sample from a prior (or population) distribution governed by some unknown parameter vector  $\phi$ ; thus,

$$p(\theta|\phi) = \prod_{j=1}^J p(\theta_j|\phi)$$



- In general,  $\phi$  is unknown, so the distribution for  $\theta$  must average over the uncertainty in  $\phi$  :

$$p(\theta) = \int \left( \prod_{j=1}^J p(\theta_j|\phi) \right) p(\phi) d\phi$$

- **De Finetti's theorem** said any suitably well-behaved exchangeable distribution on  $(\theta_1, \dots, \theta_J)$  can be expressed as a mixture of independent and identical distributions
- Statistically, the mixture model characterizes parameters  $\theta$  as drawn from a common ‘superpopulation’ that is determined by the unknown hyperparameters,  $\phi$

## Exchangeability and hierarchical models

Assume  $(y_1, y_2, \dots)$  are infinitely exchangeable, then by de Finetti's theorem for the  $(y_1, \dots, y_n)$  that you actually observed, there exists

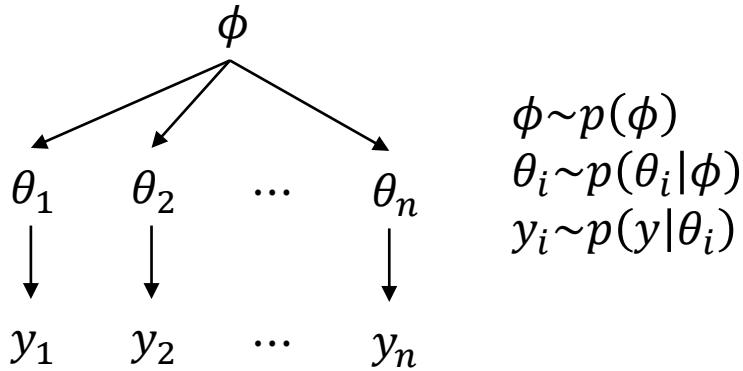
- A parameter  $\theta$
- A distribution  $p(y|\theta)$  such that  $y_j \sim p(y|\theta)$
- A distribution  $p(\theta)$

Assume  $(\theta_1, \theta_2, \dots)$  are infinitely exchangeable, then by de Finetti's theorem for the  $(\theta_1, \dots, \theta_n)$  that you actually observed, there exists

- A parameter  $\phi$
- distribution  $p(\theta|\phi)$  such that  $\theta_j \sim p(\theta|\phi)$
- A distribution  $p(\phi)$

Assume  $\phi = \phi$  with  $\phi \sim p(\phi)$

## Procedure of inferencing for Bayesian hierarchical model



- $y_i$  is observed
- $\theta = (\theta_1, \dots, \theta_n)$  and  $\phi$  are parameters
- only  $\phi$  has a prior that is set

- The joint posterior distribution of interest in hierarchical models is

$$p(\phi, \theta | y) \propto p(y | \phi, \theta) p(\phi, \theta) = p(y | \theta) p(\theta | \phi) p(\phi)$$

- We may be focused on *marginal posteriors*

$$p(\theta | y) = \int_{\phi} p(\phi, \theta | y) d\phi \quad \text{or} \quad p(\phi | y) = \int_{\theta} p(\phi, \theta | y) d\theta$$

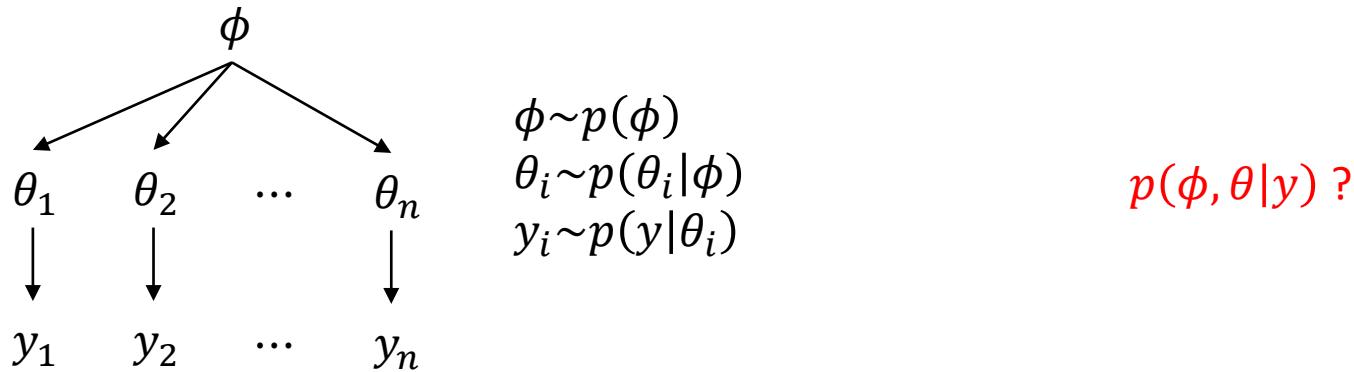
---

This integral is hard to compute due to multiple parameters involved in  $\theta$

- Two posterior predictive distributions are of interest

- ✓ The distribution of future observations  $\tilde{y}$  corresponding to an existing  $\theta_j$
- ✓ The distribution of future observations  $\tilde{y}$  corresponding to future  $\theta_j$  drawn from *hyper-prior*

## Procedure of inferencing for Bayesian hierarchical model



We present an approach that combines analytical and numerical methods to obtain simulations from the joint posterior distribution,  $p(\phi, \theta | y)$ , in the case that the population distribution  $p(\theta | \phi)$  is conjugate to the likelihood  $p(y | \theta)$

## Procedure of inferencing for Bayesian hierarchical model

### Analytical

**Step 1:** Write the *joint posterior density*  $p(\phi, \theta|y)$

$$p(\phi, \theta|y) \propto p(y|\theta)p(\theta|\phi)p(\phi) \quad (\text{un-normalized form})$$

**Step 2:** Determine analytically *the conditional posterior density*  $p(\theta|\phi, y)$

$$p(\theta|\phi, y) = \prod_{j=1}^J p(\theta_j|\phi, y)$$

- ✓  $p(\theta|\phi, y)$  is a distribution on  $\theta$  given  $\phi$  and the fixed data  $y$
- ✓ when  $\phi$  is fixed  $\rightarrow$  single level Bayesian approach can be used, thus easy for conjugate model
- ✓ Conditional posterior distribution is a product of conjugate posterior densities for the components  $\theta_j$

**Step 3:** Obtain *marginal posterior distribution*  $p(\phi|y)$  and estimate  $\phi$

$$p(\phi|y) = \int_{\theta} p(\phi, \theta|y) d\theta \quad \text{or} \quad p(\phi|y) = \frac{p(\phi, \theta|y)}{p(\theta|\phi, y)}$$

Brute force approach

## Drawing simulations from the posterior distribution

### Simulation

By factorization:  $p(\phi, \theta|y) = p(\theta|\phi, y)p(\phi|y)$  (Not Bayesian factorization)

**Step 1:** Draw the vector of hyperparameters  $\phi$  from its marginal posterior distribution,  $p(\phi|y)$

**Step 2:** Draw the parameter vector  $\theta$  from its conditional posterior distribution

$$p(\theta|\phi, y) = \prod_{j=1}^J p(\theta_j | \phi, y)$$

(The components  $\theta_j$  can be drawn independently, one at a time)

**Step 3:** Draw predictive values  $\hat{y}$  from the posterior predictive distribution given the drawn  $\theta$

$$\hat{y} \sim p(\hat{y}|\theta)$$

or draw future observations  $\tilde{y}$  corresponding to future  $\theta_j$  drawn from hyper-prior  $\phi$

Repeat L times, and compute posterior distribution of any estimand or predictive quantity of interest

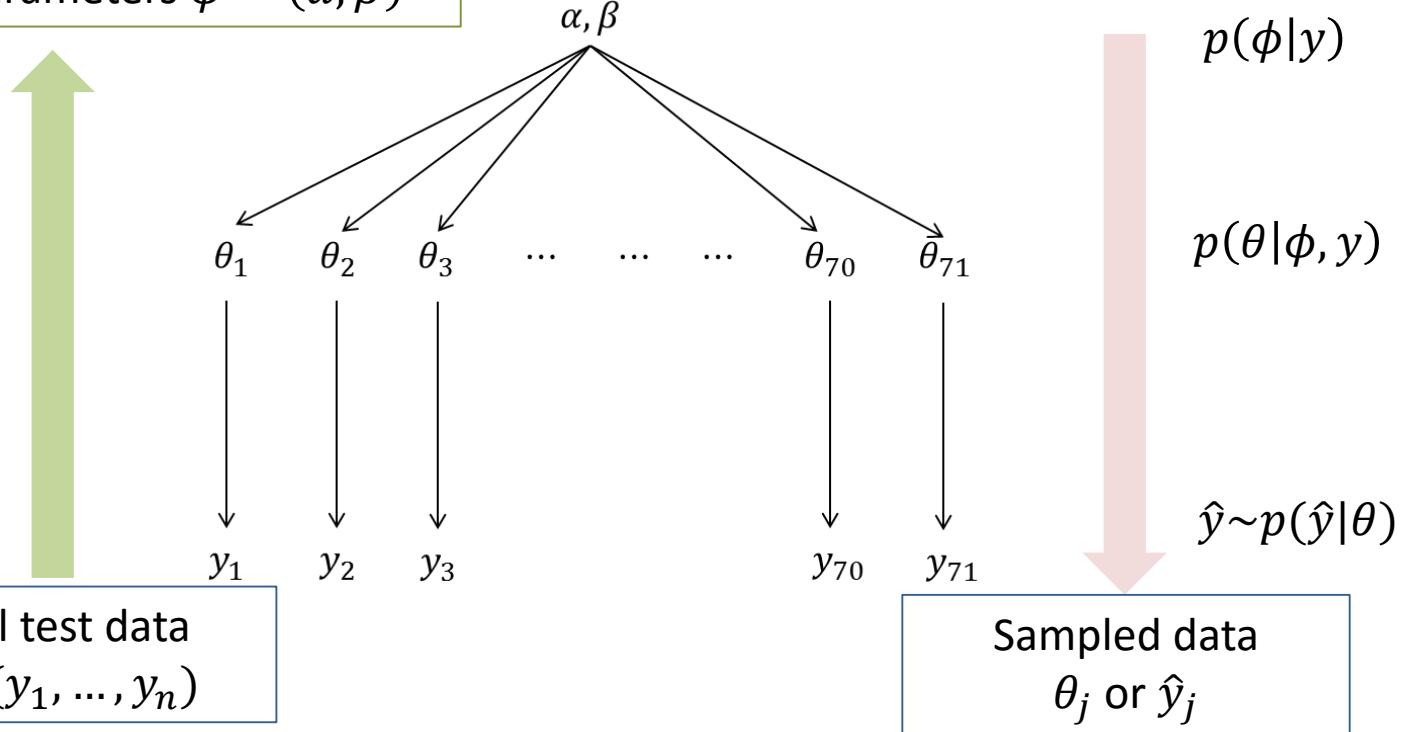
## Bayesian analysis of conjugate hierarchical models-Procedure : Rat tumor example

Analytical approach

$$p(\phi|y)$$

Construct posterior on  
the hyper parameters  $\phi = (\alpha, \beta)$

- $\phi \sim p(\phi)$  : Hyper prior
- $\theta_i \sim \text{Beta}(\alpha, \beta)$  : Prior
- $y_i \sim \text{Bin}(n_j, \theta_j)$  : Likelihood



Simulational approach

## Bayesian analysis of conjugate hierarchical models-Procedure : Rat tumor example

### Analytical

- Models are given:

$$\begin{aligned}(\alpha, \beta) &\sim p(\alpha, \beta) && \text{: hyper prior} \\ \theta_j &\sim \text{Beta}(\alpha, \beta) && \text{: Prior} \\ y_j &\sim \text{Bin}(n_j, \theta_j) && \text{: sampling distribution}\end{aligned}$$

- Step 1** (joint posterior distribution):

$$\begin{aligned}p(\theta, \alpha, \beta | y) &\propto p(\alpha, \beta) p(\theta | \alpha, \beta) p(y | \theta, \alpha, \beta) \\ &\propto p(\alpha, \beta) \prod_{j=1}^J \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta_j^{\alpha-1} (1 - \theta_j)^{\beta-1} \prod_{j=1}^J \theta_j^{y_j} (1 - \theta_j)^{n_j - y_j}\end{aligned}$$

- Step 2** (conditional posterior distribution):

$$p(\theta | \alpha, \beta, y) = \prod_{j=1}^J p(\theta_j | \alpha, \beta, y_j) = \prod_{j=1}^J \frac{\Gamma(\alpha + \beta + n_j)}{\Gamma(\alpha + y_j)\Gamma(\beta + n_j - y_j)} \theta_j^{\alpha+y_j-1} (1 - \theta_j)^{\beta+n_j-y_j-1}$$

(Given the hyper-parameters  $(\alpha, \beta)$ , it is just a single layer Bayesian posterior)

- Step 3** (marginal posterior distribution):

$$p(\alpha, \beta | y) = \frac{p(\theta, \alpha, \beta | y)}{p(\theta | \alpha, \beta, y)} \propto p(\alpha, \beta) \prod_{j=1}^J \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \frac{\Gamma(\alpha + y_j)\Gamma(\beta + n_j - y_j)}{\Gamma(\alpha + \beta + n_j)}$$

## Bayesian analysis of conjugate hierarchical models-Procedure : Rat tumor example

### Sampling Approach

- Models are given:

$$\begin{array}{ll} (\alpha, \beta) \sim p(\alpha, \beta) & \text{: hyper prior} \\ \theta_j \sim \text{Beta}(\alpha, \beta) & \text{: Prior} \\ y_j \sim \text{Bin}(n_j, \theta_j) & \text{: sampling distribution} \end{array}$$

Compute the posterior distribution using sampling methods

$$p(\theta, \alpha, \beta | y) = p(\theta | \alpha, \beta, y) p(\alpha, \beta | y)$$

Repeat  $L$  times ( $i = 1, \dots, L$ )

Sample  $(\alpha^{(i)}, \beta^{(i)})$  from

$$p(\alpha, \beta | y) = \frac{p(\theta, \alpha, \beta | y)}{p(\theta | \alpha, \beta, y)} \propto p(\alpha, \beta) \prod_{j=1}^J \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \frac{\Gamma(\alpha + n_j)\Gamma(\beta + n_j - y_j)}{\Gamma(\alpha + \beta + n_j)}$$

Sample  $\theta^{(i)}$  from

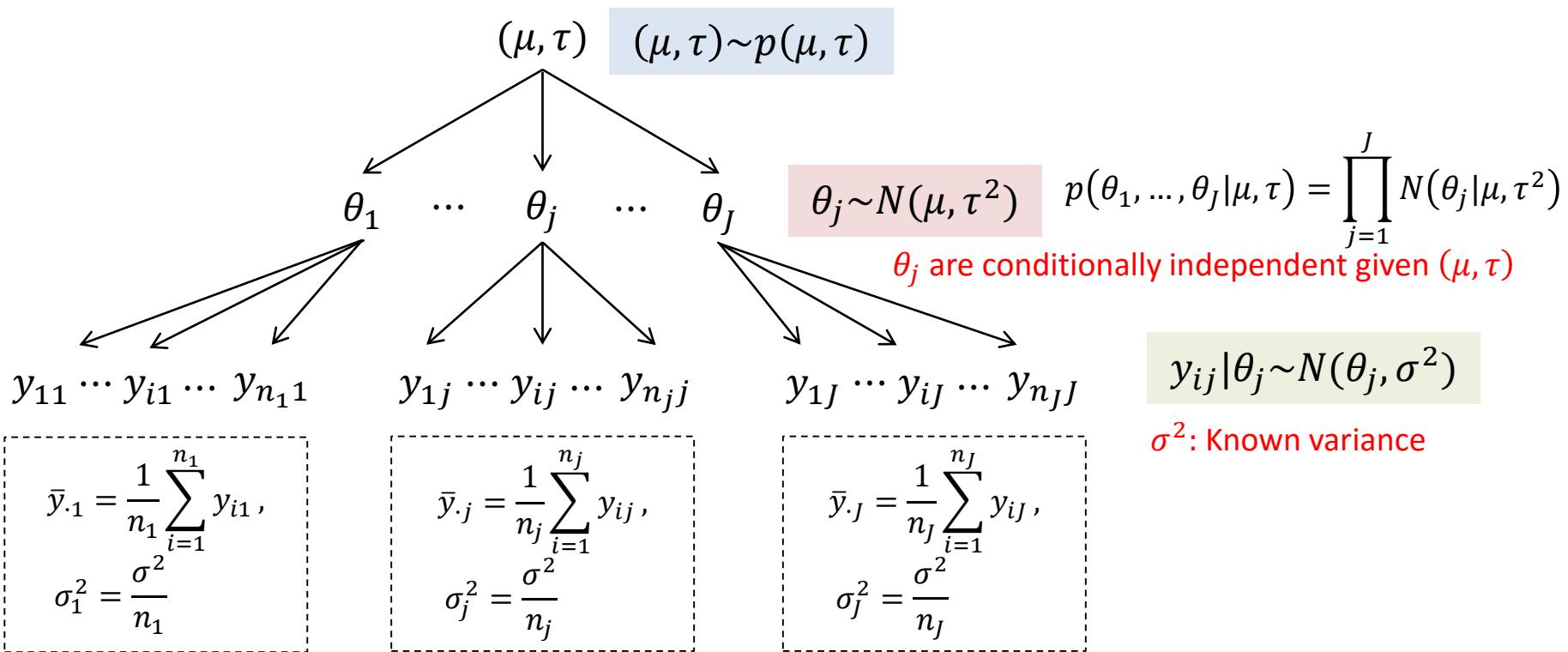
$$p(\theta | \alpha^{(i)}, \beta^{(i)}, y) = \prod_{j=1}^J p(\theta_j | \alpha^{(i)}, \beta^{(i)}, y_j) = \prod_{j=1}^J \frac{\Gamma(\alpha^{(i)} + \beta^{(i)} + n_j)}{\Gamma(\alpha^{(i)} + y_j)\Gamma(\beta^{(i)} + n_j - y_j)} \theta_j^{\alpha^{(i)} + y_j - 1} (1 - \theta_j)^{\beta^{(i)} + n_j - y_j - 1}$$
$$\theta^{(i)} = (\theta_1^{(i)}, \dots, \theta_J^{(i)})$$

We can then have  $L$  samples :  $(\alpha^{(1)}, \beta^{(1)}, \theta^{(1)}), \dots, (\alpha^{(L)}, \beta^{(L)}, \theta^{(L)})$

## Bayesian analysis of conjugate hierarchical models-Procedure : Rat tumor example

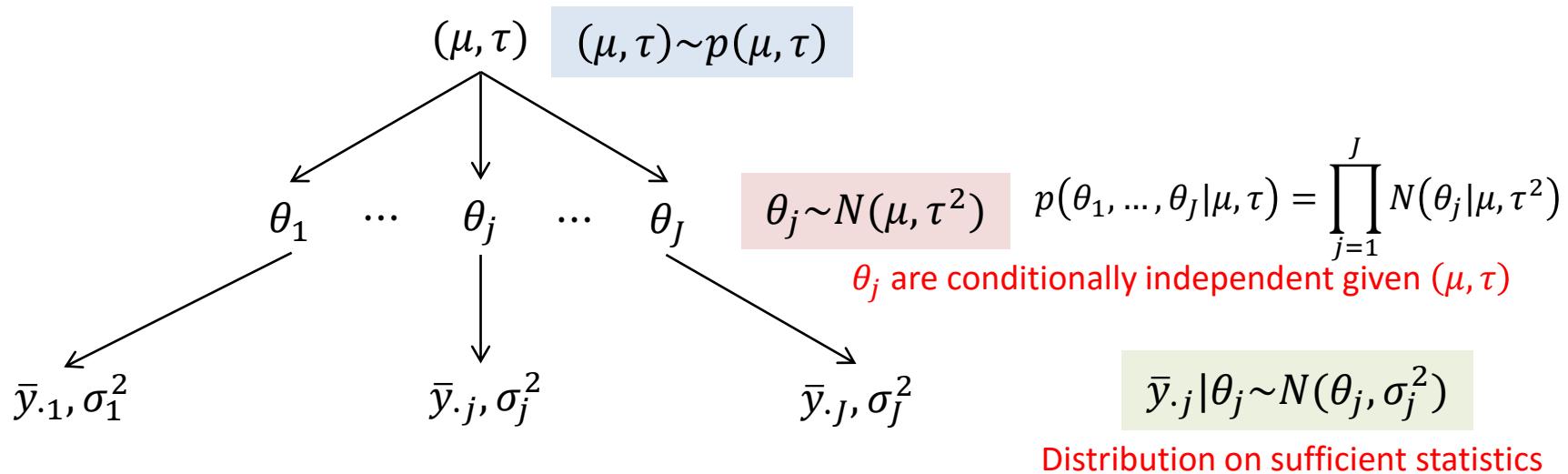
Jupyter Demo Simulation

## Normal model with exchangeable parameters



- Observed data are normally distributed with **a different mean**  $\theta_j$  for each ‘group’ with **known observation variance**  $\sigma^2$
- Normal population distribution for the group mean  $\theta_j \sim N(\mu, \tau^2)$

## Normal model with exchangeable parameters



- Observed data are normally distributed with **a different mean** for each ‘group’ with **known observation variance**
- Normal population distribution for the group mean
- Write the likelihood for each  $\theta_j$  using the sufficient statistics,  $\bar{y}_{\cdot j}$

## Normal model with exchangeable parameters

- Models are given:
  - $(\mu, \tau) \sim p(\mu, \tau) = p(\mu|\tau)p(\tau) \propto p(\tau)$  : hyper prior
  - $\theta_j \sim N(\mu, \tau^2) \rightarrow p(\theta|\mu, \tau) = \prod_{j=1}^J N(\theta_j|\mu, \tau^2)$  : Prior  
**(parameters are conditionally independent given hyper parameters)**
  - $y_{.j} \sim N(\theta_j, \sigma_j^2) \rightarrow p(y|\theta) = \prod_{j=1}^J N(\bar{y}_{.j}|\theta_j, \sigma_j^2)$   
**(data are conditionally independent given parameter)** : sampling distribution

### Analytical

- $p(\theta, \mu, \tau|y)$  : The joint posterior distribution
- $p(\theta|\mu, \tau, y)$  : The conditional posterior distribution
- $p(\mu, \tau|y) \rightarrow p(\mu|\tau, y)p(\tau|y)$  : The marginal posterior distribution

### Sampling

- To derive the posterior computational methods, we factorize the posterior as :  
**(Not Bayesian approach)**

$$(1) \longrightarrow (2) \longrightarrow (3)$$

$$p(\theta, \mu, \tau|y) = p(\tau|y)p(\mu|\tau, y)p(\theta|\mu, \tau, y)$$

## Procedure of inferencing for Bayesian hierarchical model

### Analytical

**Step 1:** Write the ***joint posterior density***  $p(\mu, \tau, \theta | y)$

$$p(\mu, \tau, \theta | y) \propto p(\mu, \tau) p(\theta | \mu, \tau) p(y | \theta) \quad (\text{un-normalized form})$$

$$= p(\mu, \tau) \prod_{j=1}^J N(\theta_j | \mu, \tau^2) \prod_{j=1}^J N(\bar{y}_{\cdot j} | \theta_j, \sigma_j^2) \quad \left( \sigma_j^2 = \frac{\sigma^2}{n_j} \text{ is known} \right)$$

**Step 2:** Determine analytically the ***conditional posterior density***  $p(\theta | \mu, \tau, y)$

$$p(\theta | \mu, \tau, y) = \prod_{j=1}^J p(\theta_j | \mu, \tau, y) = \prod_{j=1}^J N(\theta_j | \hat{\theta}_j, V_j)$$

Recall posterior of Gaussian Prior + Gaussian Likelihood

$$\begin{array}{ccccc} \theta_j \sim N(\mu, \tau^2) & + & y_{ij} \sim N(\theta_j, \sigma^2) & = & \theta_j | \mu, \tau, y \sim N(\hat{\theta}_j, V_j) \\ \text{Prior} & & \text{sampling distribution} & & \text{posterior} \end{array} \quad \hat{\theta}_j = \frac{\frac{1}{\sigma_j^2} \bar{y}_{\cdot j} + \frac{1}{\tau^2} \mu}{\frac{1}{\sigma_j^2} + \frac{1}{\tau^2}} \quad V_j = \frac{1}{\frac{1}{\sigma_j^2} + \frac{1}{\tau^2}}$$

$$\frac{1}{\sigma_j^2} = \frac{n}{\sigma^2}$$

Given hyper parameters, it require one layer posterior distribution

## Procedure of inferencing for Bayesian hierarchical model

### Analytical

**Step 3:** Obtain **marginal posterior distribution**  $p(\mu, \tau|y)$

$$p(\mu, \tau|y) = \int_{\theta} p(\mu, \tau, \theta|y)d\theta \quad \text{or} \quad p(\mu, \tau|y) = \frac{p(\mu, \tau, \theta|y)}{p(\theta|\mu, \tau, y)}$$

For the hierarchical normal model, we can simply consider the information supplied by the data about the hyper parameters directly

$$\begin{aligned} p(\mu, \tau|y) &\propto p(\mu, \tau)p(y|\mu, \tau) \\ &= p(\mu, \tau) \prod_j^J N(\bar{y}_i|\mu, \sigma_j^2 + \tau^2) \quad \because p(\bar{y}_i|\mu, \tau) \sim N(\bar{y}_i|\mu, \sigma_j^2 + \tau^2) \end{aligned}$$

Further factorization  $p(\mu, \tau|y) = p(\mu|\tau, y)p(\tau|y)$

**Step 3-1:** posterior distribution of  $\mu$  given  $\tau$

$$p(\mu|\tau, y) = N(\mu|\hat{\mu}, V_{\mu}) \quad \hat{\mu} = \frac{\sum_{j=1}^J \frac{1}{\sigma_j^2 + \tau^2} \bar{y}_{.j}}{\sum_{j=1}^J \frac{1}{\sigma_j^2 + \tau^2}} \quad \text{and} \quad V_{\mu}^{-1} = \sum_{j=1}^J \frac{1}{\sigma_j^2 + \tau^2}$$

**Step 3-2:** posterior distribution of  $\tau$

$$\begin{aligned} p(\tau|y) &= \frac{p(\mu, \tau|y)}{p(\mu|\tau, y)} \propto \frac{p(\tau) \prod_{j=1}^J N(\mu|\theta_j, \sigma_j^2 + \tau^2)}{N(\mu|\hat{\mu}, V_{\mu})} \\ &\propto p(\tau) V_{\mu}^{\frac{1}{2}} \prod_{j=1}^J (\sigma_j^2 + \tau^2)^{-1/2} \exp\left(-\frac{(\bar{y}_{.j} - \hat{\mu})^2}{2(\sigma_j^2 + \tau^2)}\right) \end{aligned}$$

## Computation of Posterior

### Simulation

Different factorization for simulation:

$$\begin{aligned} p(\theta, \mu, \tau | y) &= p(\theta | \mu, \tau, y) p(\mu, \tau | y) \\ p(\theta, \mu, \tau | y) &= p(\theta | \mu, \tau, y) p(\mu | \tau, y) p(\tau | y) \\ (3) &\leftarrow (2) \leftarrow (1) \end{aligned}$$

#### Step 1

$$p(\tau | y) = \frac{p(\mu, \tau | y)}{p(\mu | \tau, y)} \propto \frac{p(\tau) \prod_{j=1}^J N(\mu | \theta_j, \sigma_j^2 + \tau^2)}{N(\hat{\mu}, V_\mu)} \propto p(\tau) V_\mu^{\frac{1}{2}} \prod_{j=1}^J (\sigma_j^2 + \tau^2)^{-1/2} \exp\left(-\frac{(\bar{y}_{\cdot j} - \hat{\mu})^2}{2(\sigma_j^2 + \tau^2)}\right)$$

#### Step 2

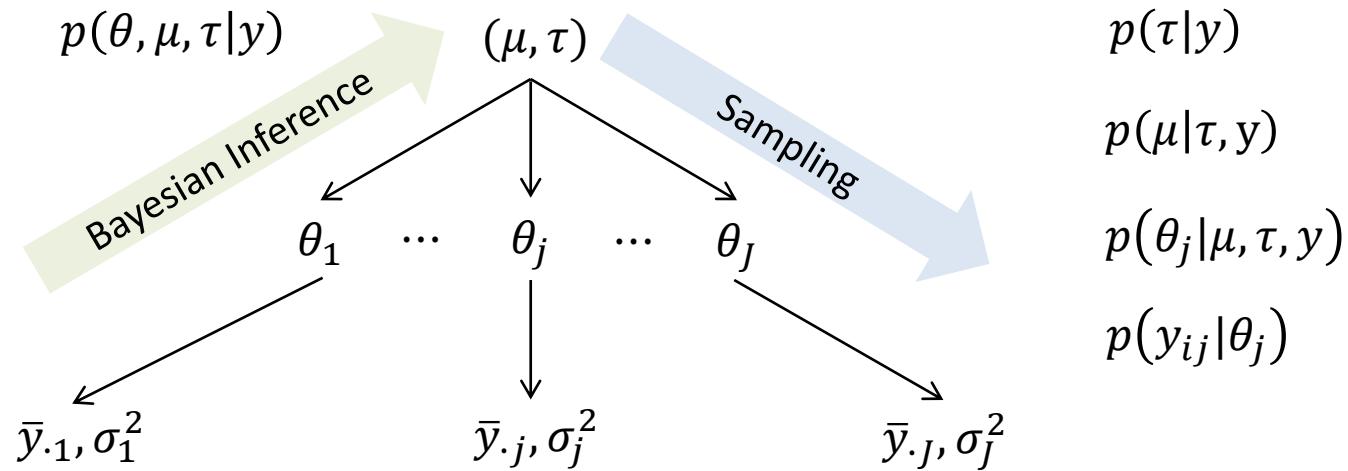
$$p(\mu | \tau, y) = N(\mu | \hat{\mu}, V_\mu) \quad \hat{\mu} = \frac{\sum_{j=1}^J \frac{1}{\sigma_j^2 + \tau^2} \bar{y}_{\cdot j}}{\sum_{j=1}^J \frac{1}{\sigma_j^2 + \tau^2}} \quad \text{and} \quad V_\mu = \frac{1}{\sum_{j=1}^J \frac{1}{\sigma_j^2 + \tau^2}}$$

#### Step 3

$$p(\theta_j | \mu, \tau, y) \sim N(\hat{\theta}_j, V_j) \quad \hat{\theta}_j = \frac{\frac{1}{\sigma_j^2} \bar{y}_{\cdot j} + \frac{1}{\tau^2} \mu}{\frac{1}{\sigma_j^2} + \frac{1}{\tau^2}} \quad \text{and} \quad V_j = \frac{1}{\frac{1}{\sigma_j^2} + \frac{1}{\tau^2}}$$

## Posterior Predictive distribution

Sampling using given the posterior distribution of the parameters



- Future data  $\tilde{y}$  from the current set of batches, with means  $\theta = (\theta_1, \dots, \theta_J)$ 
  1. First draw  $\theta = (\theta_1, \dots, \theta_J)$  from  $p(\theta, \mu, \tau | y)$  and sample data  $\tilde{y}$  using  $p(y_{ij} | \theta_j)$
- Future data  $\tilde{y}$  from  $\tilde{J}$  future batches, with means  $\tilde{\theta} = (\tilde{\theta}_1, \dots, \tilde{\theta}_{\tilde{J}})$ 
  1. First draw  $(\mu, \tau)$  from  $p(\theta, \mu, \tau | y)$
  2. Second draw  $\tilde{J}$  new parameters  $\tilde{\theta} = (\tilde{\theta}_1, \dots, \tilde{\theta}_{\tilde{J}})$  from  $p(\tilde{\theta}_j | \mu, \tau)$
  3. Third, draw  $\tilde{y}$  given  $\tilde{\theta}$  from the data distribution  $p(y_{ij} | \tilde{\theta}_j)$

## Example: Parallel experiments in eight schools

School	Estimated treatment effect, $y_j$	Standard error of estimate, $\sigma_j$
A	28	15
B	8	10
C	-3	16
D	7	11
E	-1	9
F	1	11
G	18	10
H	12	18

- A study was performed for the Educational Testing Service to analyze the effects of special coaching programs on test scores in eight schools.
- There was no prior reason to believe that any of the eight programs was more effective than any other or that some were more similar in effect to each other than to any other
- The estimates  $y_j$  and  $\sigma_j$  are obtained by independent experiments

## Example: Parallel experiments in eight schools

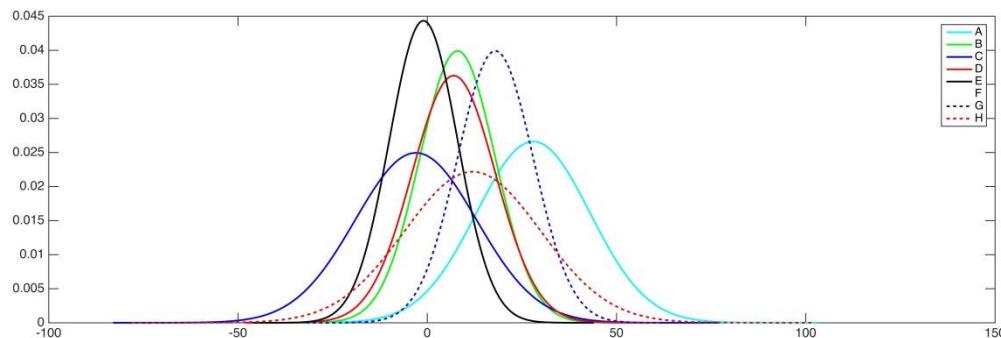
### Independent estimate

- The effects of special coaching programs on test scores.
- The estimates  $y_j$  are obtained by independent experiments and have approximately normal distribution

School	Estimated treatment effect, $y_j$	Standard error of effect estimate, $\sigma_j$
A	28	15
B	8	10
C	-3	16
D	7	11
E	-1	9
F	1	11
G	18	10
H	12	18

$$\begin{array}{c} \theta_1 \dots \theta_j \dots \theta_8 \\ \searrow \quad \downarrow \quad \swarrow \\ y_j, \sigma_j^2 \qquad y_j, \sigma_j^2 \qquad y_8, \sigma_8^2 \qquad y_j \sim N(\theta_j, \sigma_j^2) \end{array}$$

$\sigma^2$ : Known variance



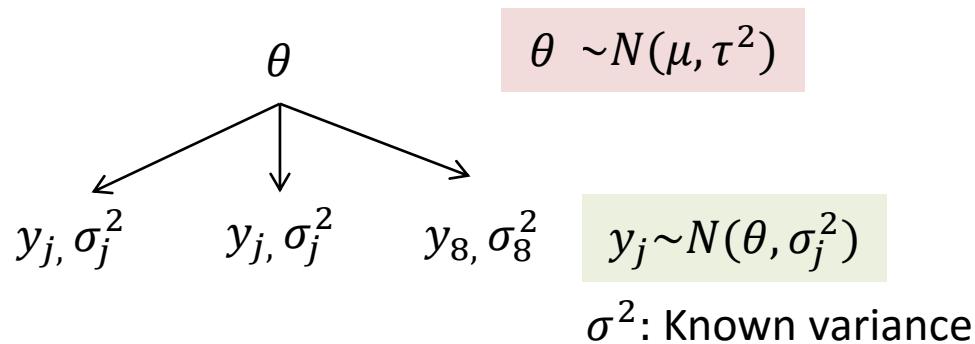
$$p(A > 28 | \theta) = 0.5?$$

## Example: Parallel experiments in eight schools

### Pooled estimate

- All experiments have the same effect and produce independent estimates of this common effect
- Treat the data as eight normally distributed observations with known variances

School	Estimated treatment effect, $y_j$	Standard error of effect estimate, $\sigma_j$
A	28	15
B	8	10
C	-3	16
D	7	11
E	-1	9
F	1	11
G	18	10
H	12	18



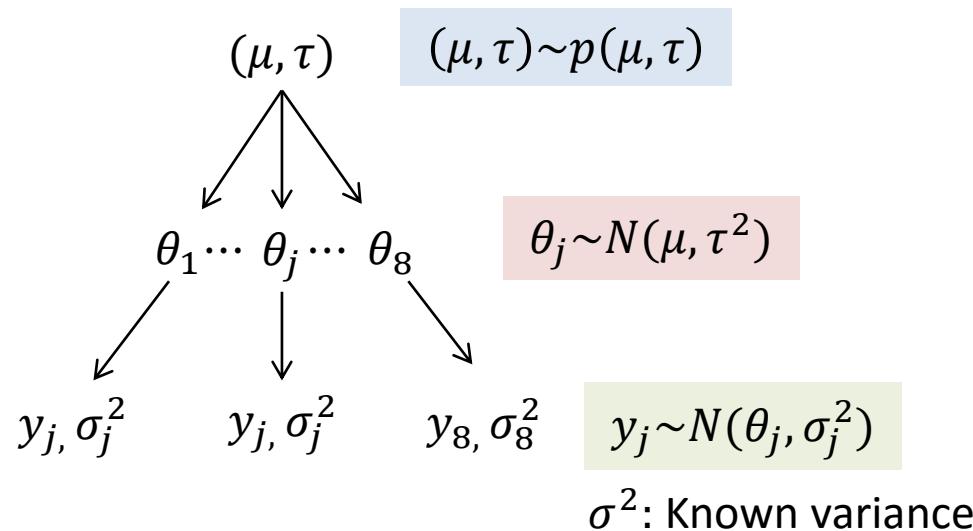
$$\mu = \frac{\sum_{j=1}^J \frac{1}{\sigma_j^2} y_j}{\sum_{j=1}^J \frac{1}{\sigma_j^2}} = 7.7 \quad \tau^2 = \frac{1}{\sum_{j=1}^J \frac{1}{\sigma_j^2}} = 16.6, \sigma = 4.1$$

$$p(A < 7.7 | \theta) = 0.5?$$

## Example: Parallel experiments in eight schools

### Hierarchical Bayesian Model

We would like a compromise that combines information from all eight experiments without assuming all the  $\theta_j$ 's to be equal.



## Example: Parallel experiments in eight schools

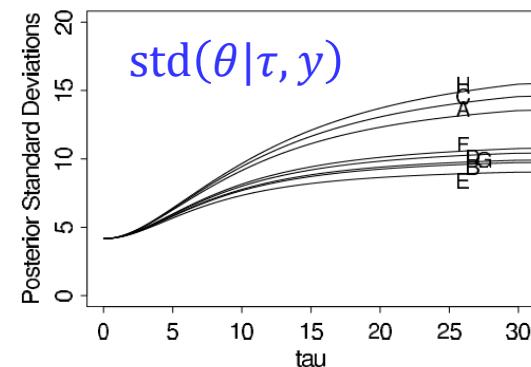
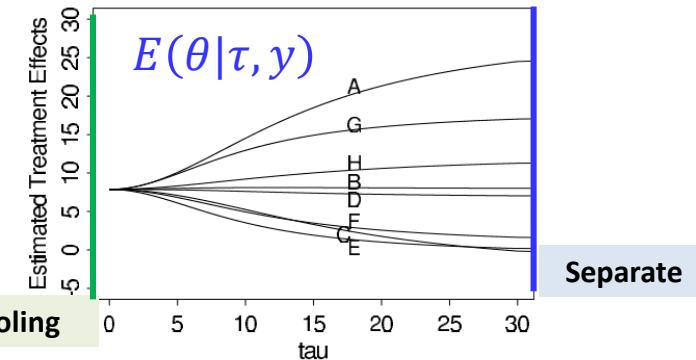
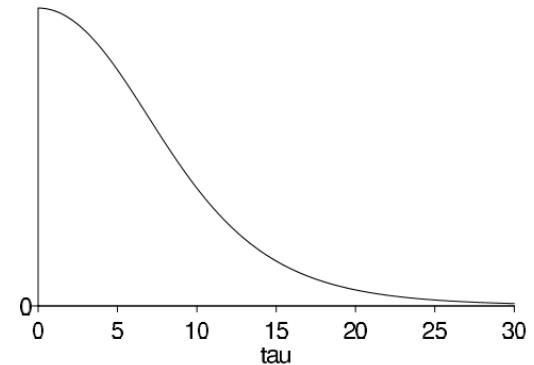
### Sampling procedure

$$p(\tau|y) = \frac{p(\mu, \tau|y)}{p(\mu|\tau, y)} \propto \frac{p(\tau) \prod_{j=1}^J N(\mu|\theta_j, \sigma_j^2 + \tau^2)}{N(\hat{\mu}, V_\mu)}$$

 Sample  $\tau$  given data  $y$

$$\begin{aligned} p(\theta|\tau, y) &= \int_{\mu} p(\theta, \mu|\tau, y) d\mu \\ &= \int_{\mu} p(\theta|\mu, \tau, y) p(\mu|\tau, y) d\mu \end{aligned}$$

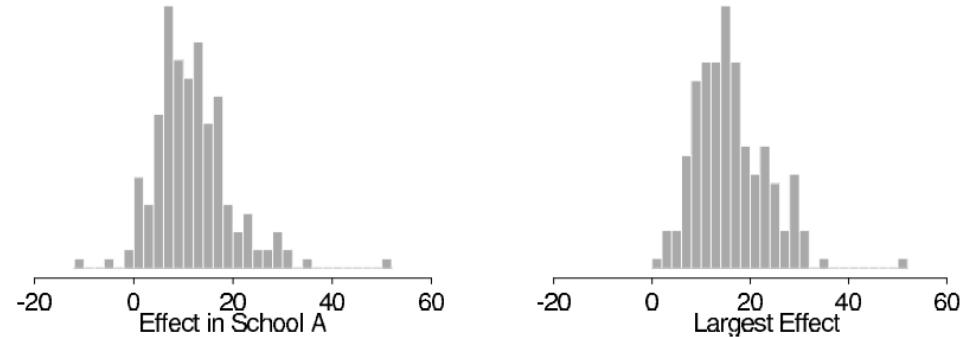
 Sample  $\theta$  given  $\tau$  and data  $y$



## Example: Parallel experiments in eight schools

### Results from the posterior distribution estimated using the sampled data

School	Posterior quantiles				
	2.5%	25%	median	75%	97.5%
A	-2	7	10	16	31
B	-5	3	8	12	23
C	-11	2	7	11	19
D	-7	4	8	11	21
E	-9	1	5	10	18
F	-7	2	6	10	28
G	-1	7	10	15	26
H	-6	3	8	13	33



- The Bayesian probability that the effect in school A is as large as 28 points:  
Individual test: 50% → Bayesian : less than 10%
- What is the maximum  $\{\theta_j\}$ ?
- What is the maximum  $\Pr(\theta_j > \theta_j | y)$ ?

Hierarchical model is flexible enough to adapt to the data, thereby providing posterior inferences that account for the partial pooling as well as uncertainty in the hyper parameters

## Example: Parallel experiments in eight schools

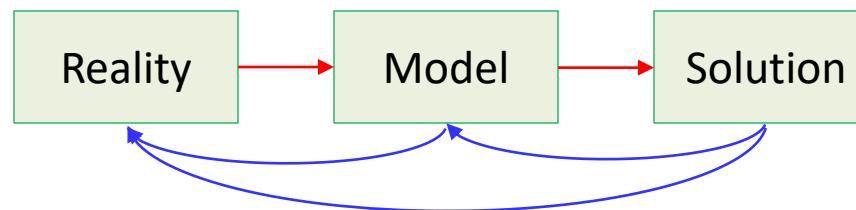
Jupyter Demo Simulation

## L5. Model Checking, Evaluating, Comparing and Expanding models

## Model checking is a crucial step

1. Constructing a probability model
2. Computing the posterior distribution of all estimands
3. Assessing the fit of the model to the data and to our substantive knowledge

It is impossible to include in a probability distribution all of one's knowledge about a problem, and so it is wise to investigate what aspects of reality are not captured by the model



We need to check....

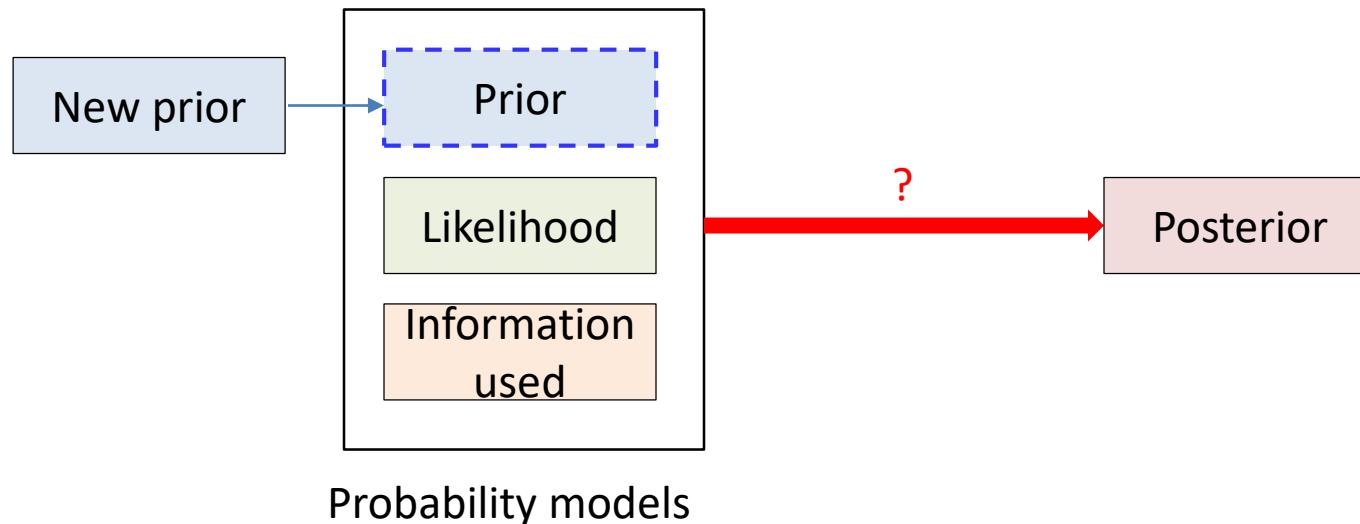
- prior
- sampling distribution
- hierarchical structure
- explanatory variables...



## Sensitivity analysis

### Sensitivity analysis :

how much do posterior inferences change when other reasonable probability models are used in place of the present model?



### We need to examine:

- How our model fail to fit reality
- How sensitive the resulting posterior distributions are to arbitrary specifications

## Sensitivity analysis

Is our model true or false?

V.S.

Do the model's deficiencies have a noticeable effect on the substantive inferences?

- Rat tumor example : Beta population distribution for tumor rates
- Eight schools example : Normal distribution for the eight school effects

→ Little bit arbitrary and convenience chaise, but they have little impact on the inferences of most interests

How to judge when *assumptions of convenience* can be made safely is a central task of Bayesian sensitivity analysis

## Do the inferences from the model make sense?

For reasons of convenience or objectivity,  
there will be knowledge that is not included formally in either the prior or likelihood



If the additional information suggests that posterior inferences of interest are false



Potential for creating a more accurate probability model for the parameters and data collection process

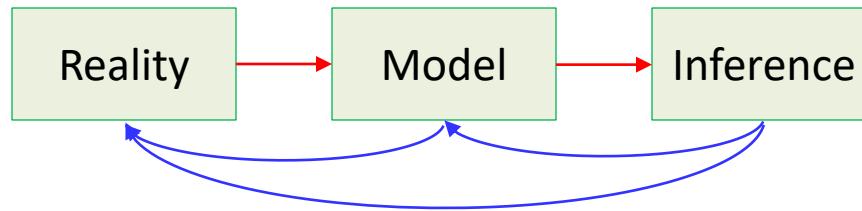
### External validation

- Prediction about future data v.s. collected external data
  - Often we need to check the model before obtaining new data or waiting for the future to happen
- Require a method of approximating external validation using **the data we already have**

#### Posterior predictive checking

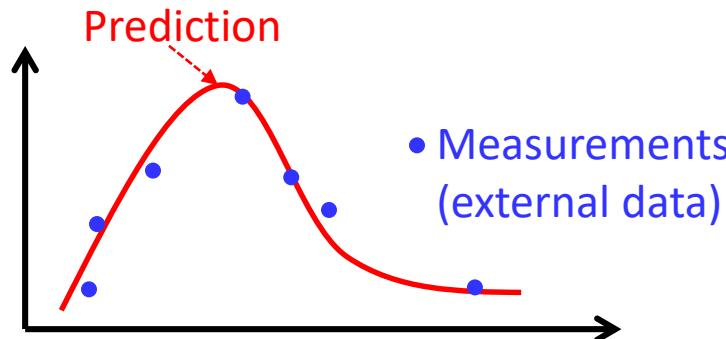
- Use global summaries to check the **joint predictive distribution**  $p(\tilde{y}|y)$

## Do the inferences from the model make sense?



When inference does not capture reality,  
we need more accurate probability model for the parameters and data collection process

### External validation



- Prediction about future data v.s. collected external data
- When there is no external data received yet,  
→ Require a method of approximating external validation using the data we already have

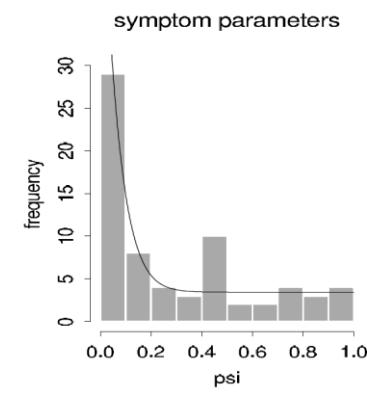
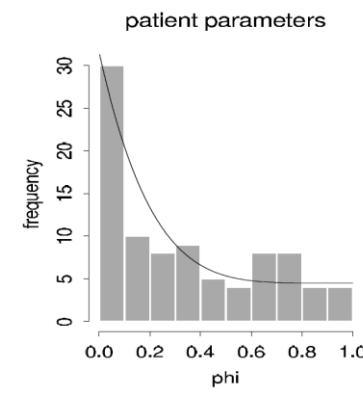
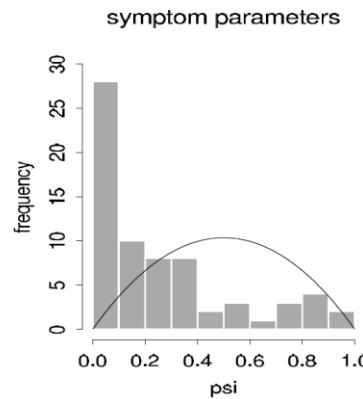
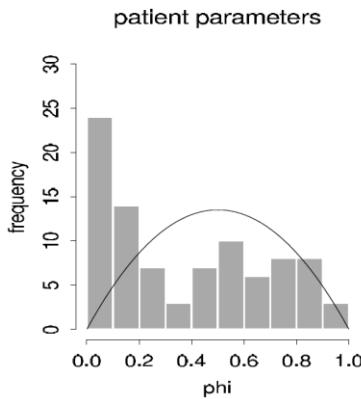
## Two approaches of modal checking

- Posterior predictive checking

$$p(\tilde{y}|y)$$

$\tilde{y} = \tilde{y}_1, \dots, \tilde{y}_n$  : A Joint prediction on future data

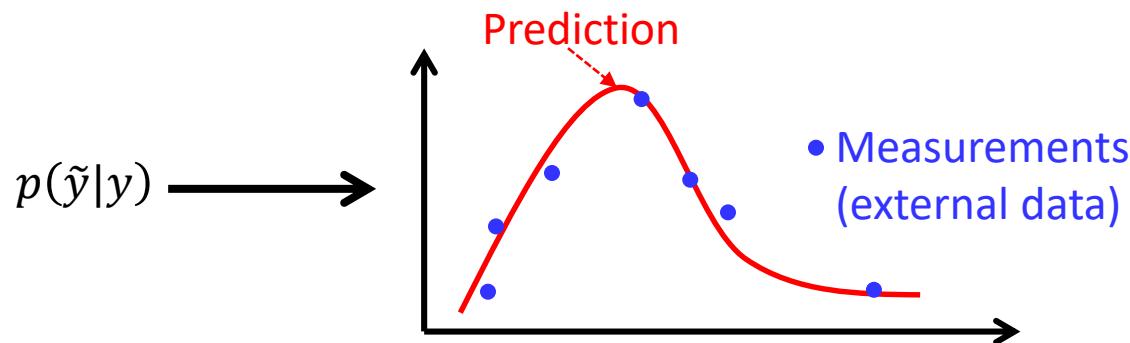
- Graphical Checking



## Posterior predictive checking

### Self-consistency check:

Replicated data generated under the model should look similar to observed data. That is, the observed data should look plausible under the posterior predictive distribution



## Posterior predictive checking

$y^{rep}$  is replication (simulation) of the observed data  $y$  using the trained model

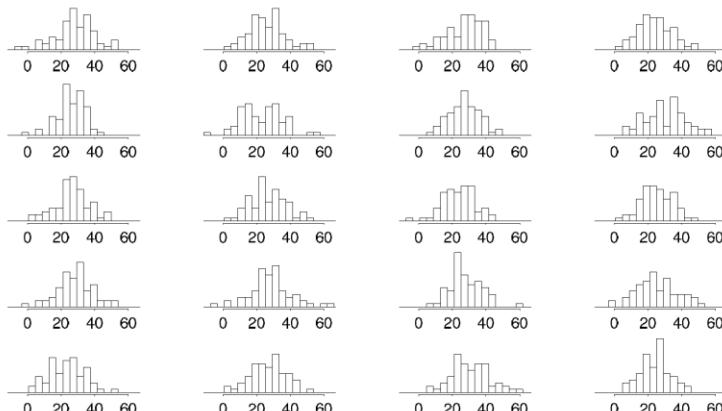


Posterior predictive distribution on the replication is

$$p(y^{rep}|y) = \int p(y^{rep}|\theta)p(\theta|y)d\theta$$

prior	$p(\mu, \sigma^2) \propto (\sigma^2)^{-1}$
Likelihood	$y \mu, \sigma^2 \sim N(\mu, \sigma^2)$

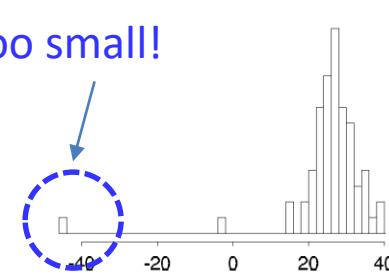
$p(\mu, \sigma^2|y) \rightarrow$  Sample  $(\mu, \sigma^2) \rightarrow$  sample 66  $\tilde{y}_i$



Replicated  $p(y^{rep}|y)$

Is it similar?

Too small!



Actual measurement  $p(y)$

## Posterior predictive checking

$y^{rep}$  is replication (simulation) of the observed data  $y$  using the trained model

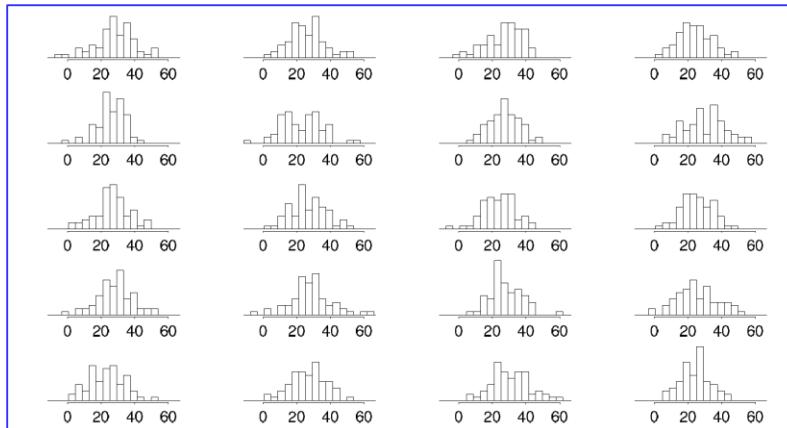


Posterior predictive distribution on the replication is

$$p(y^{rep}|y) = \int p(y^{rep}|\theta)p(\theta|y)d\theta$$

prior	$p(\mu, \sigma^2) \propto (\sigma^2)^{-1}$
Likelihood	$y \mu, \sigma^2 \sim N(\mu, \sigma^2)$

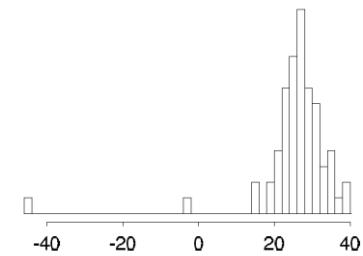
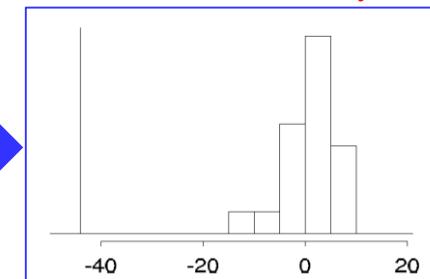
$p(\mu, \sigma^2 | y) \rightarrow$  Sample  $(\mu, \sigma^2) \rightarrow$  sample 66  $\tilde{y}_i$



Replicated  $p(y^{rep}|y)$

Test statistics  $T(y) = \min_i \tilde{y}^i$

$T(y)$



## Posterior predictive checking

### Test quantities

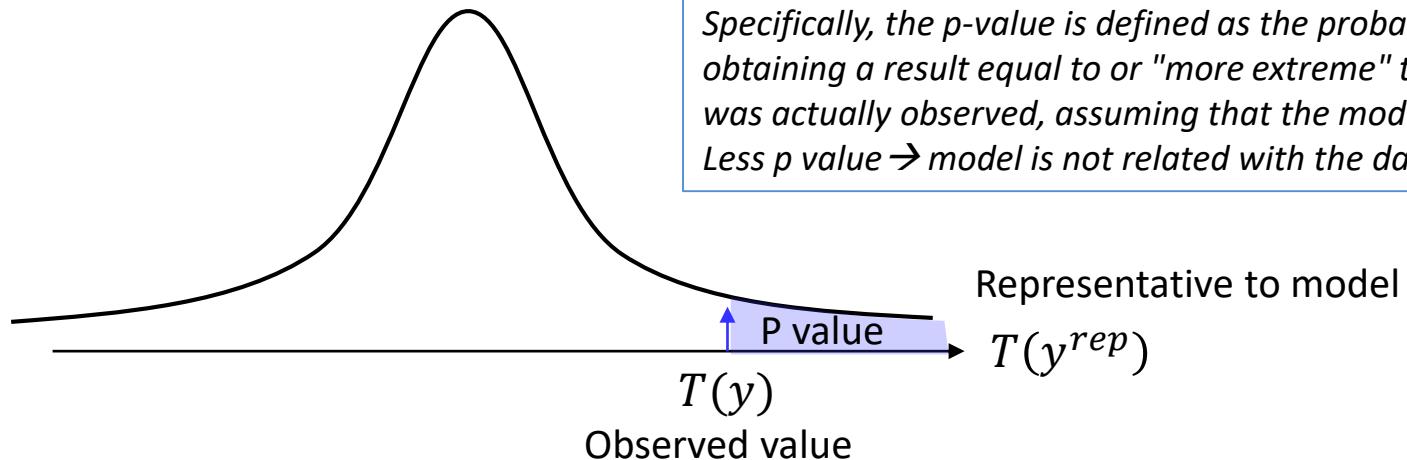
- In cases with less obvious discrepancies than the outliers in the speed of light data, it is often also useful to measure the “**statistical significance**” of the lack of fit
- Measure the discrepancy between model and data by defining *test quantities*
  - ✓ Classical setting :  $T(y)$ , called test statistic
  - ✓ Bayesian setting :  $T(y, \theta)$ , which take into account the dependency on the model parameter  $\theta$

### The procedure for carrying out a posterior predictive model check

1. Specify a test quantity,  $T(y)$  or  $T(y, \theta)$
2. Specify an appropriate predictive distribution for the replications  $y^{rep}$
3. Measure discrepancy between  $T(y)$  and  $T(y^{rep})$ , for example using *p – value*

## Posterior predictive checking

### Tail area probability



Specifically, the  $p$ -value is defined as the probability of obtaining a result equal to or "more extreme" than what was actually observed, assuming that the model is true  
Less  $p$  value → model is not related with the data

- Classical  $p$  – values :  $p_C = Pr(T(y^{rep}) \geq T(y)|\theta)$  : Parameter is fixed  
 $\theta$  is fixed and should be substituted
- Bayesian  $p$  – values :  $p_B = Pr(T(y^{rep}|\theta) \geq T(y|\theta)|y)$  : Data is fixed
$$= \int \int I_{T(y^{rep}|\theta) \geq T(y|\theta)} p(y^{rep}, \theta|y) dy^{rep} d\theta$$
  - ✓ Where the probability is taken over the posterior distribution of  $\theta$  and the posterior predictive distribution of  $y^{rep}$ , i.e.,  $p(\theta, y^{rep}|\theta)$
  - ✓ Usually, the posterior predictive distribution can be computed using simulation : count the number of  $T(y^{rep_s}, \theta^s) \geq T(y, \theta^s)$ ,  $s = 1, \dots, S$

## Posterior predictive p-values

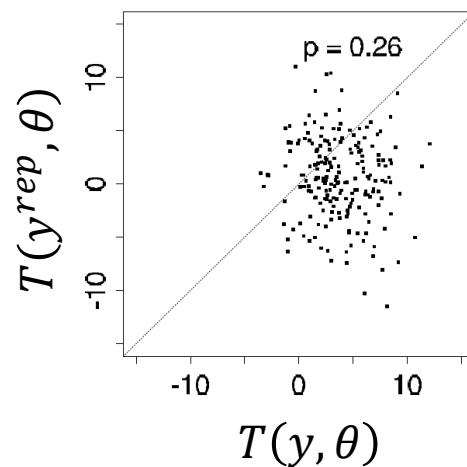
- An extreme p-values implies that the model cannot be expected to capture this aspect of the data.
- Extreme tail-area probabilities (less than 0.01 or higher than 0.99) can be addressed by expanding the model appropriately.
- Typically, we evaluate a model with respect to several test quantities
  - **Not** : Do the data come from the assumed model?
  - **Yes** : Quantify the discrepancies between data and model, and assess whether they could have arisen by chance, under the model's own assumptions.
- Bayesian predictive checking generalizes classical hypothesis testing by averaging over the posterior distribution of the unknown parameter vector  $\theta$  rather than fixing it at some estimate  $\hat{\theta}$ .

## Speed of light example with test quantity

- Use other test quantities to illustrate how the fit of a model depends on the aspects of the data and parameters being monitored.
- Assess whether the model is adequate except for the extreme tails by considering a model check based on a test quantity sensitive to asymmetry in the center of the distribution

$$T(y, \theta) = |y_{61} - \theta| - |y_6 - \theta|$$

- The 61<sup>st</sup> and 6<sup>th</sup> order statistics are chosen to represent approximately the 90% and 10% points of the distribution.
- The test quantity should be scattered about zero for a symmetric distribution.



## Choosing test quantities

### Example: Check the independence assumption

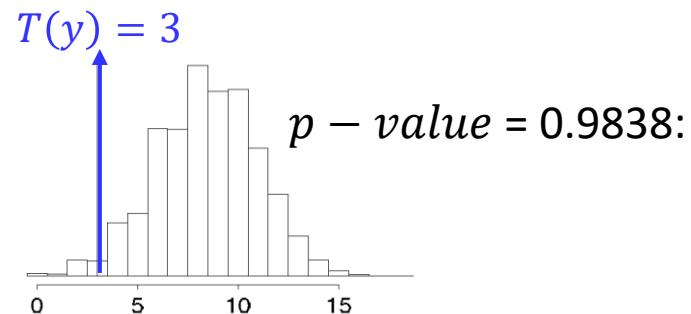
- A sequence of binary outcomes,  $y_1, \dots, y_n$ , modeled as specified number of independent trial with a common probability of success,  $\theta$ , that is given a uniform prior distribution.
- The posterior distribution is  $p(\theta|y) \propto \theta^{\sum_{i=1}^n y_i} (1 - \theta)^{n - \sum_{i=1}^n y_i}$

Perform a posterior predictive test using with

$$T(y) = \text{number of switches between 0 and 1}$$

Observed data:  $y = (y_1, \dots, y_n) = (1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0) \rightarrow T(y) = 3$

1. draw  $\theta^s$  for  $s = 1, \dots, S = 1,0000$
2. for each  $s$  draw  $y^{rep_s} = (y_1^{rep_s}, \dots, y_{20}^{rep_s})$
3. compute  $T(y^{rep_s})$



The discrepancy cannot be explained by chance, but is likely to be explained by modeling failure

## Choosing test quantities

- Posterior predictive checking is a useful direct way of assessing the fit of the model to these various aspects of the data.
- Ideally, the test quantities  $T$  will be chosen to reflect aspects of the model that are relevant to the scientific purposes to which the inference will be applied.
- Test quantities are commonly chosen to measure a feature of the data not directly addressed by the probability model; for example, ranking and correlation

## Interpreting posterior (Bayesian) p-values

- The goal of examining p-value is not to answer the question ‘Do the data come from the assumed model?’, but to quantify the discrepancies between data and model, and assess whether they could have arisen by chance, under the model’s own assumption
- An extreme p-value implies that the model cannot be expected to capture this aspect of the data
- Bayesian predictive checking generalized classical hypothesis testing by averaging over the posterior distribution of the unknown parameter vector  $\theta$  rather than fixing it at some estimate  $\hat{\theta}$
- In Bayesian inference, the data enter the posterior distribution only through the likelihood function; thus, it is sometimes stated as a principle that inferences should depend on the likelihood and no other aspect of the data.

## Marginal predictive checks

- The focus has been on replicated data from the joint posterior predictive distribution  
→ An alternative approach is to compute the probability distribution for each marginal prediction  $p(\tilde{y}_i|y)$  separately and then compare these separate distributions to data in order to find outliers or check overall calibration.

$$p_i = \Pr(T(y_i^{rep}) \leq T(y_i)|y)$$

If  $y_i$  is scalar,

$$p_i = \Pr(y_i^{rep} \leq y_i|y)$$

A related approach is to replace predictive distributions with cross-validation predictive distributions, for each data point comparing to the inference given all the other data.

$$p_i = \Pr(y_i^{rep} \leq y_i|y_{-i})$$

## Graphical posterior predictive checks

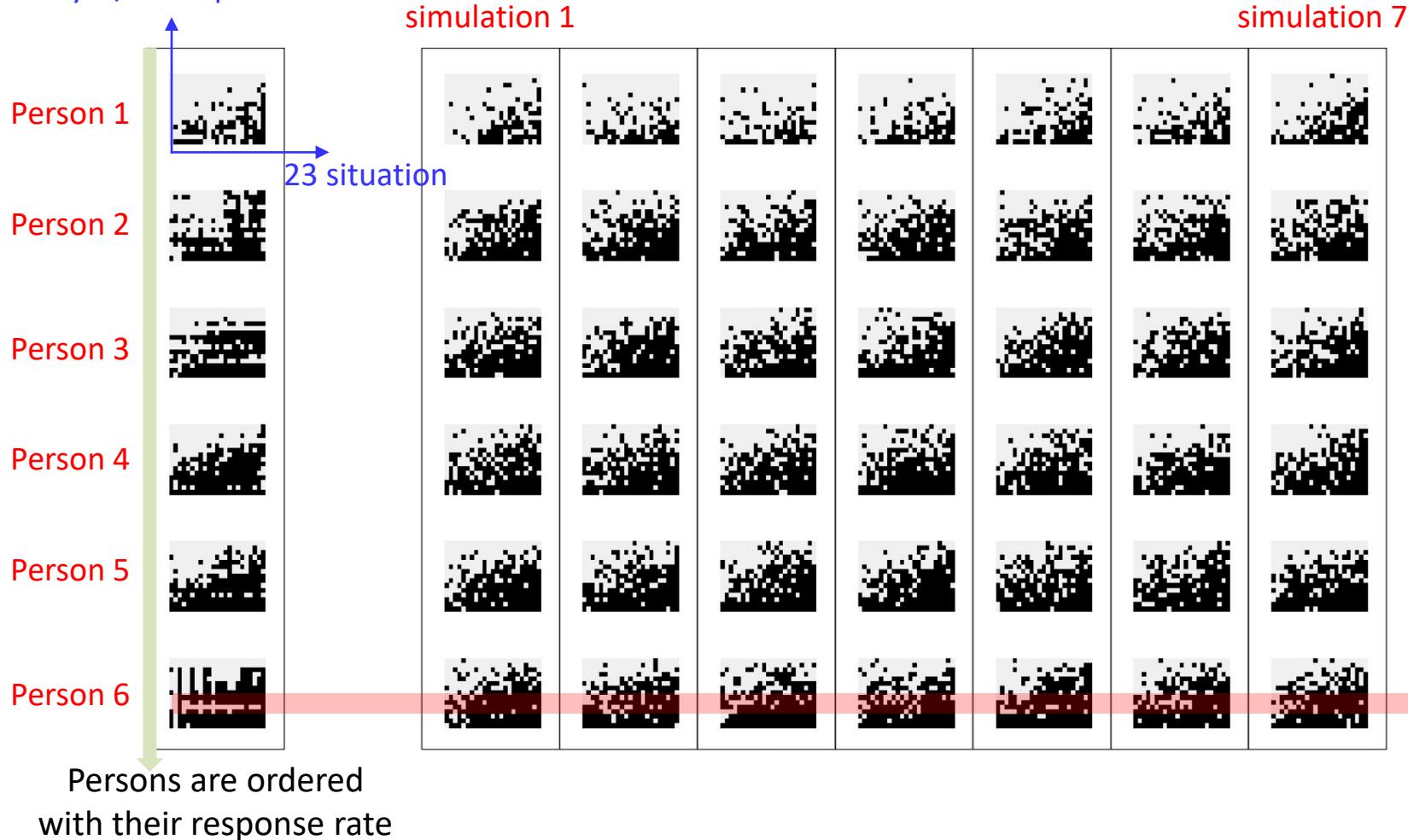
Graphical model checking is to **display the data alongside simulated data from the fitted model**, and to look for systematic discrepancies between real and simulated data from the fitted model, and look for systematic discrepancies between real and simulated data.

- Direct display of all the data
- Display of data summaries or parameter inferences
- Graphs of residuals or other measures of discrepancy between model and data

## Direct display of all the data

### Experiment in Psychology

15 yes/no responses



$$T(y)$$

$$T(y^{rep})$$

Graphical representations with special ordering itself is a test statistics

Model  
misfit

## Displaying summary statistics or inferences

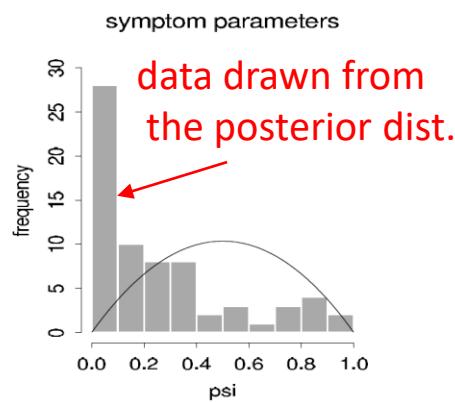
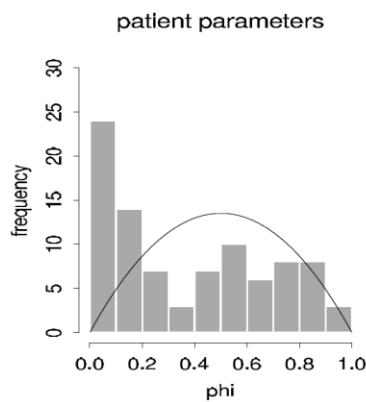
Inference from a hierarchical model from psychology (Model specification is now shown here)

Two vector of parameters:

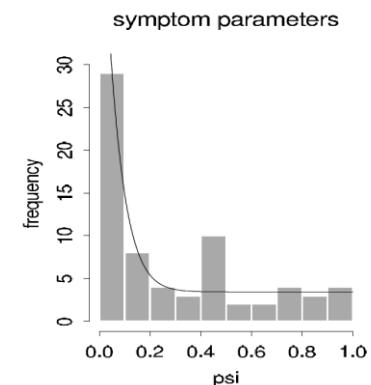
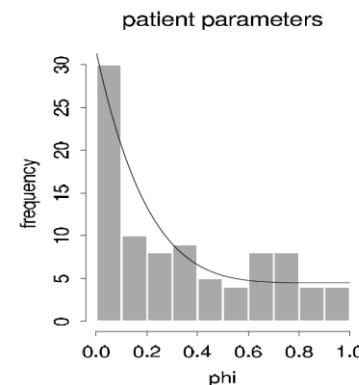
- Patient parameters:  $\phi_1, \dots, \phi_{90}$
- Psychological symptom parameters:  $\psi_1, \dots, \psi_{69}$

$$p(\phi_i) \sim \text{Beta}(\phi_i | 2, 2)$$

$$p(\psi_i) \sim \text{Beta}(\psi_i | 2, 2)$$



$$p(\phi_i) = 0.5\text{Beta}(\phi_i | 1, 6) + 0.5\text{Beta}(\phi_i | 1, 1)$$
$$p(\psi_i) = 0.5\text{Beta}(\psi_i | 1, 16) + 0.5\text{Beta}(\psi_i | 1, 1)$$



Prior distribution can be viewed as a posterior predictive check in which a new set of patients and symptom parameters.

## Residual plots and binned residual plots

- Linear or nonlinear regression model

$$g(x, \theta) = E(y|x, \theta)$$

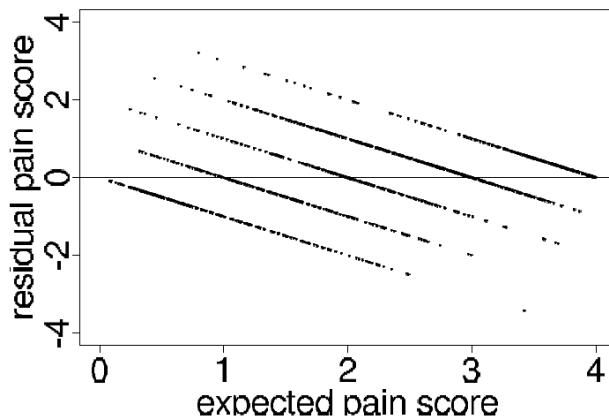
- Given unknown parameter  $\theta$  and the predictors  $x_i$  for a data point  $y_i$ , *the realized residual* is

$$y_i - g(x_i, \theta)$$

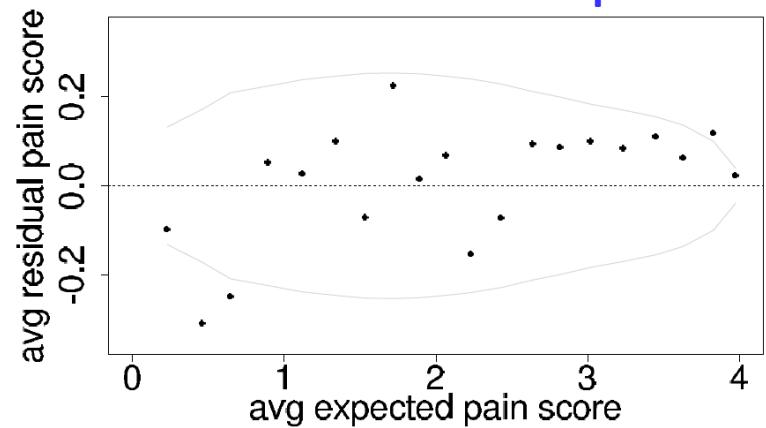
- the *classical residual* is

$$y_i - g(x_i, \hat{\theta})$$

with a fixed point estimate of  $\hat{\theta}$



The binned residual plot



## Graphical model checking for the 8 schools example

### 8 schools example assumptions:

(1) Normality of the estimates  $y_j$  given  $\theta_j$  and  $\sigma_j$  (known)

→ randomization, large sample size,...

(2) Exchangeability of the prior distribution of the  $\theta_j$

→ we will let the data tell us about the relative ordering and similarity of effects in the schools.

→ In the absence of any information, the exchangeability assumption implies that the prior distribution of the  $\theta_j$ 's can be considered as independent samples from a population whose distribution is indexed by some hyper-parameters

(3) Normality of the prior distribution of each  $\theta_j$  given  $\mu$  and  $\tau$

$$\theta_j \sim N(\mu, \tau)$$

(4) Uniformity of the hyper prior distribution of  $(\mu, \tau)$

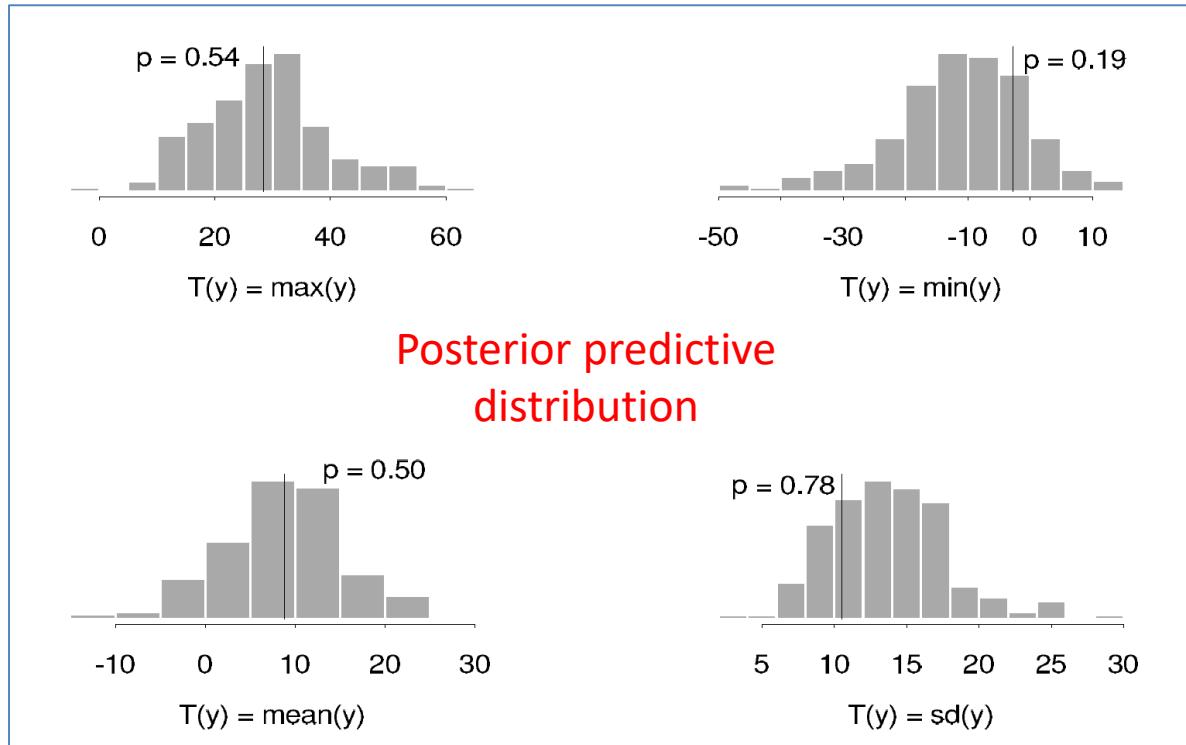
$$(\mu, \tau) \sim 1$$

(3) And (4) are hard to justify (before seeing data)

Mathematical tractability is one reason for the choice of models, but if the family of probability models is inappropriate, Bayesian answers can be misleading

## Posterior predictive checking

**School Coaching Example:** Suppose we perform 200 posterior predictive simulations of the coaching experiments



- The summaries suggest that the model generates predicted results similar to the observed data in the study: that is, the actual observations are typical of the predicted observations generated by the model
- Other reasonable models might provide just as good a fit but lead to different conclusions. **Sensitivity analysis** can then be used to assess the effect of alternative analyses on the posterior inferences.

## L6. Bayesian Decision Analysis

## Bayesian Regression

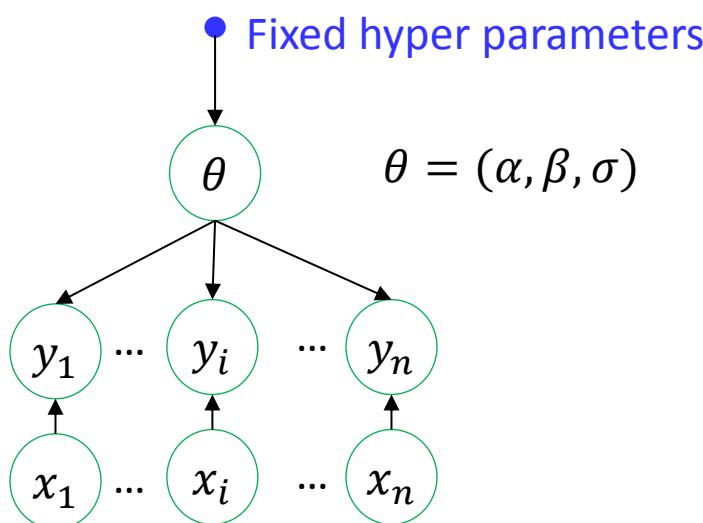
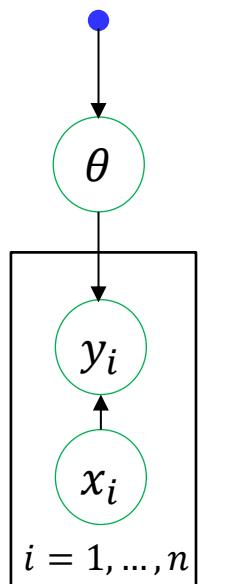
Linear regression:

$$y_i = \alpha + \beta x_i + \epsilon$$

Probabilistic reformulation:

$$y_i \sim N(\alpha + \beta x_i, \sigma^2)$$

$$\epsilon \sim N(0, \sigma^2)$$



# Bayesian Regression

Jupyter Demo Simulation

## Hierarchical Bayesian Regression

Group 1  $(x_{1,1}, y_{1,1}), (x_{2,1}, y_{2,1}), \dots, (x_{i,1}, y_{i,1}), \dots, (x_{n_1,1}, y_{n_1,1})$

Group 2  $(x_{1,2}, y_{1,2}), (x_{2,2}, y_{2,2}), \dots, (x_{i,2}, y_{i,2}), \dots, (x_{n_2,2}, y_{n_2,2})$

⋮

Group  $j$   $(x_{1,j}, y_{1,j}), (x_{2,j}, y_{2,j}), \dots, (x_{i,j}, y_{i,j}), \dots, (x_{n_j,j}, y_{n_j,j})$

⋮

Group  $J$   $(x_{1,J}, y_{1,J}), (x_{2,J}, y_{2,J}), \dots, (x_{i,J}, y_{i,J}), \dots, (x_{n_J,J}, y_{n_J,J})$

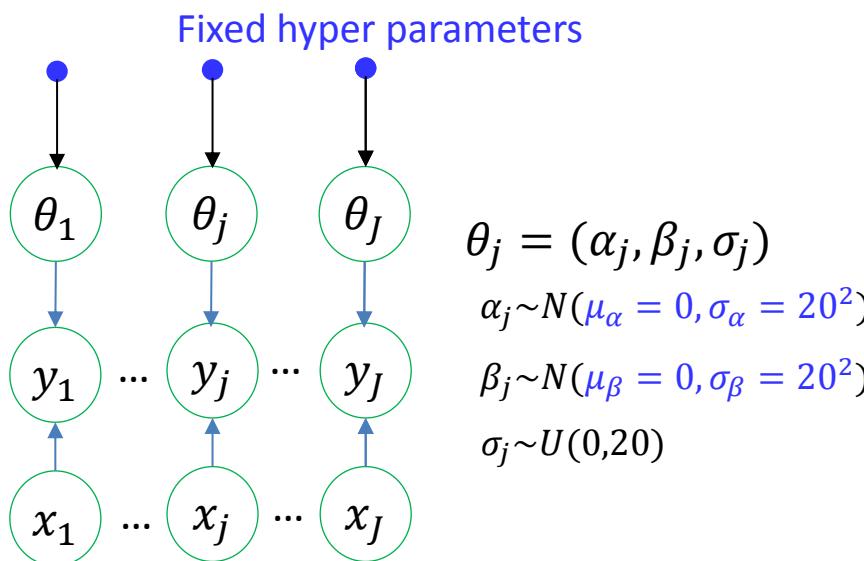
# Hierarchical Bayesian Regression

**Linear regression:**

$$y_{i,j} = \alpha_j + \beta_j x_{i,j} + \epsilon$$

**Probabilistic reformulation:**

$$\begin{aligned} y_{i,j} &\sim N(\alpha_j + \beta_j x_{i,j}, \sigma_j^2) \\ \epsilon &\sim N(0, \sigma_j^2) \end{aligned}$$



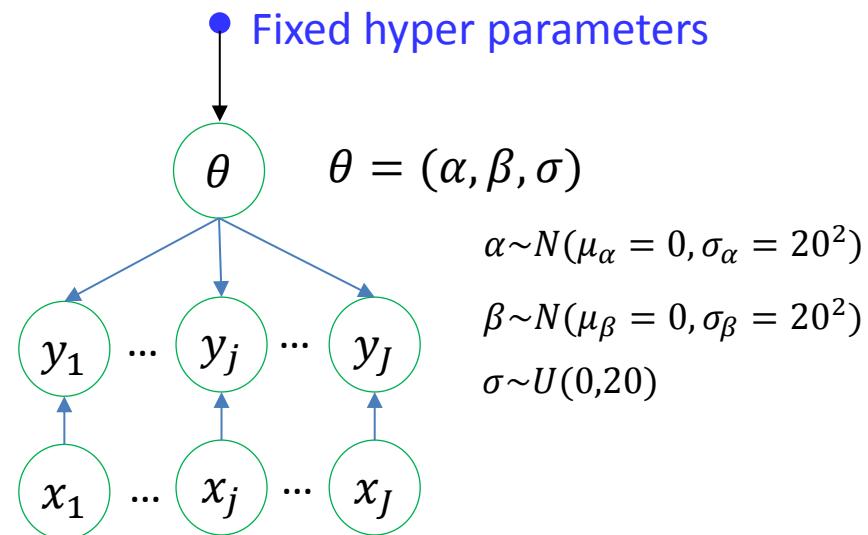
$$\begin{aligned} y_j &= (y_{1,j}, y_{2,j}, \dots, y_{n_j,j}) \\ x_j &= (x_{1,j}, x_{2,j}, \dots, x_{n_j,j}) \end{aligned}$$

**Linear regression:**

$$y_{i,j} = \alpha + \beta x_{i,j} + \epsilon$$

**Probabilistic reformulation:**

$$\begin{aligned} y_{i,j} &\sim N(\alpha + \beta x_{i,j}, \sigma^2) \\ \epsilon &\sim N(0, \sigma^2) \end{aligned}$$



$$\begin{aligned} y_j &= (y_{1,j}, y_{2,j}, \dots, y_{n_j,j}) \\ x_j &= (x_{1,j}, x_{2,j}, \dots, x_{n_j,j}) \end{aligned}$$

# Hierarchical Bayesian Regression

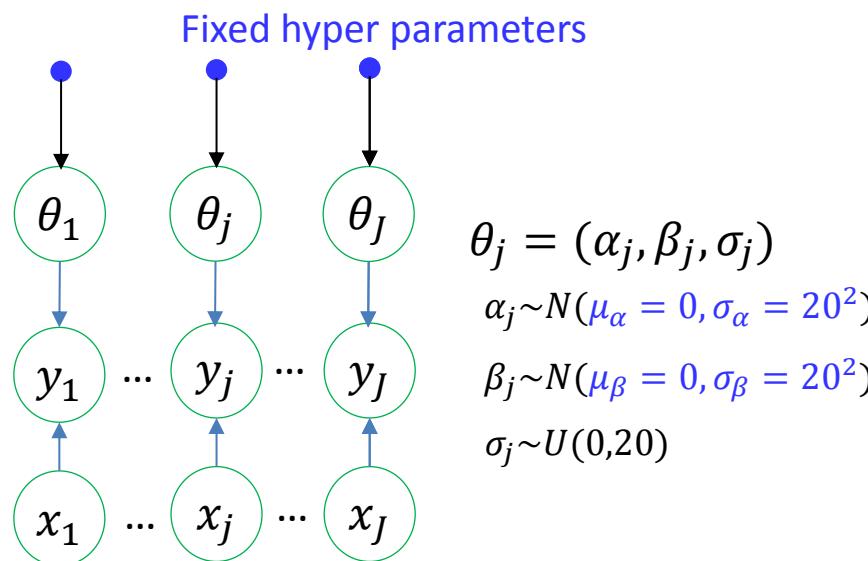
**Linear regression:**

$$y_{i,j} = \alpha_j + \beta_j x_{i,j} + \epsilon$$

**Probabilistic reformulation:**

$$y_{i,j} \sim N(\alpha_j + \beta_j x_{i,j}, \sigma_j^2)$$

$$\epsilon \sim N(0, \sigma_j^2)$$



$$y_j = (y_{1,j}, y_{2,j}, \dots, y_{n_j,j})$$

$$x_j = (x_{1,j}, x_{2,j}, \dots, x_{n_j,j})$$

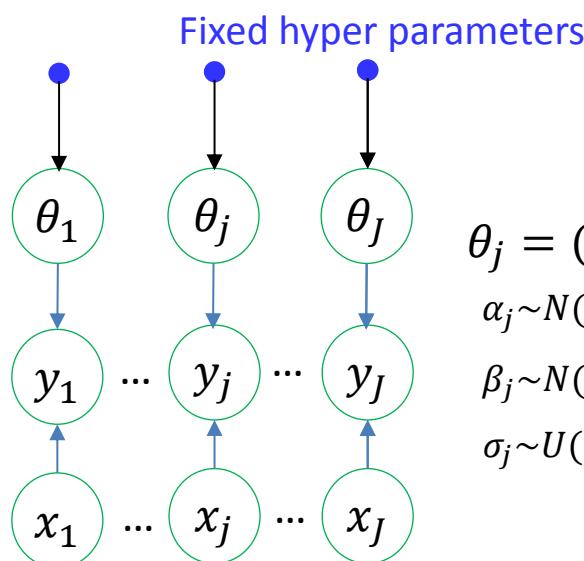
# Hierarchical Bayesian Regression

**Linear regression:**

$$y_{i,j} = \alpha_j + \beta_j x_{i,j} + \epsilon$$

**Probabilistic reformulation:**

$$\begin{aligned} y_{i,j} &\sim N(\alpha_j + \beta_j x_{i,j}, \sigma_j^2) \\ \epsilon &\sim N(0, \sigma_j^2) \end{aligned}$$



Fixed hyper parameters

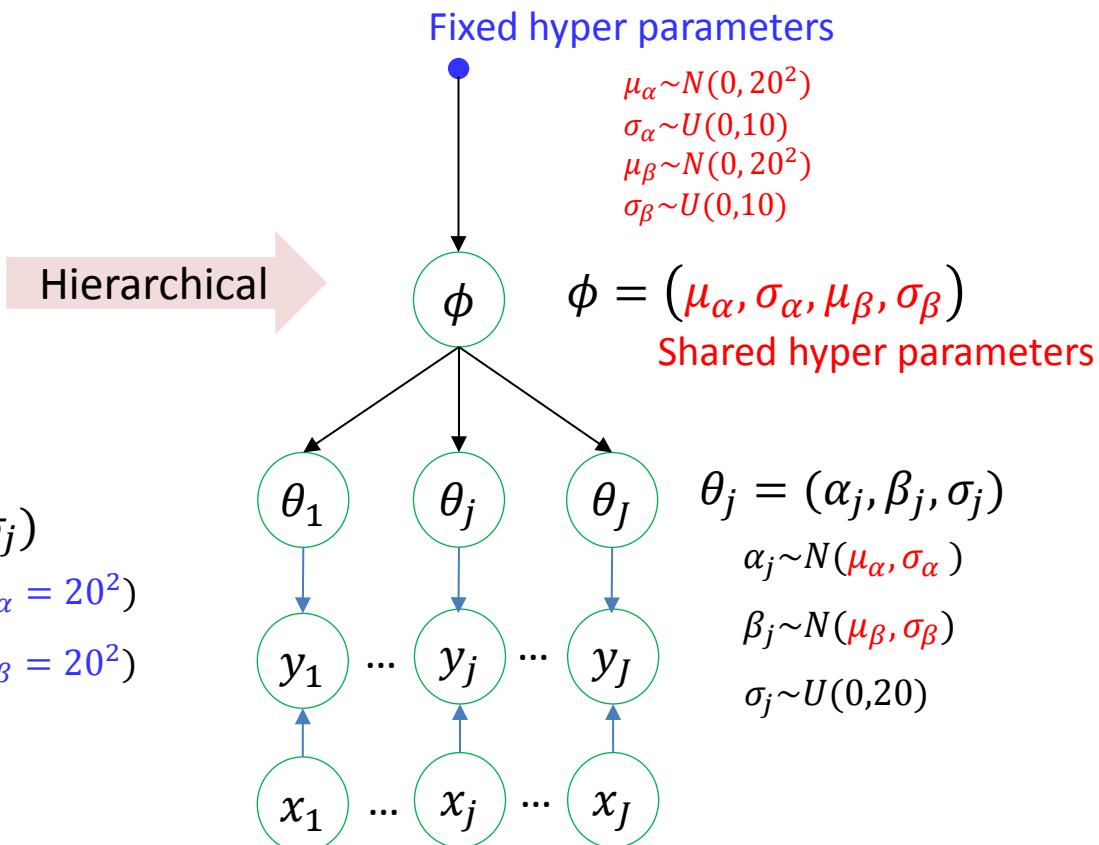
$$\theta_j = (\alpha_j, \beta_j, \sigma_j)$$

$$\alpha_j \sim N(\mu_\alpha = 0, \sigma_\alpha = 20^2)$$

$$\beta_j \sim N(\mu_\beta = 0, \sigma_\beta = 20^2)$$

$$\sigma_j \sim U(0, 20)$$

$$\begin{aligned} y_j &= (y_{1,j}, y_{2,j}, \dots, y_{n_j,j}) \\ x_j &= (x_{1,j}, x_{2,j}, \dots, x_{n_j,j}) \end{aligned}$$



Fixed hyper parameters

$$\phi = (\mu_\alpha, \sigma_\alpha, \mu_\beta, \sigma_\beta)$$

Shared hyper parameters

$$\theta_j = (\alpha_j, \beta_j, \sigma_j)$$

$$\alpha_j \sim N(\mu_\alpha, \sigma_\alpha)$$

$$\beta_j \sim N(\mu_\beta, \sigma_\beta)$$

$$\sigma_j \sim U(0, 20)$$

$$y_j = (y_{1,j}, y_{2,j}, \dots, y_{n_j,j})$$

$$x_j = (x_{1,j}, x_{2,j}, \dots, x_{n_j,j})$$

## Hierarchical Bayesian Regression

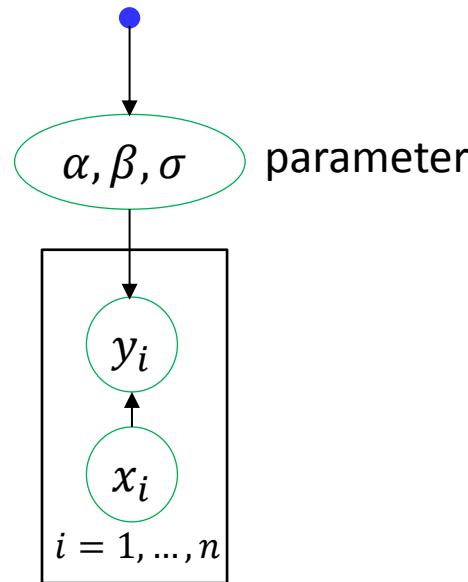
### Bayesian Regression

$$y_{i,j} = \alpha + \beta x_{i,j} + \epsilon$$

$$y_{i,j} \sim N(\alpha + \beta x_{i,j}, \sigma^2)$$

$$\epsilon \sim N(0, \sigma^2)$$

Fixed hyper parameters



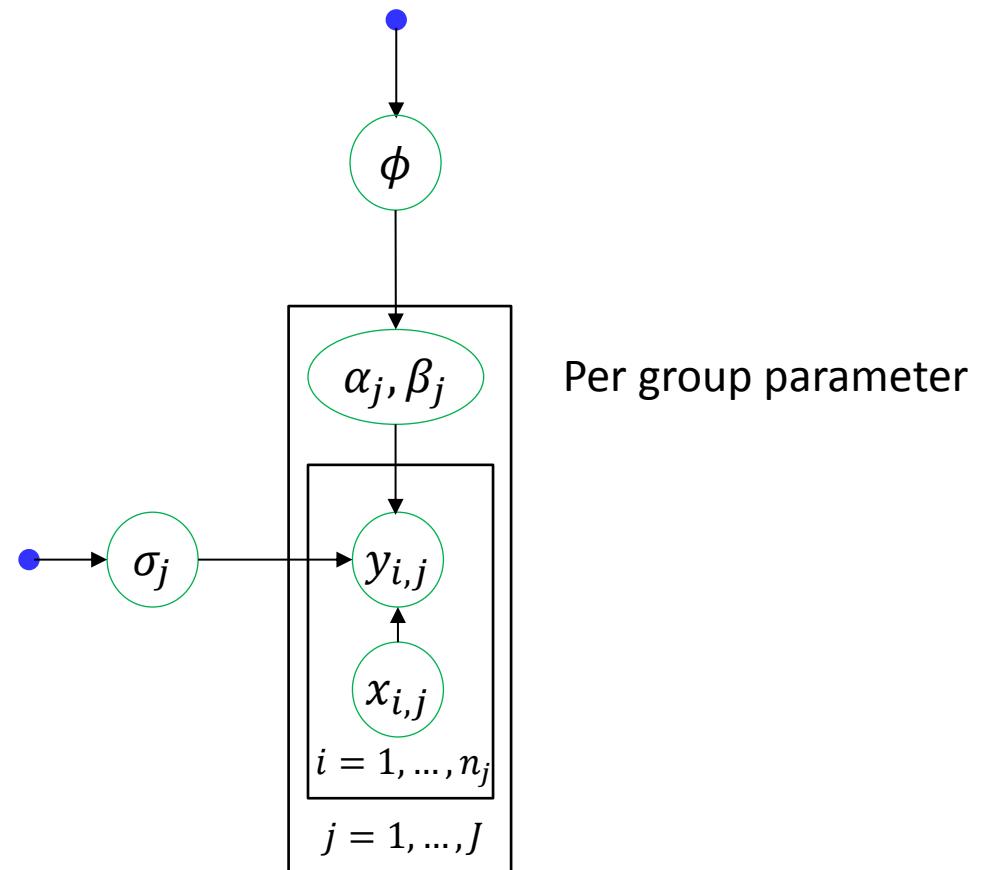
### Hierarchical Bayesian Regression

$$y_{i,j} = \alpha_j + \beta_j x_{i,j} + \epsilon$$

$$y_{i,j} \sim N(\alpha_j + \beta_j x_{i,j}, \sigma_j^2)$$

$$\epsilon \sim N(0, \sigma_j^2)$$

Fixed hyper parameters



## Hierarchical Bayesian Regression

Jupyter Demo Simulation

<http://pymc-devs.github.io/pymc3/notebooks/GLM-hierarchical.html>

## Bayesian decision theory

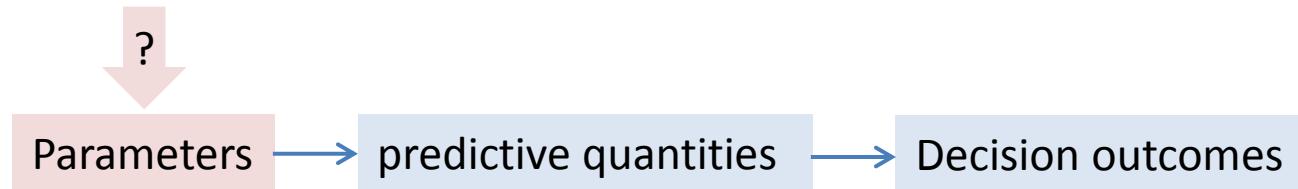
- ✓ Model has been built
- ✓ The inferences are conducted
- ✓ Model has been checked

What else?

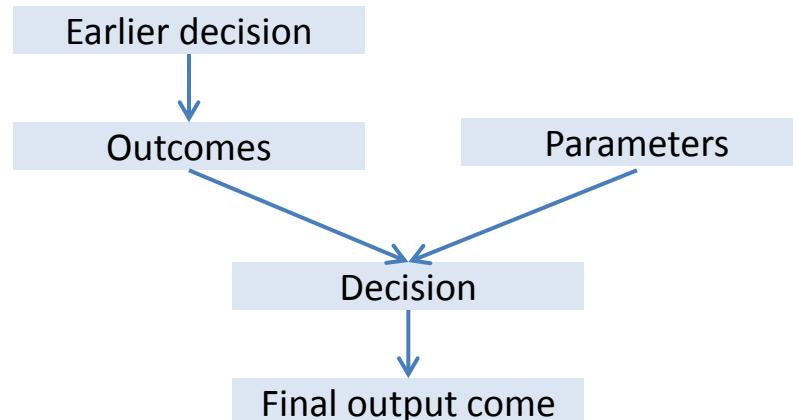
- Two ways of using *Bayesian inference in decision making*

### (1) Measure uncertainties regarding predictive quantiles

Bayesian inference



### (2) Within decision analysis (Multi-stages decision problems)



## Elements of Bayesian decision analysis

Decision analysis is inherently more complicated than statistical inference because it involves *optimization over decision* as well *averaging over uncertainties*

### Elements of Bayesian decision analysis

1. Enumerate the space of all possible decisions (actions)  $a$  and outcome  $o$ 
  - The vector of outcomes  $o$  can include observables (predicted values  $\tilde{y}$ ) and parameters  $\theta$
2. Determine  $p(o|a)$ , a *conditional posterior probability distribution* of outcomes  $o$  for each decision option  $a$ 
  - Outcome  $o$  is random variable, while decision  $a$  is deterministic (set by user)
3. Define a utility function  $U(o)$  mapping outcomes  $o$  onto the real numbers
  - If multiple attributes (vector)  $o$  is considered, the utility function must trade off different goods
4. Compute the  $E = \text{expected utility } E(U(o)|a)$  as a function of the decision  $a$ , and choose the decision with highest expected utility

## Examples that will discussed in the lecture

1. Survey incentives
  - ✓ Conduct only step 1 and 2 to estimate the expected effect depending on the option that can be taken
2. Medical test decision
  - ✓ Conduct only step 1 and 2 in a sequential decision making and computed value of information
3. Risk analysis of Radon exposure and making prevention strategies
  - ✓ Conduct the full Bayesian analysis

## Example 1: Survey incentives

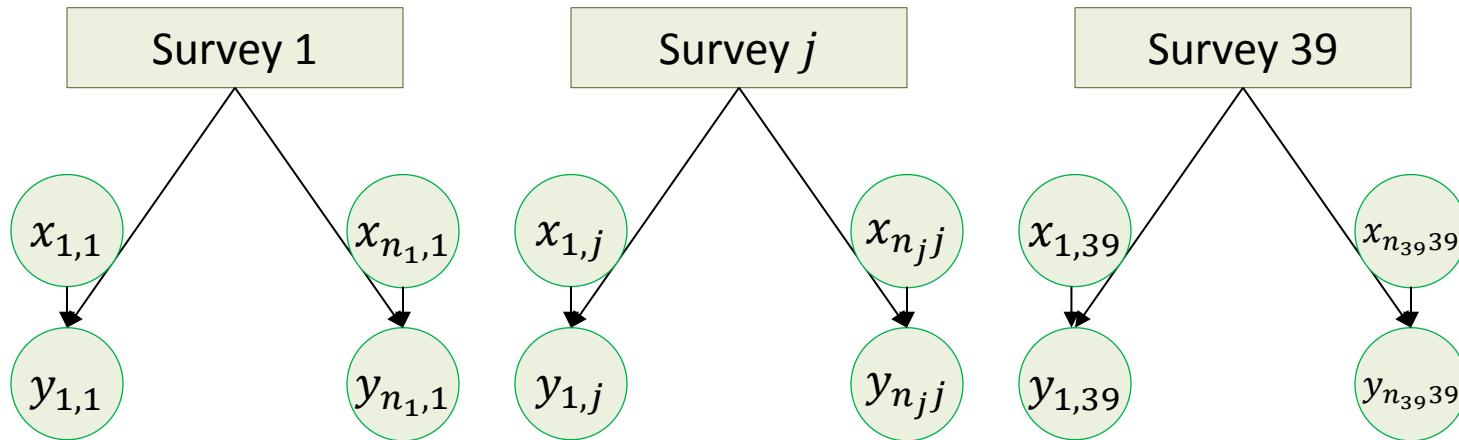


- Do the benefits of incentives outweigh the cost?
- If an incentive is given,
  - ✓ how and when should it be offered
  - ✓ whom should it be offered to
  - ✓ what form should it take
  - ✓ how large should its value be?

## Example 1: Survey incentives

### Data descriptions:

The New York City Social Indicators Survey, a telephone study conducted every two years that has had a response rate below 50%



- In total 39 surveys including 101 experiment conditions are conducted:
- $y_i$  is the observed response rate for observation  $i = 1, \dots, 101$
- All experiments have different conditions  $x_i$  on:
  - The **value of the incentive** (in tens of 1999 dollars)
  - The **timing of the incentive payment** (given before the survey or after)
  - The **form of the incentive** (cash or gift)
  - The **mode of the survey** (face-to-face or telephone)
  - The **burden or effort**, required to survey respondents (high burden or low burden)
- Use the differences,  $z_i = y_i - y_i^0$ ,  
where  $y_i^0$  corresponds to the lowest valued incentive condition

## Example 1: Survey incentives

### A simple linear regression approach

Response rate :  $y_i = X_i\beta + \beta_0$

$$y_i = \frac{n_i}{N_i} = \frac{\text{respondents}}{\text{Trial}}$$

Predictor variables :  $X_i = (X_{1,i}, X_{2,i}, X_{3,i}, X_{4,i}, X_{5,i})$

$X_{1,i}$  : The value of the incentive (in tens of 1999 dollars)

$X_{2,i}$  : The timing of the incentive payment (given before the survey or after)

$X_{3,i}$  : The form of the incentive (cash or gift)

$X_{4,i}$  : The mode of the survey (face-to-face or telephone)

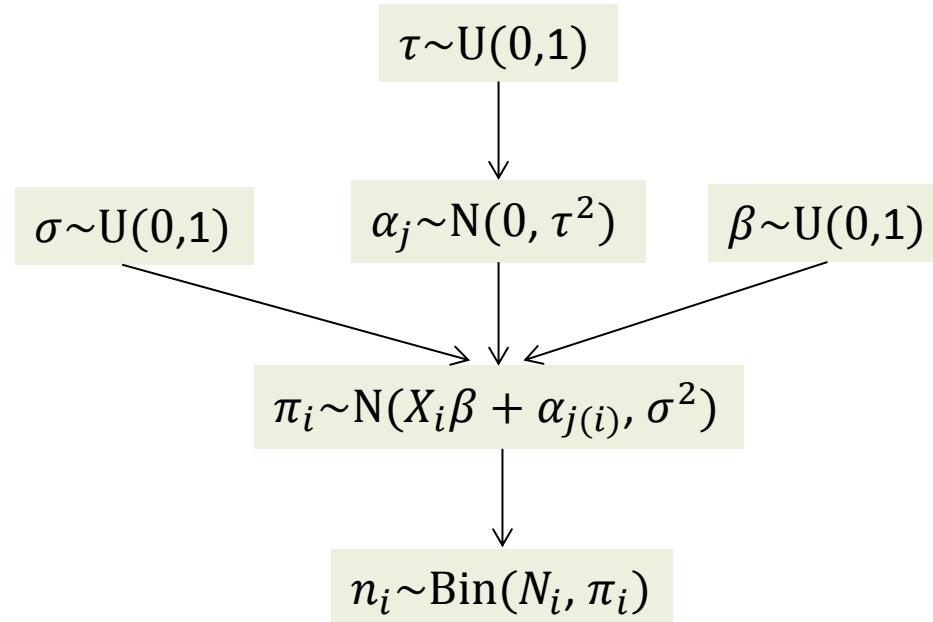
$X_{5,i}$  : The burden or effort, required to survey respondents (high burden or low burden)

**Limitations** in employing a classical regression model relating  $y_i$  to the predictor variables:

- Unable to model interactions
- Unable to reflect the hierarchical structure of the data
- Difficulty in dealing with unequal sample size

## Example 1: Survey incentives

- **Step1** : perform a meta-analysis to estimate the effects of incentives on response rate, as a function of the amount of the incentive and the way it is implemented



$\alpha_{j(i)}$  : random effect for the survey  $j = 1, \dots, 39$   
**(address the hierarchical structure)**

$X_i\beta$  : linear predictor for conditioned on data point  $i$

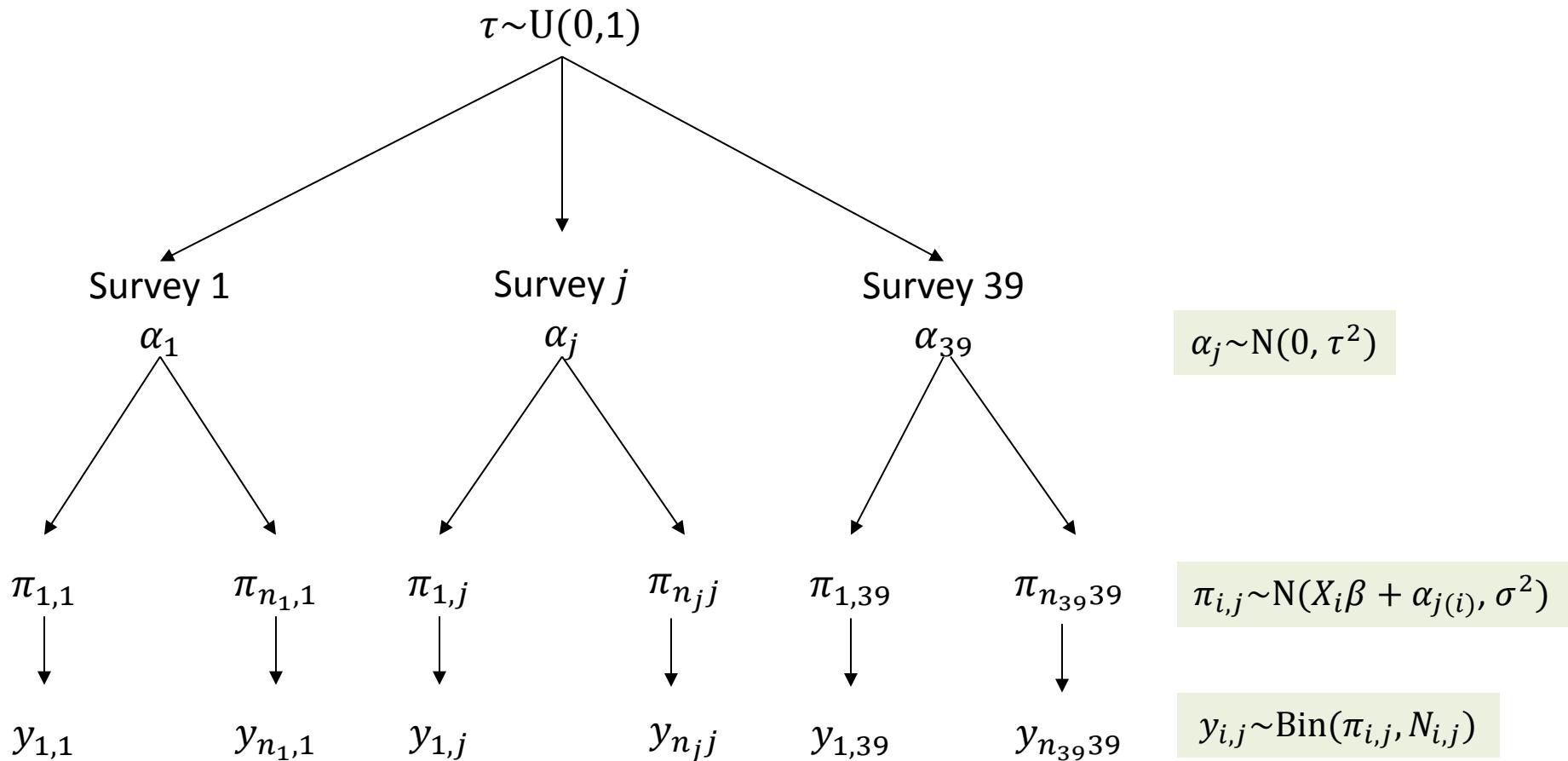
$\pi_i$  : population response probability  $i = 1, \dots, 101$

$N_i$  : number of persons contacted

$n_i$  : number of respondents ( $y_i = n_i/N_i$ )

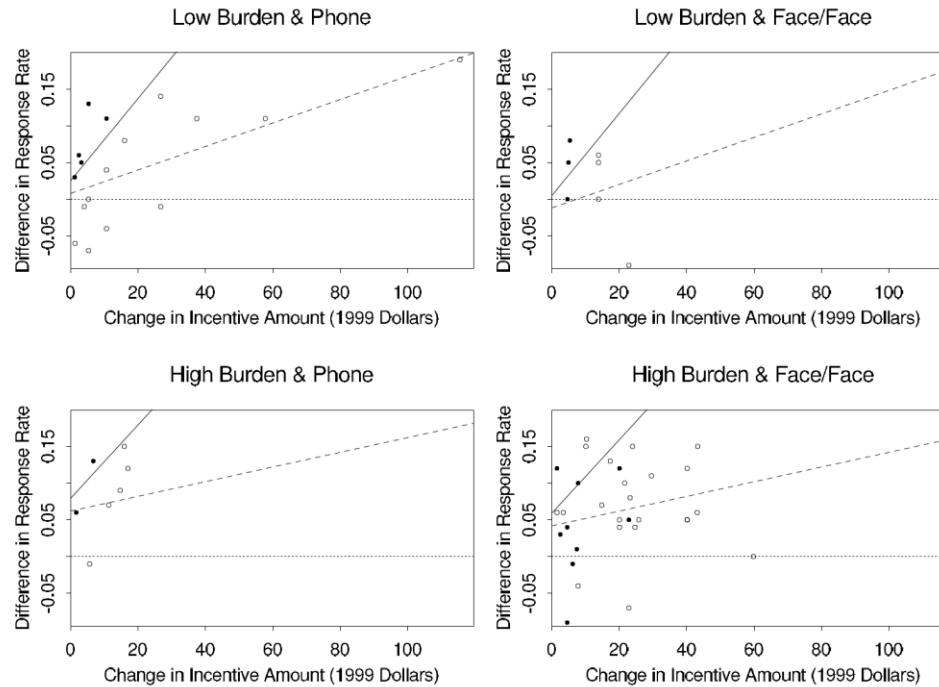
## Example 1: Survey incentives

- Step1** : perform a meta-analysis to estimate the effects of incentives on response rate, as a function of the amount of the incentive and the way it is implemented



## Example 1: Survey incentives

- Step2 : use the inference to estimate the costs and benefits of incentives



$$y_i = X_i \hat{\beta} + \beta_0$$

● Prepaid

○ After paid

Some of the findings are :

- an extra \$10 in incentive is expected to increase the response rate by 3–4 percentage points
- cash incentives increase the response rate by about 1 percentage point relative to noncash
- prepaid incentives increase the response rate by 1–2 percentage points relative to postpaid
- incentives have a bigger impact (by about 5 percentage points) on high-burden surveys compared to low-burden surveys.

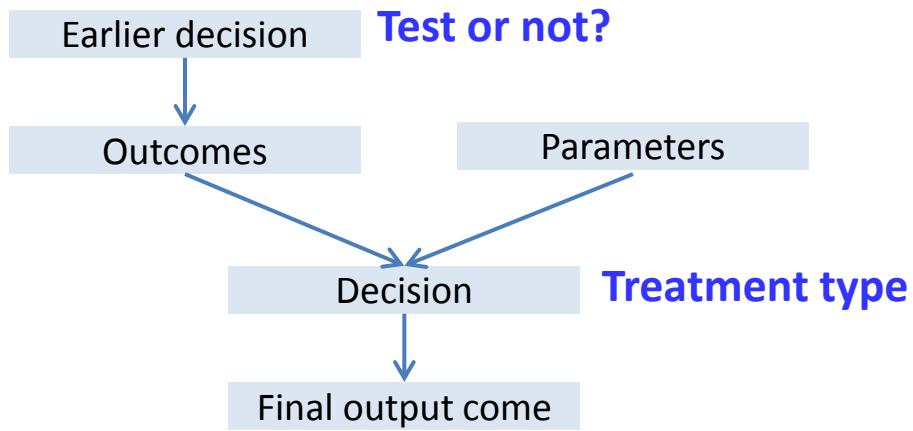
These inferences can be used to design a new survey (cost vs. response rate)

## Example 2: Multistage decision making for medical screening

95-year-old man with an apparently malignant tumor in the lung



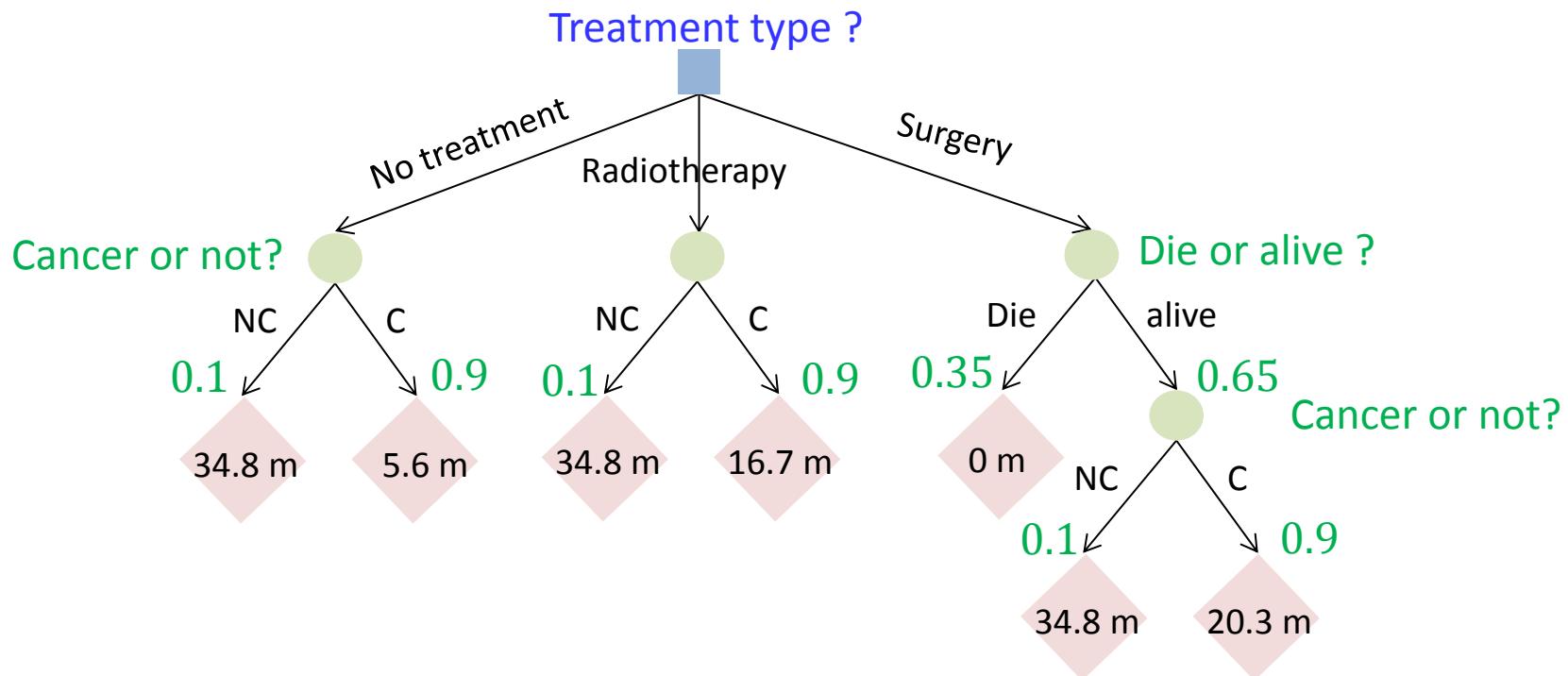
<Multi-stages decision problems>



Bayesian inference is particularly useful in updating the state of knowledge with the information gained at each step.

## Example 2: Multistage decision making for medical screening

### Single decision point (without test)

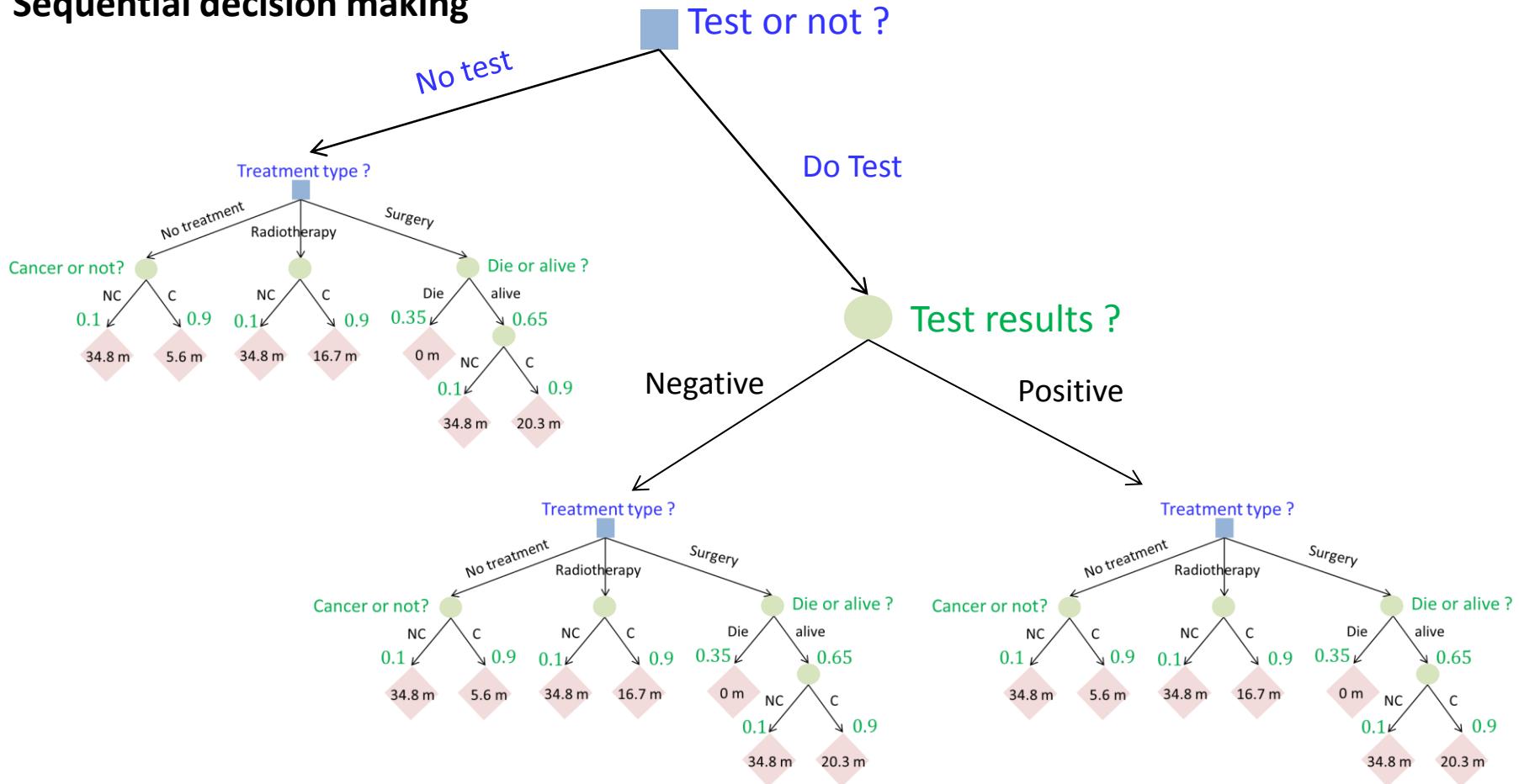


Quality-adjusted life expectancy under each treatment:

- With no treatment :  $0.1 \times 34.8 + 0.9 \times 5.6 = 8.5 \text{ m}$
- With radiotherapy :  $0.1 \times 34.8 + 0.9 \times 16.7 - 1 = 17.5 \text{ m}$  ← Best option
- With surgery :  $0.35 \times 0 + 0.65 \times (0.1 \times 34.8 + 0.9 \times 20.3 - 1) = 13.5 \text{ m}$

## Example 2: Multistage decision making for medical screening

### Sequential decision making



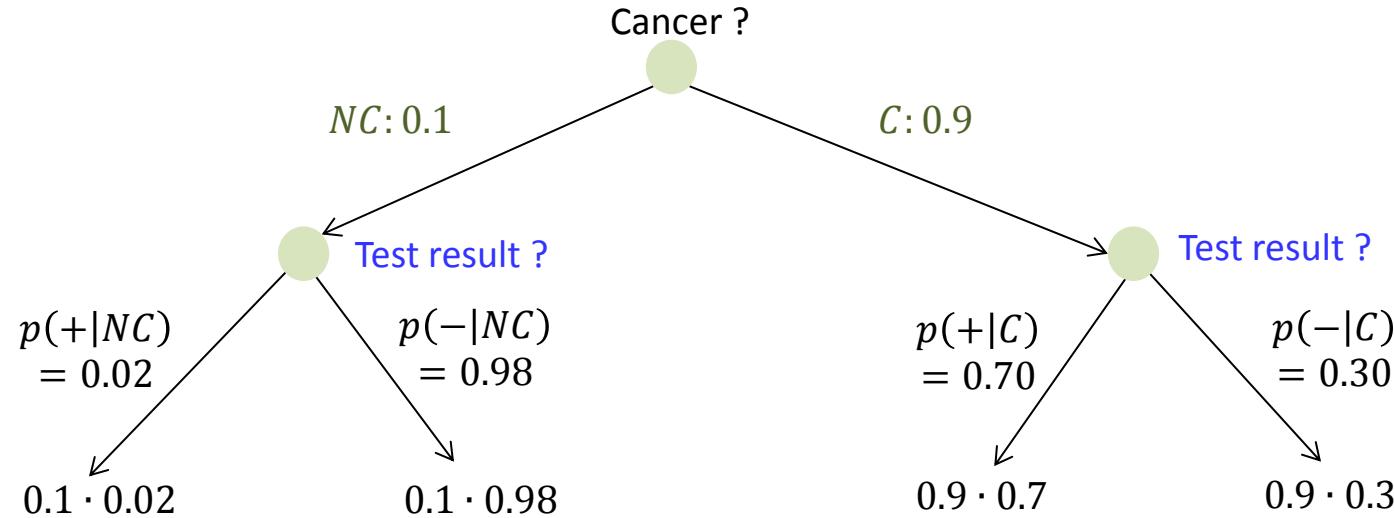
- Based on the test result, the probability of cancer can be updated using Bayes rule:

$$Pr(C|T) = \frac{Pr(C) p(T|C)}{Pr(C) p(T|C) + Pr(NC) p(T|NC)}$$

- Updated information can be used for sequential decision making

## Example 2: Multistage decision making for medical screening

Update information based on the test results



$$\begin{aligned} \Pr(T = +) &= \Pr(NC) p(T = +|NC) + \Pr(C) p(T = +|C) \\ &= 0.1 \cdot 0.02 + 0.9 \cdot 0.7 = 0.632 \end{aligned}$$

$$\begin{aligned} \Pr(T = -) &= \Pr(NC) p(T = -|NC) + \Pr(C) p(T = -|C) \\ &= 0.1 \cdot 0.98 + 0.9 \cdot 0.3 = 0.368 \end{aligned}$$

$$\Pr(C|T) = \frac{\Pr(C) p(T|C)}{\Pr(T)}$$

$$\Pr(C|T = +) = \frac{0.9 \cdot 0.7}{0.1 \cdot 0.02 + 0.9 \cdot 0.7} = 0.997$$

$$\Pr(C|T = -) = \frac{0.9 \cdot 0.3}{0.1 \cdot 0.98 + 0.9 \cdot 0.3} = 0.734$$

$$\Pr(NC|T) = \frac{\Pr(NC) p(T|NC)}{\Pr(T)}$$

$$\Pr(NC|T = +) = \frac{0.1 \cdot 0.7}{0.1 \cdot 0.02 + 0.9 \cdot 0.7} = 0.003$$

$$\Pr(NC|T = -) = \frac{0.1 \cdot 0.3}{0.1 \cdot 0.98 + 0.9 \cdot 0.3} = 0.266$$

## Example 2: Multistage decision making for medical screening

Choose the best action at the second stage using the updated information

$$Pr(cancer|T = +) = \frac{0.9 \cdot 0.7}{0.9 \cdot 0.7 + 0.1 \cdot 0.02} = 0.997$$

Quality-adjusted life expectancy under each treatment:

- With no treatment :  $0.997 \cdot 5.6 + 0.003 \cdot 34.8 = 5.7$  months
- With radiotherapy :  $0.997 \cdot 16.7 + 0.003 \cdot 34.8 - 1 = 15.8$  months
- With surgery :  $0.35 \cdot 0.65(0.997 \cdot 20.3 + 0.003 \cdot 34.8 - 1) = 12.6$  months

Best action

$$Pr(cancer|T = -) = \frac{0.9 \cdot 0.3}{0.9 \cdot 0.3 + 0.1 \cdot 0.98} = 0.734$$

Quality-adjusted life expectancy under each treatment:

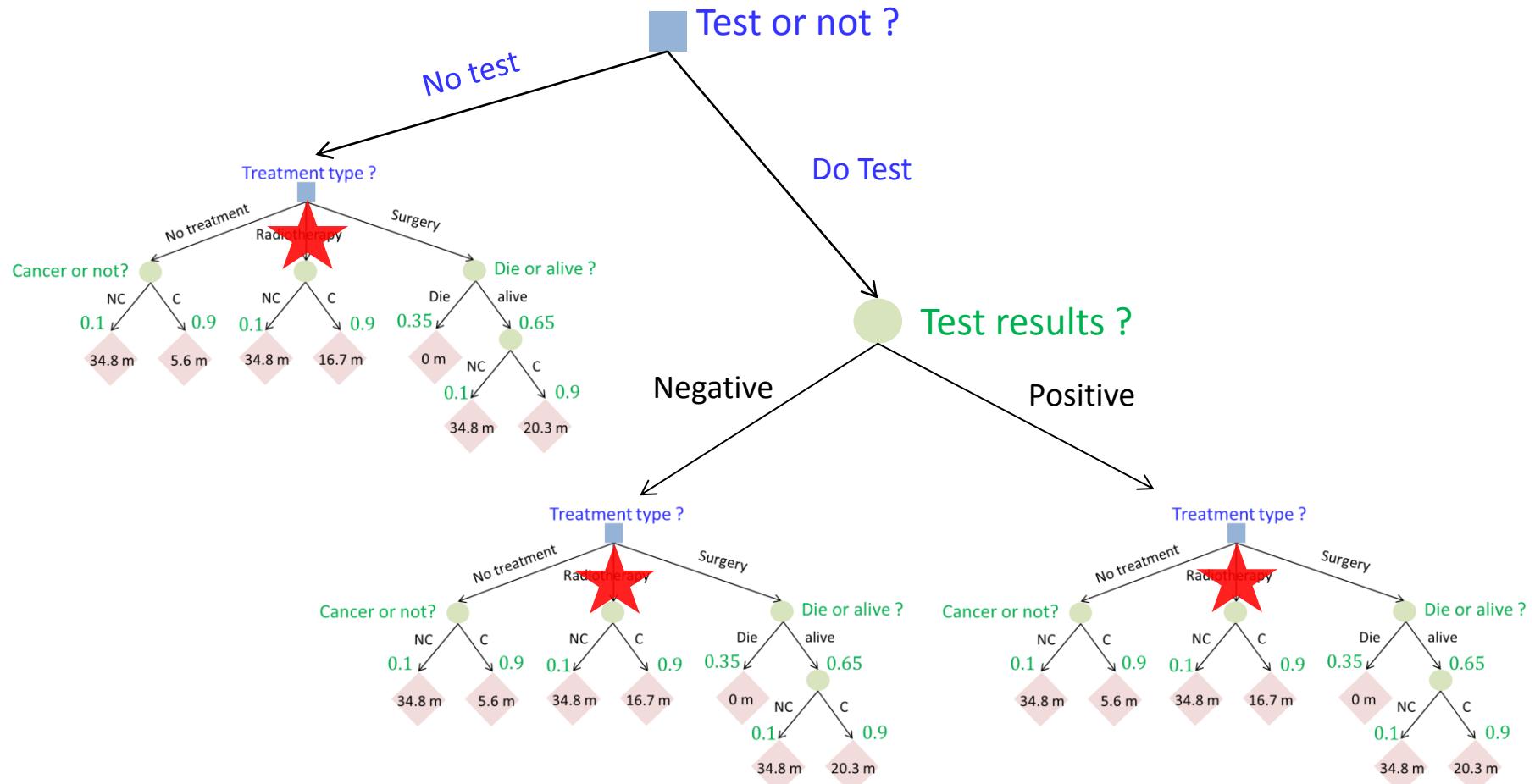
- With no treatment :  $0.734 \cdot 5.6 + 0.266 \cdot 34.8 = 13.4$  months
- With radiotherapy :  $0.734 \cdot 16.7 + 0.266 \cdot 34.8 - 1 = 20.5$  months
- With surgery :  $0.35 \cdot 0.65(0.734 \cdot 20.3 + 0.266 \cdot 34.8 - 1) = 15.1$  months

Best action

It is clear conducting a test is not a good idea since no change in the selected option!

## Example 2: Multistage decision making for medical screening

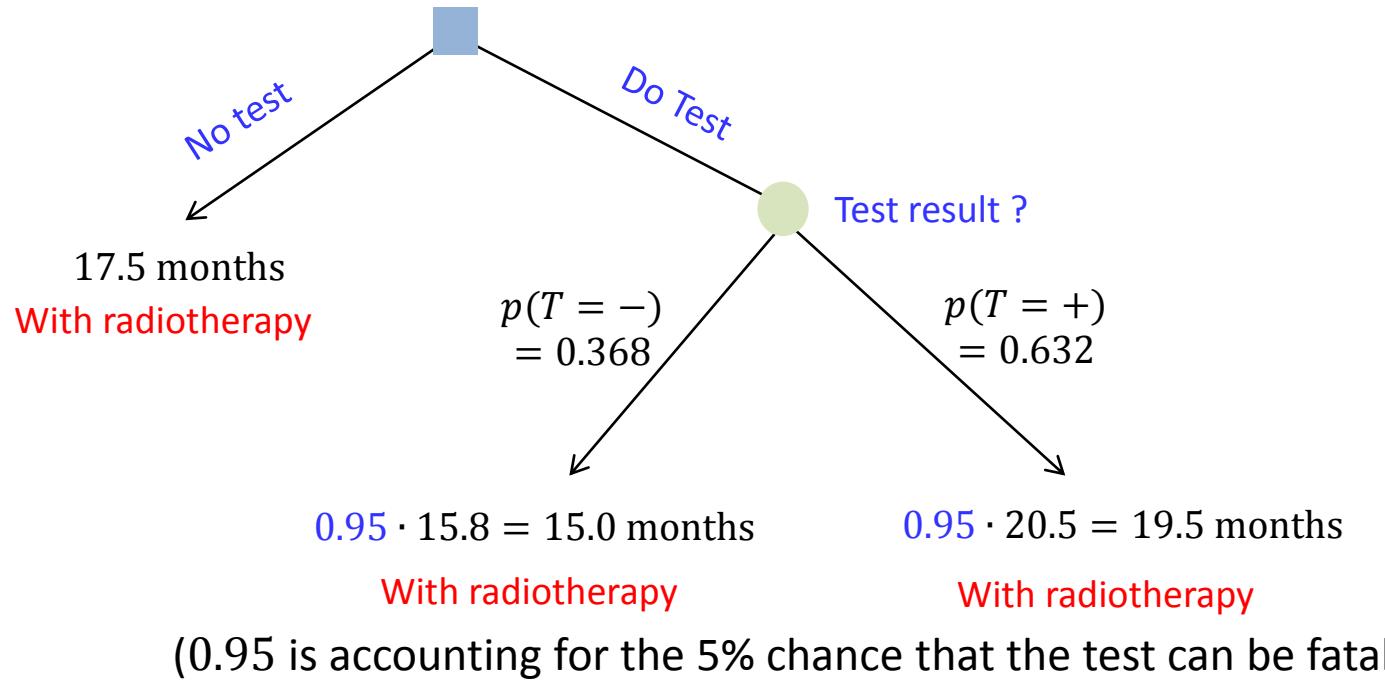
### Sequential decision making



When selecting a decision at the first state, we should assume that all the decisions at the subsequent stages will be made optimum!! (key idea of sequential decision making: Control, MDP)

## Example 2: Multistage decision making for medical screening

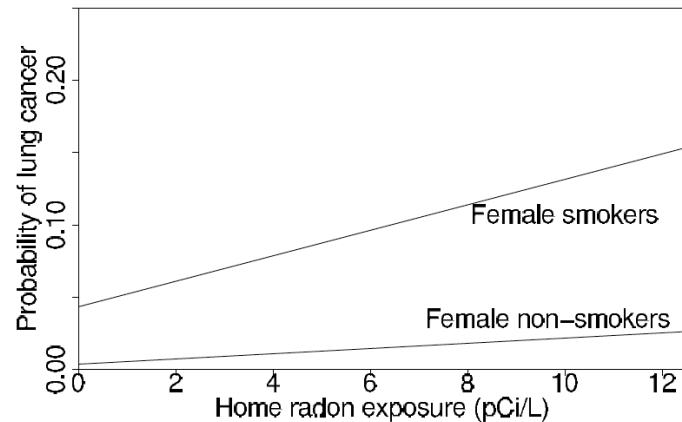
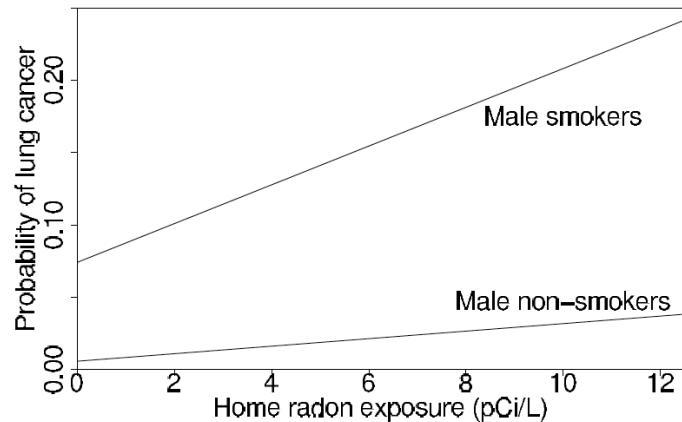
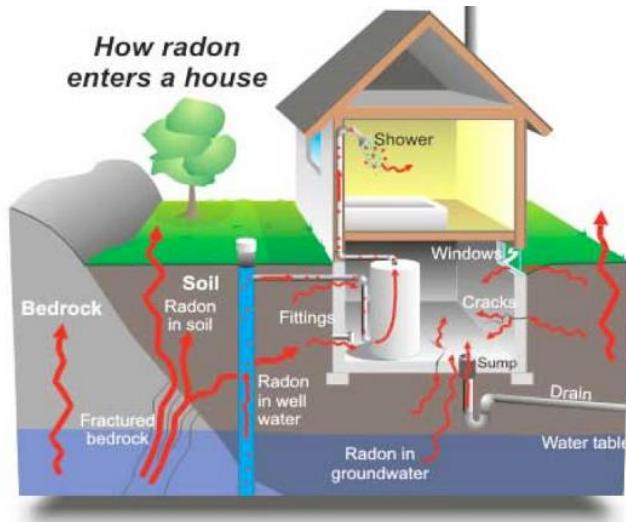
### Decision analysis for performing test



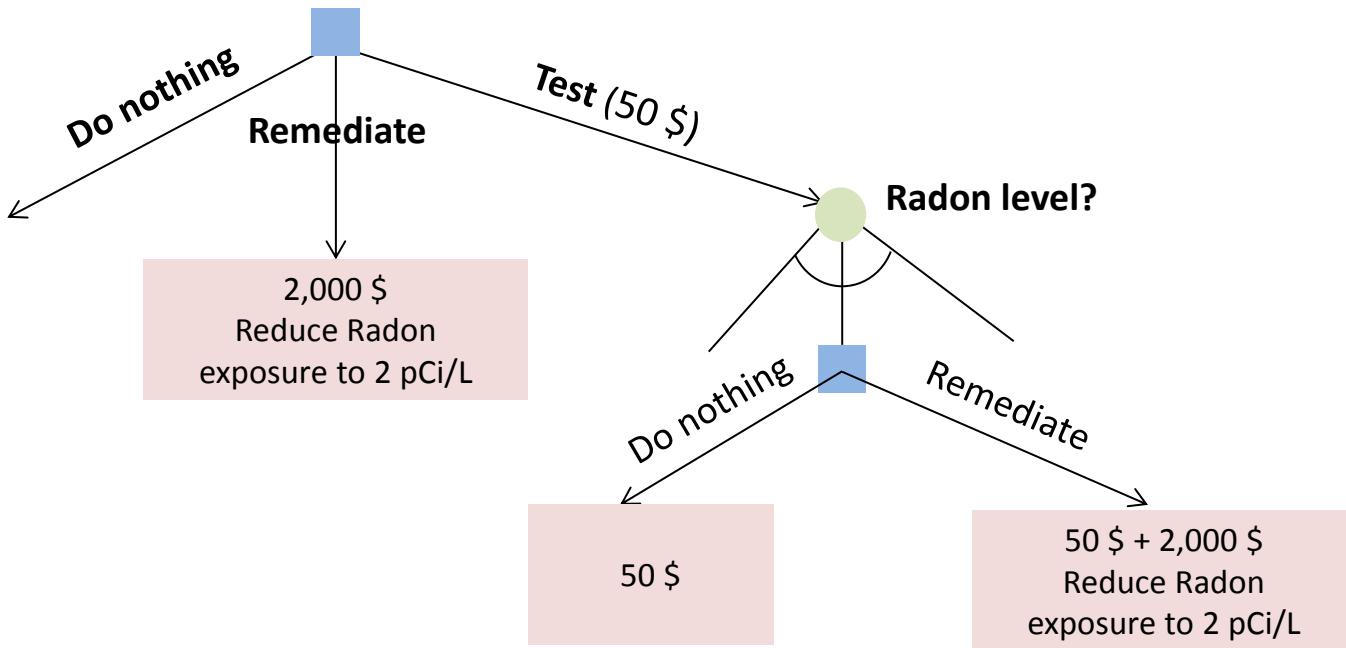
- No test : 17.5 months is the best
- Test :  $0.368 \cdot 19.5 + 0.632 \cdot 15.0 = 16.6 \text{ months}$

No change in the selected option and no improvement in the expected cost  
→ Not conducting the test is the best option

## Example 3: Hierarchical decision analysis for home radon



## Example 3: Hierarchical decision analysis for home radon



**Two tasks:**

- Given prior information about Radon concentration, decide whether to conduct a measurement test (first decision)
- Given the measurement of Radon concentration (we have a posterior of Radon con.), decide whether to remediate (second decision)

**Performing the decision analysis requires estimating for the risks, which is done by using hierarchical Bayesian model**

## Example 3: Hierarchical decision analysis for home radon

Define utility : trades off dollars and lives

- $D_d$ , the dollar value associated with a reduction of  $10^{-6}$  in probability of death for lung cancer (the money that you are willing to pay to reduce  $10^{-6}$  in probability of death for lung cancer)
- $D_r$ , the dollar value associated with a reduction of 1 pCi/L in home radon level for 30-year period
  - depends on the number of lives saved by a drop in the Radon level
  - affected by variety of factors including gender, smoking status...
  - $D_r = 4800D_d$  (typical U.S. household)
- $R_{action}$ , the home radon level above which you should remediate if your radon level is known
  - depends on the dollar value of radon reduction and the benefits of remediation

Action level  $R_{action}$  is determined as the value at which the benefit of remediation

$$\$D_r(R_{action} - R_{remed}) = \text{remediation cost of } \$2,000$$

$$R_{action} = \frac{\$2,000}{D_r} + R_{remedy}$$

$R_{action} = 4 \text{ pCi/L}$  is used as exemplary value, which can vary depending on financial resources, general risk tolerance, attitude toward risk, smoking status, etc.

## Example 3: Hierarchical decision analysis for home radon

### Bayesian inference for country radon level

Two data sets:

- Long-term measurements from approximately 5,000 houses, selected as a cluster sample from 125 randomly selected countries
- Short-term measurements from about 80,000 houses, sampled at random from all the counties in the U.S.

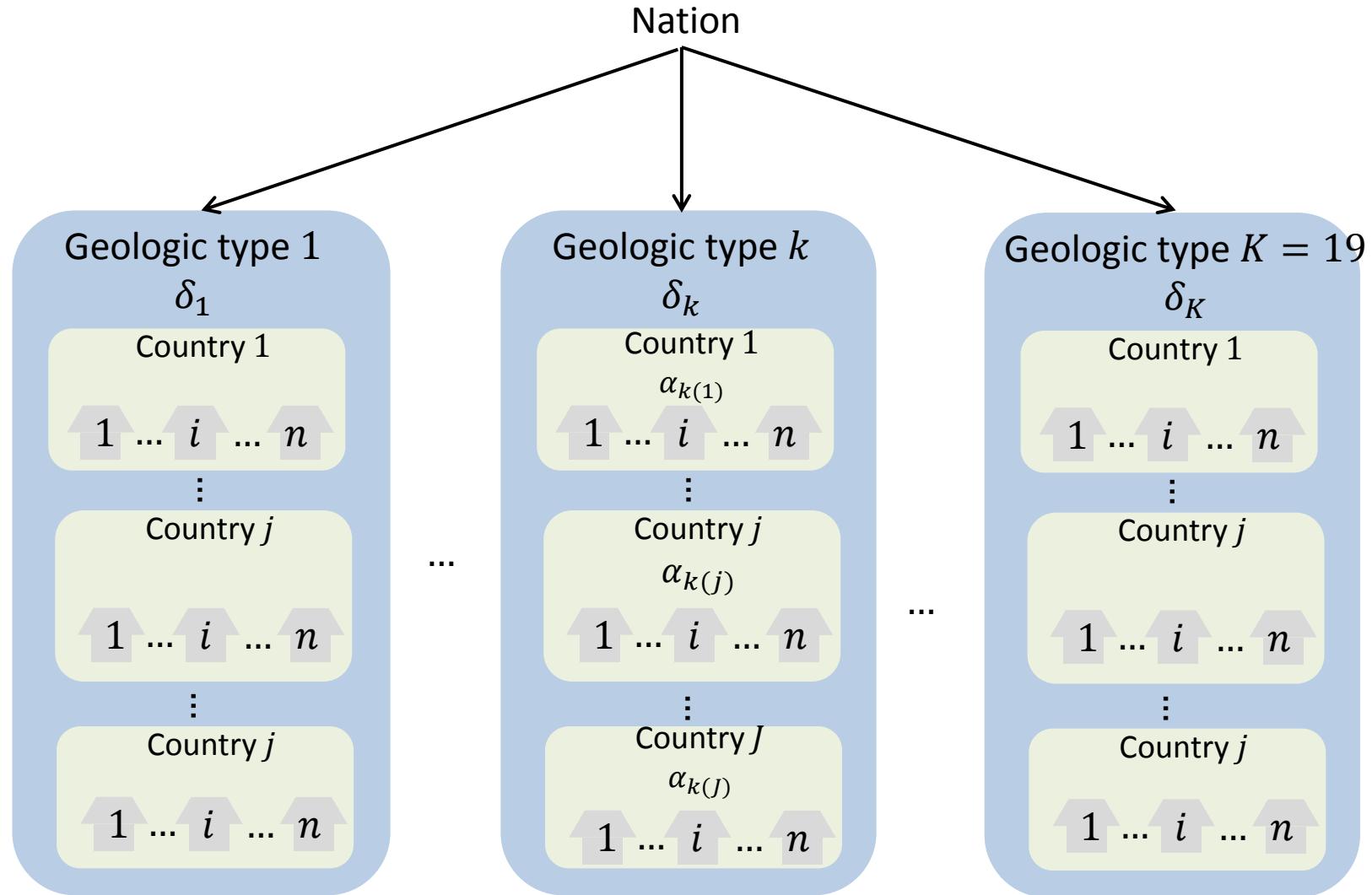
a relatively small amount of accurate data VS. a large amount of biased and imprecise data

Challenges:

Use the good data to calibrate the bad data, so that inference can be made about the entire country, not merely the 125 countries in the sample of long-term measurements

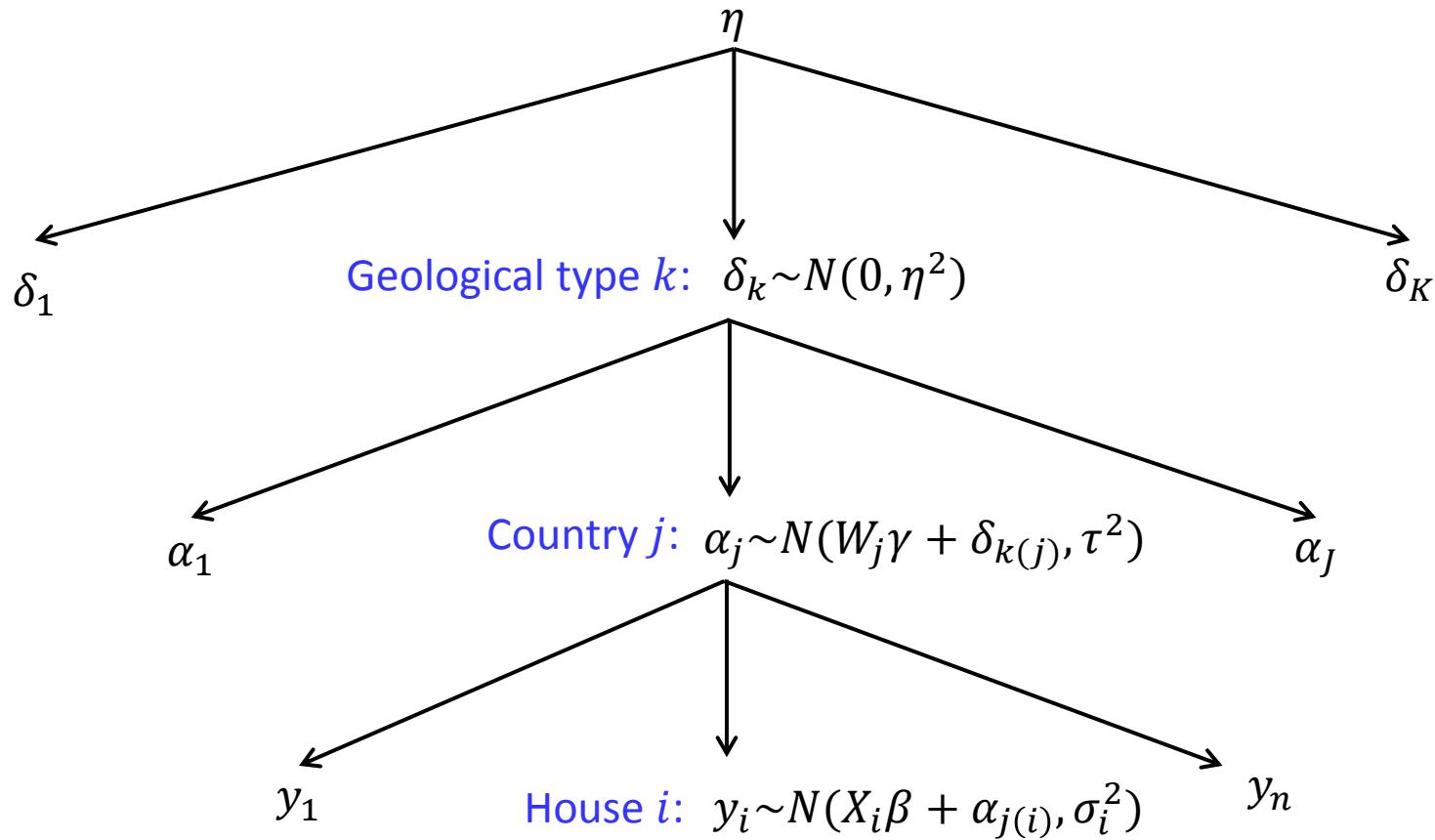
## Example 3: Hierarchical decision analysis for home radon

### Hierarchical modeling for individual house radon level



## Example 3: Hierarchical decision analysis for home radon

### Hierarchical modeling for individual house radon level



$W_j$ : county-level predictors including climate data and a measure of the uranium level in the soil

$X_i$ : household-level predictors including indicators for whether the house has a basement and

**measurement type ( $\sigma_i^2$  varies depending on measurement type (long-term or short term))**

## Example 3: Hierarchical decision analysis for home radon

### Bayesian inference for individual home level

$R_i$  = radon concentration in house  $i$

$$\theta_i = \log(R_i)$$

$$\theta_i = N(M_i, S_i^2)$$

- The mean  $M_i = X_i \hat{B} + \hat{\alpha}_{j(i)}$  is computed from the posterior simulations of the model estimation.  
 *$\hat{B}$  and  $\hat{\alpha}$  are the posterior means from the analysis in the appropriate region of the country.  
(computed from country level measurement data)*
- The variance  $S_i^2$  is computed from the posterior simulations of the model estimation taking into account the posterior uncertainty in the coefficients  $\alpha, \beta$  and also hierarchical variance components  $\eta^2$  and  $\tau^2$
- Serves as a **prior distribution** for the homeowner in that the distribution is constructed solely based on the basement information  $X_i$  and the county level parameter  $\hat{\alpha}_{j(i)}$

Using  $p(\theta_i)$  Decision whether to perform measurement test or not can be made

## Example 3: Hierarchical decision analysis for home radon

### Bayesian inference for individual home level

Likelihood :

$$Y_i \sim N(\theta, \sigma_Y^2) \rightarrow p(y_i | \theta, \sigma_Y^2) = \frac{1}{\sqrt{2\pi\sigma_Y^2}} \exp\left(-\frac{(y_i - \theta)^2}{2\sigma_Y^2}\right)$$

Prior:

$$\theta \sim N(\mu_0, \tau_0^2) \rightarrow p(\theta) = \frac{1}{\sqrt{2\pi\tau_0^2}} \exp\left(-\frac{(\theta - \mu_0)^2}{2\tau_0^2}\right)$$

Posterior on  $\theta = \mu_Y$

$$P(\theta | y) = N\left(\theta \middle| \frac{\frac{\mu_0}{\tau_0^2} + \frac{n\bar{y}}{\sigma_Y^2}}{\frac{1}{\tau_0^2} + \frac{n}{\sigma_Y^2}}, \left(\frac{1}{\tau_0^2} + \frac{n}{\sigma_Y^2}\right)^{-1}\right)$$

$R_i$  = radon concentration in house  $i$

$\theta_i = \log(R_i)$

Assume we measure Radon concentration  $y_i \sim N(\theta_i, \sigma_i^2)$

Prior (from previous slide) :

$$p(\theta_i) = N(M_i, S_i^2)$$

Likelihood

$$y_i \sim N(\theta_i, \sigma_i^2)$$

The posterior distribution :

$$\theta_i | M_i, y_i \sim N(\Lambda_i, V_i)$$

$$\Lambda_i = \frac{\frac{M_i}{S_i^2} + \frac{y_i}{\sigma_i^2}}{\frac{1}{S_i^2} + \frac{1}{\sigma_i^2}} \quad V_i = \frac{1}{\frac{1}{S_i^2} + \frac{1}{\sigma_i^2}}$$

## Example 3: Hierarchical decision analysis for home radon

### Decision analysis for individual homeowners

$$R_i = \text{radon concentration in house } i$$
$$\theta_i = \log(R_i)$$

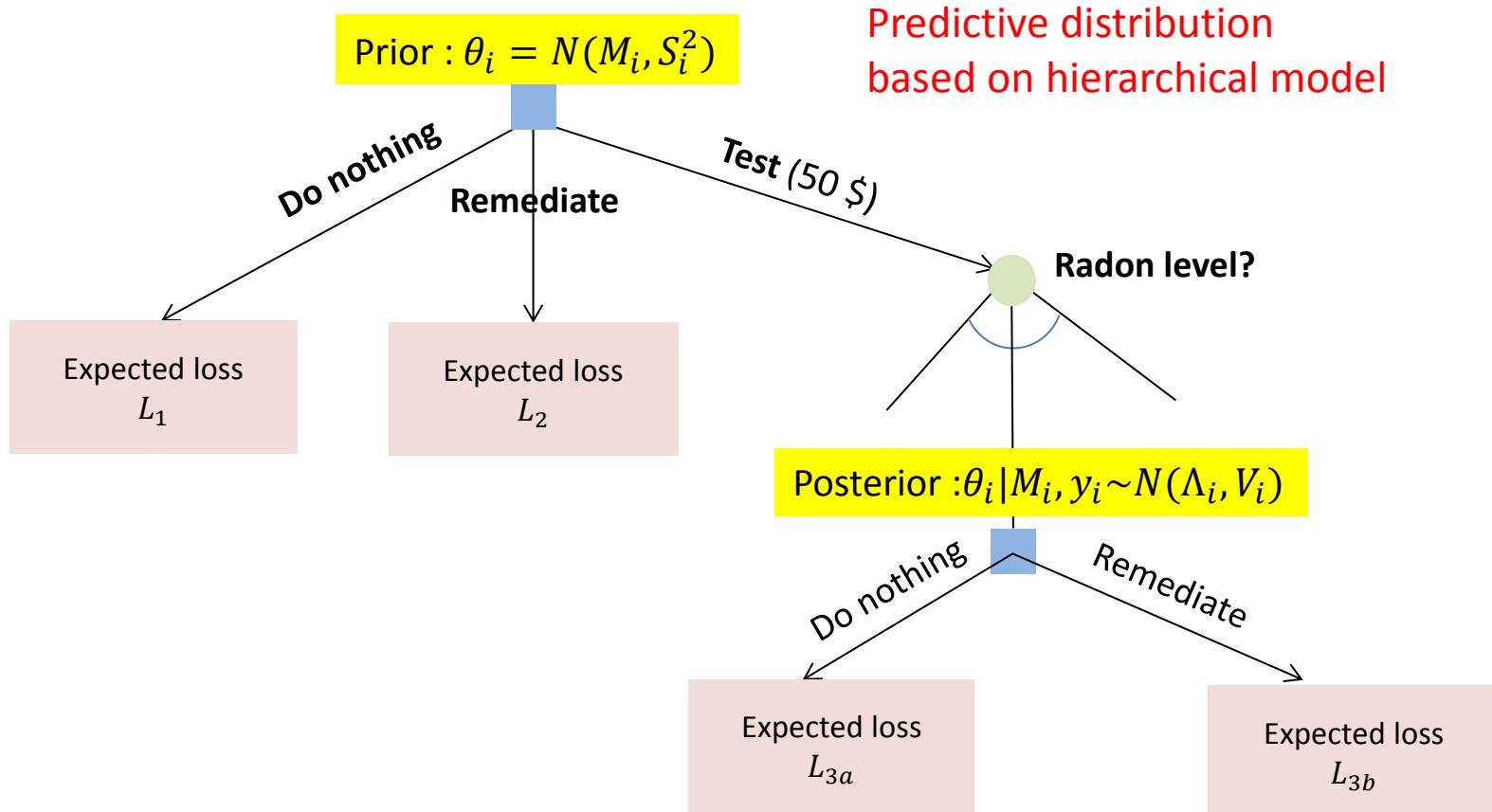
Prior :  $\theta_i = N(M_i, S_i^2)$  Predictive distribution based on hierarchical model

Decision whether to perform measurement test or not

Posterior :  $\theta_i | M_i, y_i \sim N(\Lambda_i, V_i)$

Decision whether to remediate or not

### Example 3: Hierarchical decision analysis for home radon



$$\text{Expected loss} = \text{cost} + D_r E(R)$$

## Example 3: Hierarchical decision analysis for home radon

### Decision analysis for individual homeowners

$$R_i = \text{radon concentration in house } i$$
$$\theta_i = \log(R_i)$$

$$R = e^\theta, \theta \sim N(M, S^2)$$

$$E[R] = E[e^\theta] = e^{M + \frac{1}{2}S^2}$$

$$E[R|\theta > a] = E[e^\theta|\theta > a] = e^{M + \frac{1}{2}S^2} \left( 1 - \Phi\left(\frac{M + S^2 - a}{S}\right) \right)$$

Do nothing

$$L_1 = D_r E(R) = D_r e^{M + \frac{1}{2}S^2}$$

Remediate without test

$$\begin{aligned} L_2 &= \$2000 + D_r E(\min(R, R_{remed})) \\ &= \$2000 + D_r [R_{remed} \Pr(R \geq R_{remed}) + E(R|R < R_{remed}) \Pr(R < R_{remed})] \\ &= \$2000 + D_r \left[ R_{remed} \Phi\left(\frac{M - \log(R_{remed})}{S}\right) + e^{M + \frac{1}{2}S^2} \left( 1 - \Phi\left(\frac{M + S^2 - \log(R_{remed})}{S}\right) \right) \right] \end{aligned}$$

## Example 3: Hierarchical decision analysis for home radon

### Decision analysis for individual homeowners

Remediate without test

$$\begin{aligned}
 L_2 &= \$2000 + D_r E(\min(R, R_{remed})) \\
 &= \$2000 + D_r [R_{remed} \Pr(R \geq R_{remed}) + E(R|R < R_{remed}) \Pr(R < R_{remed})] \\
 &= \$2000 + D_r \left[ R_{remed} \Phi\left(\frac{M - \log(R_{remed})}{S}\right) + e^{M+\frac{1}{2}S^2} \left(1 - \Phi\left(\frac{M + S^2 - \log(R_{remed})}{S}\right)\right) \right]
 \end{aligned}$$

Prior  
( $M, S^2$ )  
↓  
( $\Lambda, V$ )  
Posterior

Perform measurement test

- Do nothing

$$L_{3a} = \$50 + D_r \frac{1}{30} e^{M+\frac{1}{2}S^2} + D_r e^{\Lambda+\frac{1}{2}V}$$

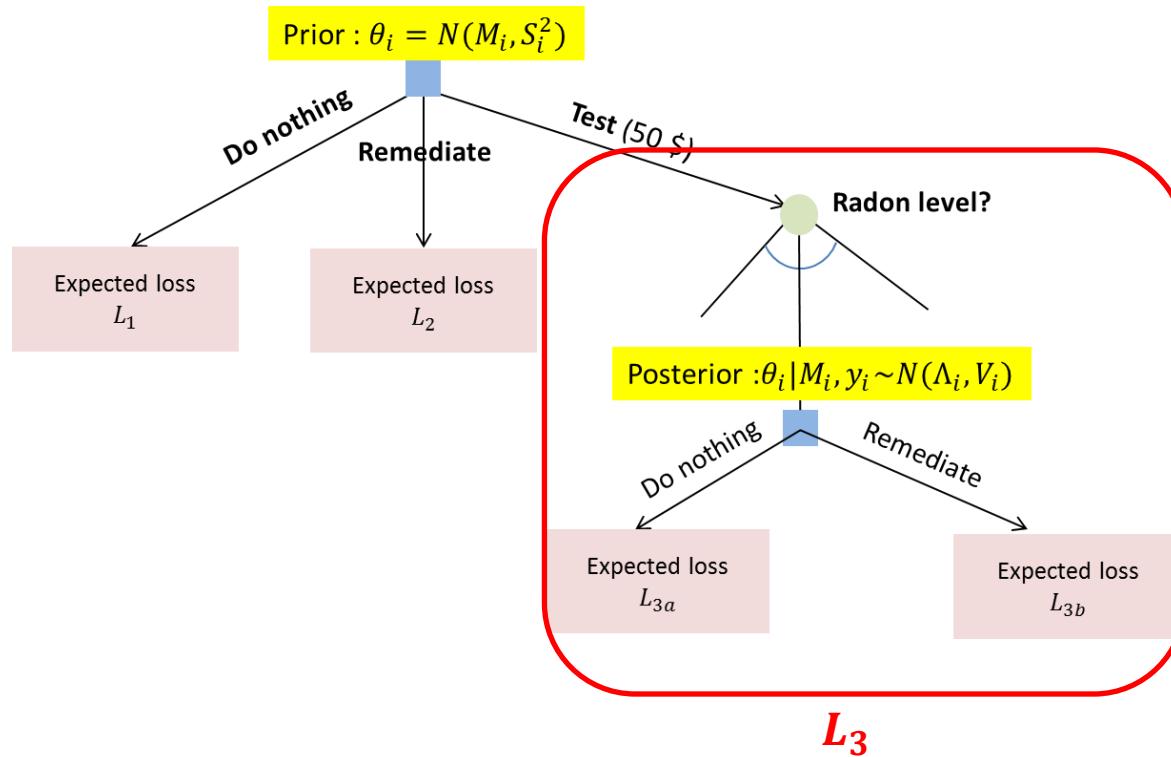
Exposure of 1 year      29 years of exposure after test is done

- Remediate

$$\begin{aligned}
 L_{3b} &= \$50 + D_r \frac{1}{30} e^{M+\frac{1}{2}S^2} + \$2000 \\
 &\quad + D_r \left[ R_{remed} \Phi\left(\frac{\Lambda - \log(R_{remed})}{\sqrt{V}}\right) + e^{M+\frac{1}{2}S^2} \left(1 - \Phi\left(\frac{\Lambda + V - \log(R_{remed})}{\sqrt{V}}\right)\right) \right]
 \end{aligned}$$

## Example 3: Hierarchical decision analysis for home radon

### Decision analysis for individual homeowners

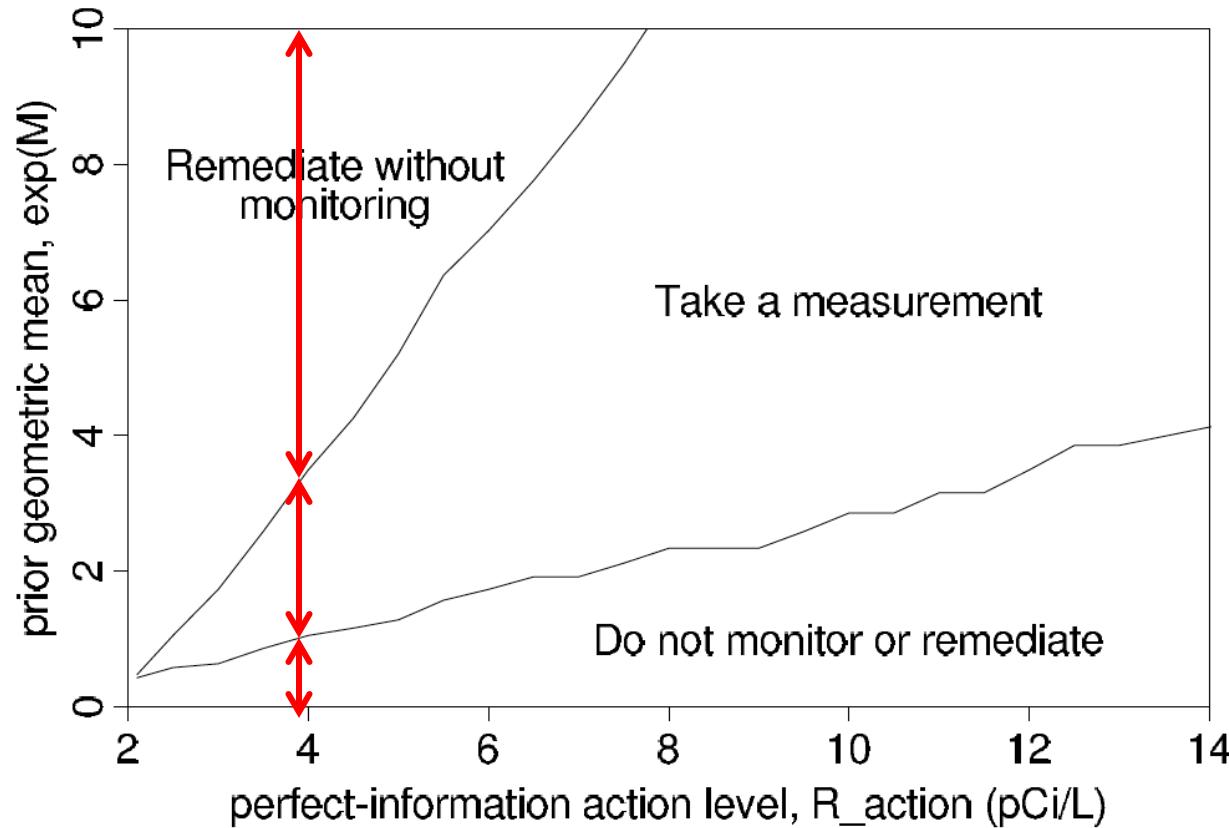


Expected loss for performing test :  $L^3 = E(\min(L_{3a}, L_{3b}))$  :  
 (optimum decision in the second state is imbedded)

1. Simulate 5000 draws of  $y \sim N(M, S^2 + \sigma^2)$  (Considering uncertainty of  $y$ )
2. For each draw of  $y$ , compute  $\min(L_{3a}, L_{3b})$
3. Estimate  $L_3$  as the average of these 5000 values

## Example 3: Hierarchical decision analysis for home radon

### Decision analysis for individual homeowners



## **L7. Machine Learning (Regression)**

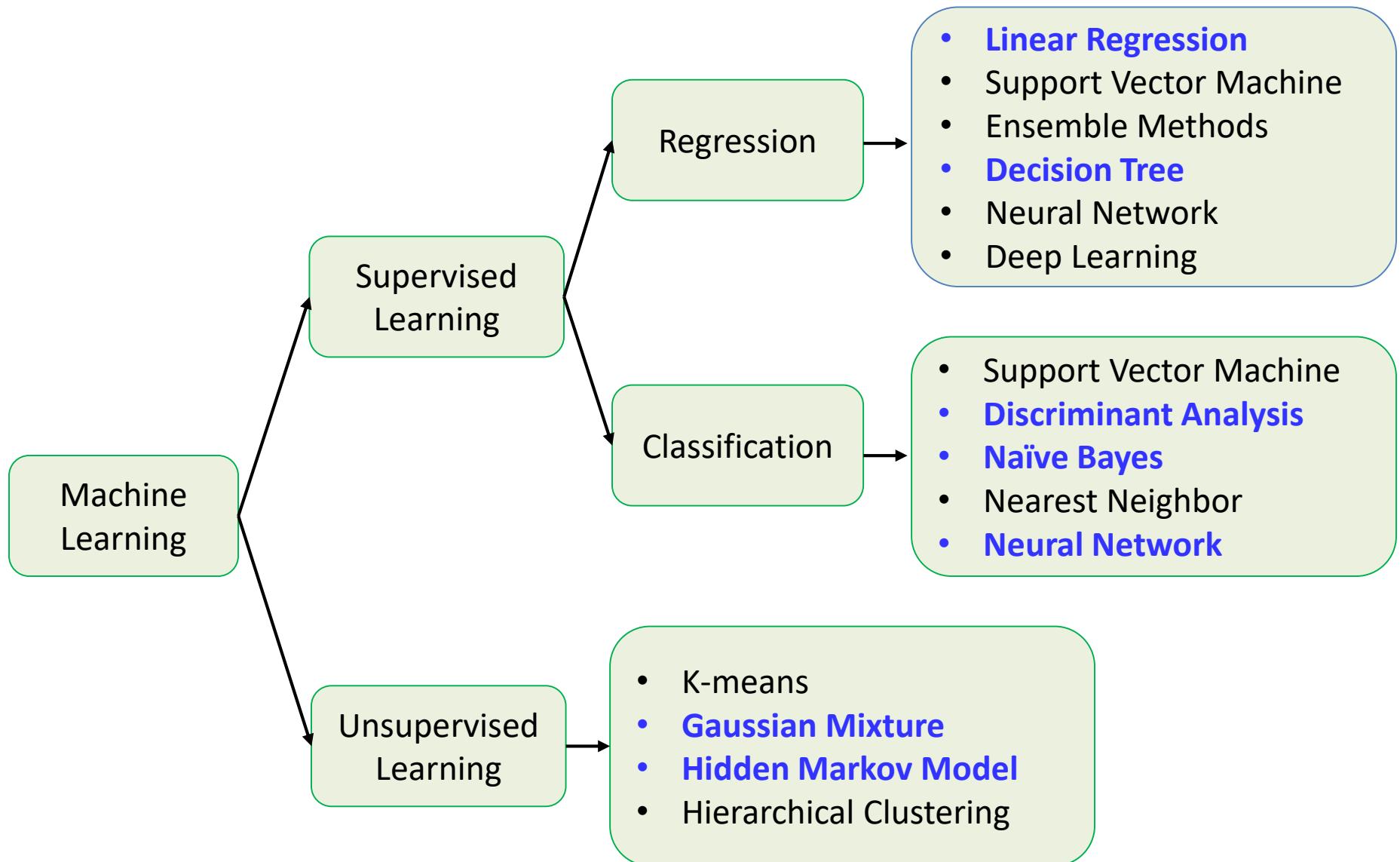
<https://github.com/JWarmenhoven/ISLR-python>

## What is Machine Learning?

data + model = prediction

- Data : observations, experience,...
- Model: a form of prior knowledge, assumptions, belief
  - ✓ Functional model
  - ✓ Probabilistic model
- Prediction : the new knowledge obtained by combining the data and model
  - ✓ Regression
  - ✓ Classification
  - ✓ Clustering

# What is Machine Learning?



# Supervised learning

## Data

$$D = \{(x_i, y_i); i = 1, \dots, m\}$$

$$\mathbf{x}_i = (x_{i1}, \dots, x_{in})$$

$$x_i = (x_{i1}, \dots, x_{in})$$

$x_1$	$x_2$	$\cdots$	$x_n$	$y$
$x_{11}$	$x_{12}$	$\cdots$	$x_{1n}$	$y_1$
$x_{21}$	$x_{22}$	$\cdots$	$x_{2n}$	$y_2$
$\vdots$	$\vdots$		$\vdots$	$\vdots$
$x_{m1}$	$x_{m2}$	$\cdots$	$x_{mn}$	$y_m$

*m input Feature vectors*

*m outputs*

## model

Functional form  $f(x; \theta)$   
Is usually given

Training  
data set  $D$

Learning  
Algorithm

Using training data set, a learning algorithm finds the best hypothesis function  $h(x)$  that **is believed to accurately predict** the output  $y$  for a given query input  $x$

## prediction

Query input

$$x_* \longrightarrow$$

Hypothesis  
 $f(x; \theta^*)$

Predicted output

$$\longrightarrow y_*$$

if  $y_* \in \mathbb{R}$  : **Regression**

if  $y_* \in \{1, \dots, N\}$  : **Classification**

Input feature vector  
 $x_* = (x_{*1}, x_{*2}, \dots, x_{*n})$

## Two different learning approaches

- **Machine Learning as Optimization**
  - ✓ Relate variables through a basis function (parametric function)
  - ✓ Formulate learning problem as an optimization problem
  - ✓ Employ optimization algorithm to solve the formulated problem
- **Machine Learning as Probabilistic Modeling (not necessarily Bayesian)**
  - ✓ Relate variables through probability distributions
  - ✓ Formulate learning problem as inference
  - ✓ If Bayesian, treat parameters with probability distributions
  - ✓ Requires inference methods (integral or sampling) to solve the formulated problem

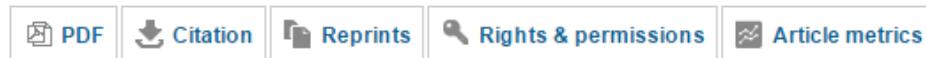
Let's explore different views on Machine Learning by taking a linear regression as an example

## Probabilistic machine learning and artificial intelligence

Zoubin Ghahramani

Nature 521, 452–459 (28 May 2015) | doi:10.1038/nature14541

Received 12 February 2015 | Accepted 21 April 2015 | Published online 27 May 2015



### Abstract

Abstract • Probabilistic modelling and representing uncertainty • Flexibility through non-parametrics • Probabilistic programming • Bayesian optimization • Data compression • Automatic discovery of interpretable models from data • Perspective • References • Acknowledgements • Author information • Comments

How can a machine learn from experience? Probabilistic modelling provides a framework for understanding what learning is, and has therefore emerged as one of the principal theoretical and practical approaches for designing machines that learn from data acquired through experience.

The probabilistic framework, which describes how to represent and manipulate uncertainty about models and predictions, has a central role in scientific data analysis, machine learning, robotics, cognitive science and artificial intelligence. This Review provides an introduction to this framework, and discusses some of the state-of-the-art advances in the field, namely, probabilistic programming, Bayesian optimization, data compression and automatic model discovery.



How is oil wealth fuelling Saudi Arabia's science output?

### Editors' pick



Image credit: Desmond Boylan for Nature

Cuban science unleashed: With new freedom to interact with the world, Cuban researchers are hoping for evolution, if not revolution. ►

Science jobs

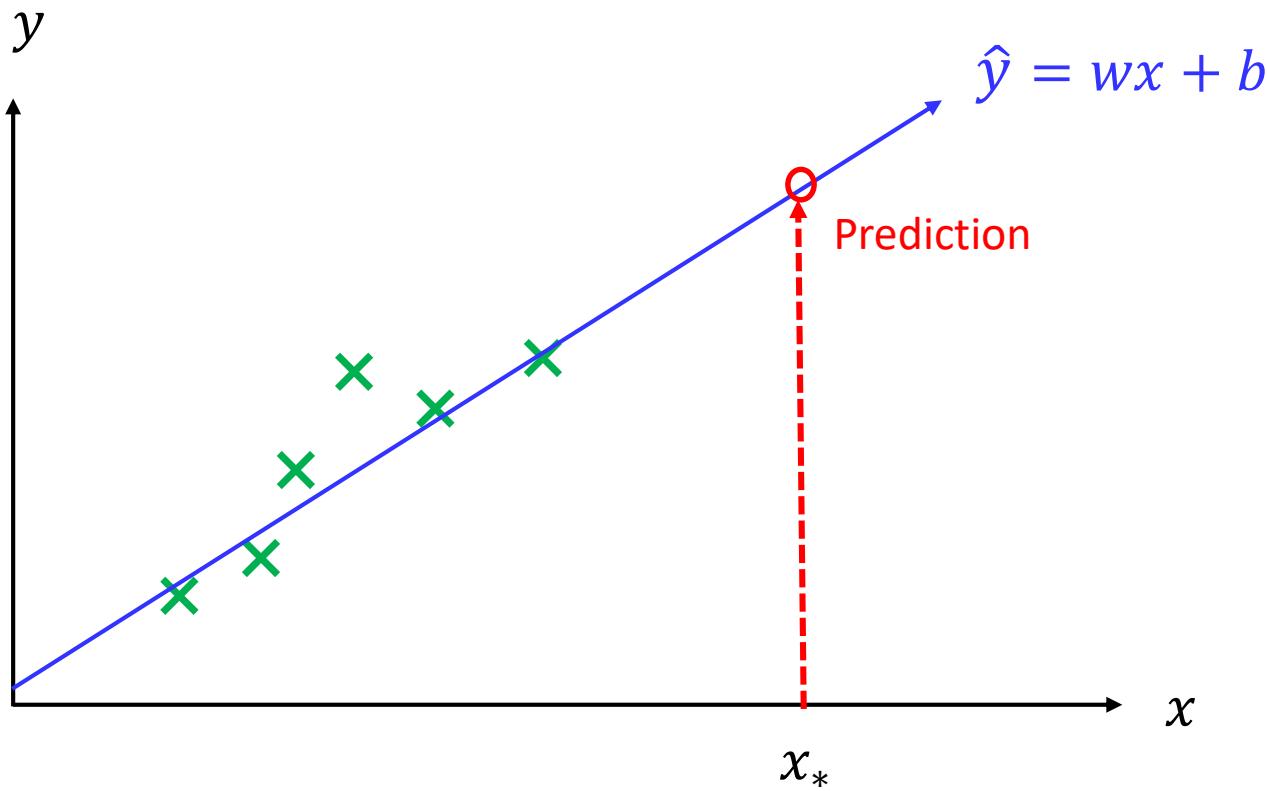
Science events

**nature** events directory

3rd International Conference on Self

1. Optimization Approach (Normal Equation)
2. Maximum Likelihood Estimation (MLE) Approach
3. Maximum A Posteriori Estimation (MAP) Approach
4. Full Bayesian Approach
  - ✓ Analytical approach
  - ✓ Sampling approach
5. Regularization regression (Ridge and Lasso)
  - ✓ Optimization view
  - ✓ Bayesian View

## 1D Linear Regression

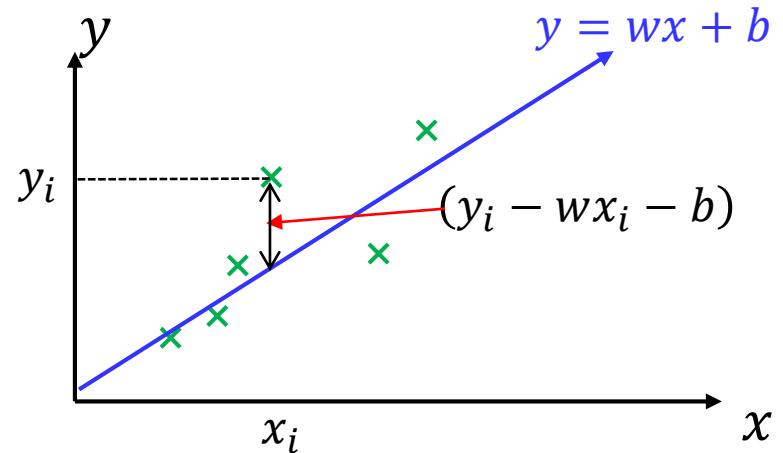


- **Data** :  $(x_1, y_1), \dots, (x_m, y_m)$
- **Model**: Linear model  $\hat{y} = wx + b$  ( $y = w^T x + b$  for multidimensional)
- **Learning**: What are  $w$  and  $b$ ?
- **Prediction** : What is  $\hat{y}_* = wx_* + b$

## Learning as optimization

- Define an objective (cost) function

$$J(w, b) = \sum_{i=1}^m (y_i - wx_i - b)^2$$



- Minimize the error function with respect to  $w$  and  $b$

$$\frac{dJ(w, b)}{dw} = -2 \sum_{i=1}^m x_i (y_i - wx_i - b) = 0 \rightarrow w^* = \frac{\sum_{i=1}^m (y_i - b)x_i}{\sum_{i=1}^n x_i^2}$$

$$\frac{dJ(w, b)}{db} = -2 \sum_{i=1}^m (y_i - wx_i - b) = 0 \rightarrow b^* = \frac{\sum_{i=1}^m (y_i - wx_i)}{n}$$

## Learning as optimization

Notation for general cases  $x_i \in \mathbb{R}^n$

- A linear regression model

$$\hat{y}_i = w_0 + w_1 x_{i1} + \cdots + w_n x_{in}$$

with  $w = (w_0, w_1, \dots, w_n)^T$  and  $x_i = (x_{i1}, \dots, x_{in})^T$

- If we introduce  $x_{i0} = 1$ ,

$$\hat{y}_i = w^T x_i$$

with  $w = (w_0, w_1, \dots, w_n)^T$  and  $x_i = (x_{i0}, x_{i1}, \dots, x_{in})^T$

- In a Matrix form

$$\begin{pmatrix} \hat{y}_1 \\ \vdots \\ \hat{y}_m \end{pmatrix} = \begin{pmatrix} -x_1^T & - \\ \vdots & \\ -x_m^T & - \end{pmatrix} \begin{pmatrix} w_0 \\ \vdots \\ w_n \end{pmatrix} = \begin{pmatrix} x_1^T w \\ \vdots \\ x_m^T w \end{pmatrix} \rightarrow \hat{y} = Xw$$

$m$ : # of data points

with  $\hat{y} = (\hat{y}_1, \dots, \hat{y}_m)^T$ ,  $X = \begin{pmatrix} -x_1^T & - \\ \vdots & \\ -x_m^T & - \end{pmatrix}$

## Learning as optimization (Normal Equation)

- The cost function for the optimization can be defined as :

$$J(w) = \frac{1}{2} \sum_{i=1}^m (x_i^T w - y_i)^2 = \frac{1}{2} \|y - wX\|_2^2 = \frac{1}{2} (Xw - y)^T (Xw - y)$$
$$\sqrt{\sum_{i=1}^m z_i^2} = \|z\|_2 = \sqrt{z^T z}$$

- The optimum parameters  $\hat{w}$  can be computed as one minimizing the cost function :

$$\hat{w} = \arg \min_w J(w) = \arg \min_w \frac{1}{2} \|y - wX\|_2^2$$

- For reference, other norms are summarized here :

$$\|z\|_1 = \sqrt{\sum_{i=1}^n |z_i|}, \quad \|z\|_p = \sqrt{\sum_{i=1}^n |z_i|^p}, \quad \|z\|_\infty = \max_i |z_i|$$

## Learning as optimization (Normal Equation)

Linear Algebra Approach for finding the optimum parameters:

$$\hat{w} = \arg \min_w J(w) = \arg \min_w \frac{1}{2} \|y - wX\|_2^2$$

Optimality condition :  $\nabla_w J(w) = 0$  at  $\hat{w}$

$$\begin{aligned}\nabla_w J(w) &= \nabla_w \frac{1}{2} (Xw - y)^T (Xw - y) \\&= \frac{1}{2} \nabla_w (w^T X^T X w - w^T X^T y - y^T X w + y^T y) \\&= \frac{1}{2} \nabla_w \text{tr}(w^T X^T X w - w^T X^T y - y^T X w + y^T y) \\&= \frac{1}{2} \nabla_w (\text{tr}(w^T X^T X w) - 2\text{tr}(y^T X w)) \\&= \frac{1}{2} (X^T X w + X^T X w - 2X^T y) \\&= X^T X w - X^T y\end{aligned}$$

$$\nabla_w J(w) = X^T X w - X^T y = 0$$

$$\rightarrow X^T X w = X^T y$$

$$\rightarrow \hat{w} = (X^T X)^{-1} X^T y$$

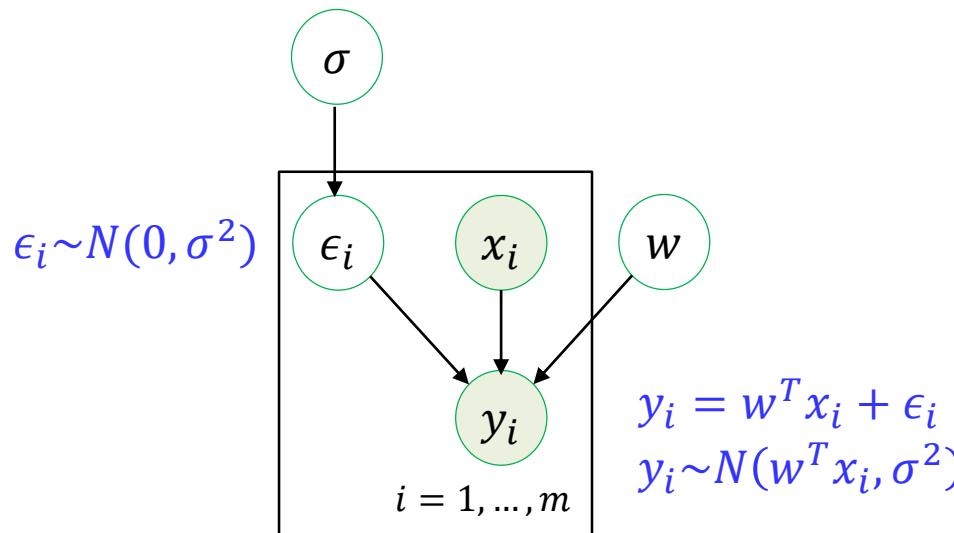
## Probabilistic view on linear regression

- Assume there is uncertainty in the predicted value :

$$y_i = w^T x_i + \epsilon_i \text{ with } \epsilon_i \sim N(0, \sigma^2)$$

- Then the probabilistic model on output  $y_i$  can be represented as

$$y_i \sim N(w^T x_i, \sigma^2) \text{ or } p(y_i | w^T x_i, \sigma) = N(y_i | w^T x_i, \sigma^2)$$



An error  $\epsilon_i$  is independently identically distributed (i.i.d assumption)

- The likelihood of the data is defined as

$$p(y|X, w, \sigma) = \prod_{i=1}^m N(y_i | w^T x_i, \sigma^2) = \prod_{i=1}^m \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_i - w^T x_i)^2}{2\sigma^2}\right)$$

## Learning as probabilistic model (MLE Approach)

- The log likelihood is

$$\begin{aligned} L(w, \sigma) &= \log p(y|X, w, \sigma) \\ &= \log \prod_{i=1}^m \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_i - w^T x_i)^2}{2\sigma^2}\right) \\ &= \log\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^m \exp\left(-\sum_{i=1}^m \frac{(y_i - w^T x_i)^2}{2\sigma^2}\right) \\ &= m \log \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{\sigma^2} \underbrace{\frac{1}{2} \sum_{i=1}^m}_{\textcolor{red}{m}} (y_i - w^T x_i)^2 \\ &= m \log \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{\sigma^2} J(w) \end{aligned}$$

- The optimum parameters is determined by maximizing log likelihood

$$(w^*, \sigma) = \max_{(w, \sigma)} L(w, \sigma) = \max_{(w, \sigma)} \log p(y|X, w, \sigma)$$

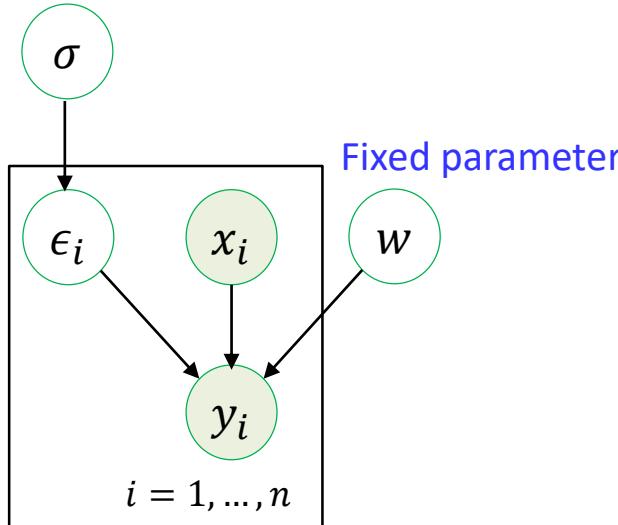
Minimizing the square error sum  $J(w) =$

Maximizing the log likelihood  $\log p(y|X, w, \sigma)$  with respect to  $w$

## Learning as probabilistic model (Bayesian Approach)

### MLE approach (point estimation)

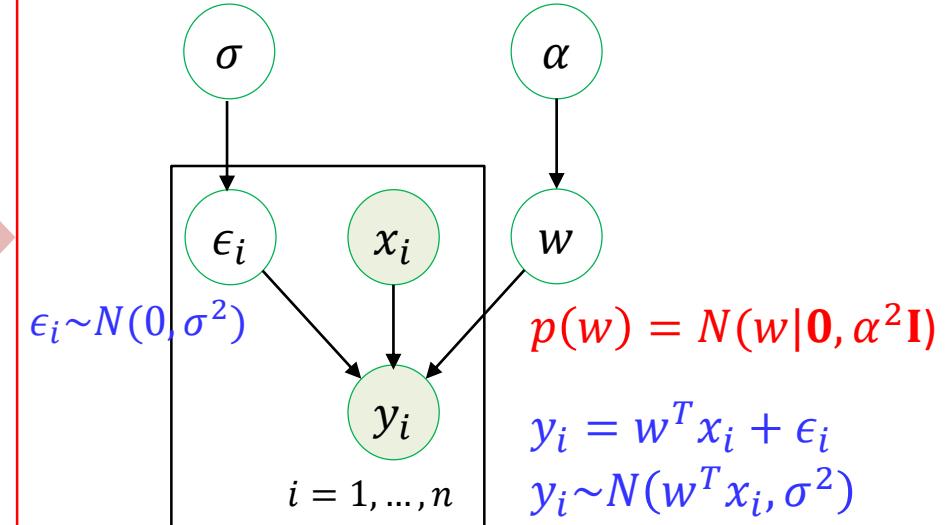
Fixed hyper-parameter



Fixed parameter

### Bayesian approach

Fixed hyper-parameter



- Consider the parameter  $w$  as **stochastic variables** (represented as a distribution)
- (Assume  $\sigma$  is known for simple derivation)
- Find the distribution on parameter  $w$

$$p(w|y, X) = \frac{p(y|X, w)p(w|X)}{\int_w p(y|X, w)p(w|X)dw} \rightarrow p(w|y) = \frac{p(y|w)p(w)}{\int_w p(y|w)p(w)dw}$$

We will assume  $X$  is fixed for the data  $y$

## Learning as probabilistic model (Bayesian Approach)

- Multivariate regression likelihood is

$$\begin{aligned} p(y|w) &= \prod_{i=1}^m p(y_i|x_i, w) \\ &= \frac{1}{(2\pi\sigma^2)^{m/2}} \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^m (y_i - w^T x_i)^2\right) \end{aligned}$$

*m = # of data points*

- Multivariate Gaussian prior on parameter  $w$

$$p(w) = N(w|\mathbf{0}, \alpha^2 \mathbf{I})$$

$$p(w) = \frac{1}{(2\pi\alpha^2)^{n/2}} \exp\left(-\frac{1}{2\alpha^2} w^T w\right)$$

*n = Dimension of w*

## Learning as probabilistic model (Bayesian Approach)

- We want to find the posterior

$$p(w|X, y) \propto p(y|X, w)p(w)$$

$$= \frac{1}{(2\pi\sigma^2)^{m/2}} \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^m (y_i - w^T x_i)^2\right) \frac{1}{(2\pi\alpha^2)^{k/2}} \exp\left(-\frac{1}{2\alpha^2} w^T w\right)$$

- Take log:

$$\begin{aligned} \log p(w|X, y) &= -\frac{1}{2\sigma^2} \sum_{i=1}^m (y_i - w^T x_i)^2 - \frac{1}{2\alpha^2} w^T w + const \\ &= -\frac{1}{2\sigma^2} \sum_{i=1}^m y_i^2 + \frac{1}{\sigma^2} \sum_{i=1}^m y_i x_i^T w - \frac{1}{2\sigma^2} \sum_{i=1}^m w^T x_i x_i^T w - \frac{1}{2\alpha^2} w^T w + const \\ &= -\frac{1}{2\sigma^2} y^T y + \frac{1}{\sigma^2} y^T X w - \frac{1}{2\sigma^2} w^T X^T X w - \frac{1}{2\alpha^2} w^T w + const \\ &= -\frac{1}{2\sigma^2} y^T y + \frac{1}{\sigma^2} y^T X w - \frac{1}{2} w^T \left[ \frac{1}{\sigma^2} X^T X + \frac{1}{\alpha^2} I \right] w + const \end{aligned}$$

- Posterior distribution is

$$p(w|X, y) = N(w|\mu_w, \Sigma_w)$$

$$\mu_w = \Sigma_w \left( \frac{1}{\sigma^2} X^T y \right) \quad \Sigma_w = \left[ \frac{1}{\sigma^2} X^T X + \frac{1}{\alpha^2} I \right]^{-1}$$

## Learning as probabilistic model (Bayesian Approach)

- Predicative distribution

$$p(y_*|x_*, X, y) = \int_w p(y_*|x_*, w)p(w|X, y)dw$$

Jupyter Demo Simulation  
Bayesian Regression **Analytical**

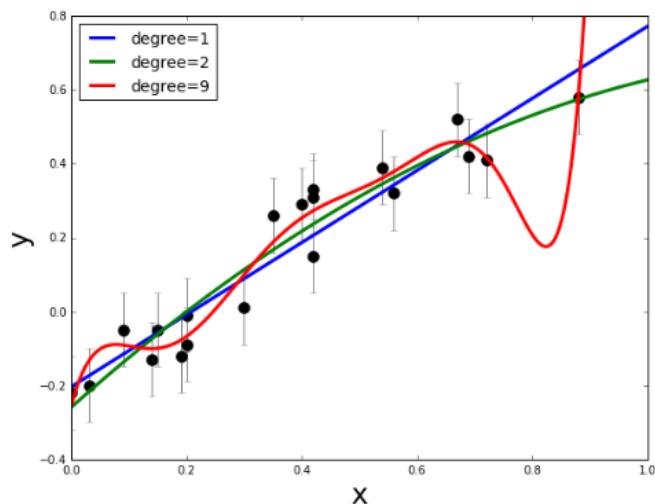
## Learning as probabilistic model (Bayesian Approach)

- Predicative distribution

$$p(y_*|x_*, X, y) = \int_w p(y_*|x_*, w)p(w|X, y)dw$$

Jupyter Demo Simulation  
Bayesian Regression ([Sampling using PyMC](#))

## Regularized linear regression



What is a good regression function?

- The goal of regression is to come up with some good prediction function:

$$\hat{f}(x) = \hat{w}^T x^T$$

- So far, we have found  $\hat{w}$  by finding (Ordinary Least Square Estimation)

$$\hat{w} = \arg \min_w J(w) = \arg \min_w \frac{1}{2} \|y - wX\|_2^2$$

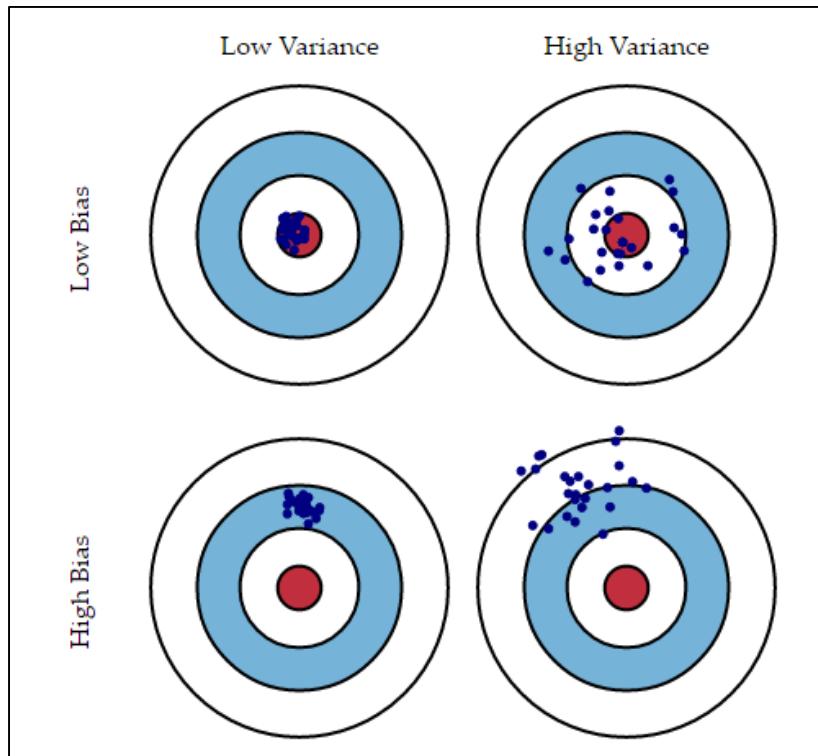
- To see  $\hat{f}(x)$  is good candidate, we need to check

- ✓ Is  $\hat{w}$  close to the true  $w$
- ✓ Will  $\hat{f}(x)$  fit future observation well? (**Generalization**)

## Regularized linear regression

Jupyter Demo Simulation  
Regularization Motivation Example

## The Bias and Variance Trade off



Each hit represents an individual realization of our model, given the chance variability in the training data we gather.

- Imagine you could repeat the whole model building process more than once: each time you gather new data and run a new analysis creating a new model.
- Due to randomness in the underlying data sets, the resulting models will have a range of predictions.
  - **Bias** measures how far off in general these models' predictions are from the correct value
  - **Variance** measures the variability of a model prediction for a given data point

## The Bias and Variance Trade off

Estimation :  $\hat{f}(x) = x^T \hat{w}$

True :  $f(x)$

Observation :  $y = f(x) + \epsilon, \epsilon \sim N(0, \sigma^2)$

- *The expected prediction error* of a regression fit  $\hat{f}(x_0)$ , using square-loss error :

$$\begin{aligned} \text{EPP}(x_0) &= E\left[\left(y - \hat{f}(x_0)\right)^2 \middle| x_0\right] \\ &= E[y^2 + \hat{f}(x_0)^2 - 2y\hat{f}(x_0) \mid x_0] \\ &= E[y^2 \mid x_0] + E[\hat{f}(x_0)^2] - E[2y\hat{f}(x_0) \mid x_0] \\ &= E[y^2 \mid x_0] + E[\hat{f}(x_0)^2] - 2f(x_0)E[\hat{f}(x_0)] \end{aligned}$$

$$\begin{aligned} E[2y\hat{f}(x_0) \mid x_0] &= E[2(f(x) + \epsilon)\hat{f}(x_0) \mid x_0] \\ &= 2E[f(x)\hat{f}(x_0) \mid x_0] + 2E[\epsilon\hat{f}(x_0) \mid x_0] \\ &= 2f(x_0)E[\hat{f}(x_0)] + 2E[\epsilon\hat{f}(x_0) \mid x_0] \quad (\because f(x_0) \text{ is constant}) \\ &= 2f(x_0)E[\hat{f}(x_0)] \quad (\because \epsilon \perp \hat{f}(x_0)) \end{aligned}$$

## The Bias and Variance Trade off

Estimation :  $\hat{f}(x) = x^T \hat{w}$

True :  $f(x)$

Observation :  $y = f(x) + \epsilon, \epsilon \sim N(0, \sigma^2)$

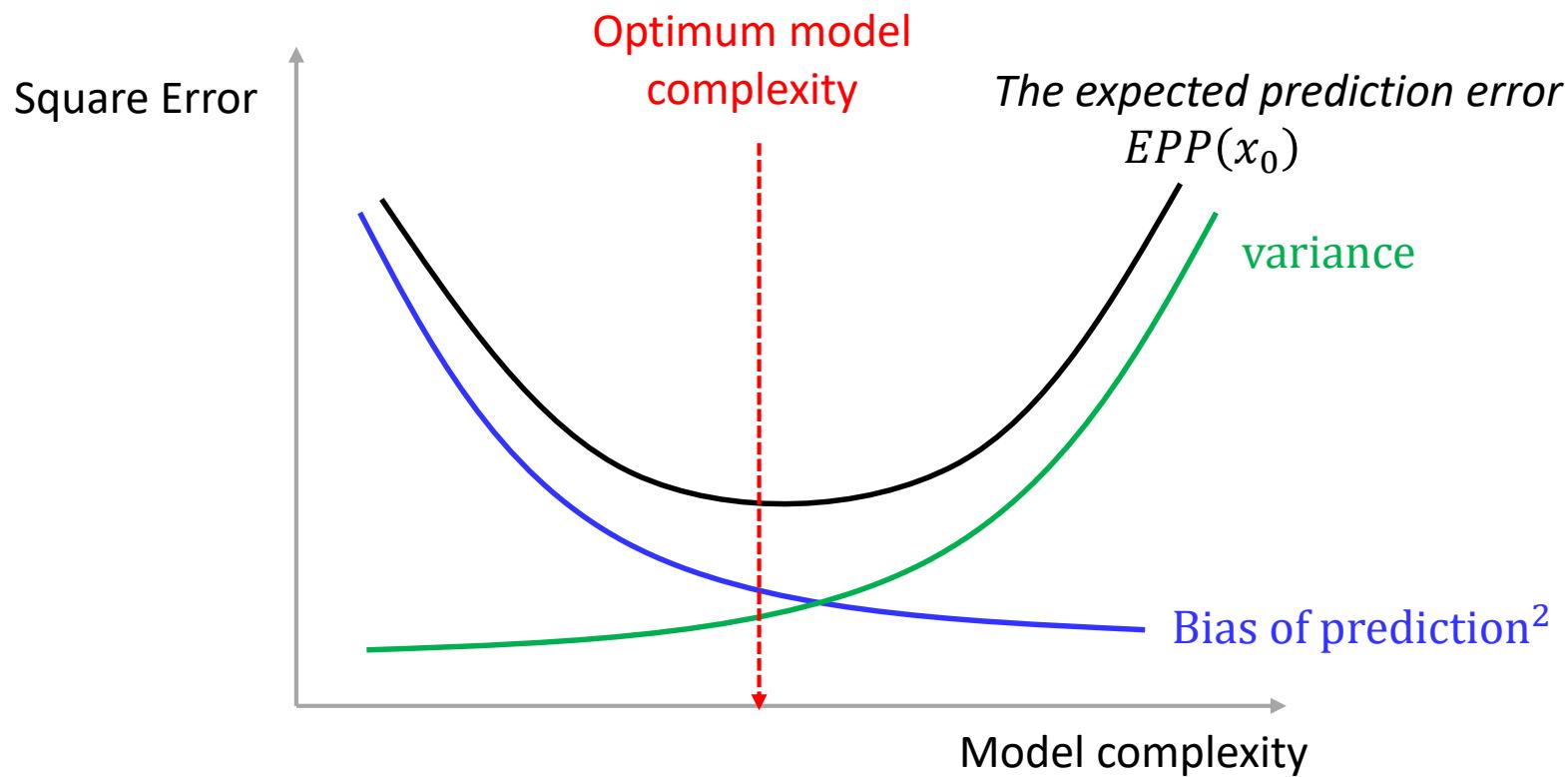
- The expected prediction error* of a regression fit  $\hat{f}(x_0)$ , using square-loss error :

$$\begin{aligned}
 \text{EPP}(x_0) &= E \left[ (y - \hat{f}(x_0))^2 \mid x_0 \right] && \text{E is over the training data} \\
 &= E[y^2 + \hat{f}(x_0)^2 - 2y\hat{f}(x_0) \mid x_0] \\
 &= E[y^2 \mid x_0] + E[\hat{f}(x_0)^2] - E[2y\hat{f}(x_0) \mid x_0] \\
 &= E[y^2 \mid x_0] + E[\hat{f}(x_0)^2] - 2f(x_0)E[\hat{f}(x_0)] \\
 &= \text{var}[y \mid x_0] + E[y \mid x_0]^2 + \text{var}[\hat{f}(x_0)] + E[\hat{f}(x_0)]^2 - 2f(x_0)E[\hat{f}(x_0)] \\
 &= \text{var}[y \mid x_0] + \text{var}[\hat{f}(x_0)] + E[y \mid x_0]^2 + E[\hat{f}(x_0)]^2 - 2f(x_0)E[\hat{f}(x_0)] \\
 &= \text{var}[y \mid x_0] + \text{var}[\hat{f}(x_0)] + f(x_0)^2 + E[\hat{f}(x_0)]^2 - 2f(x_0)E[\hat{f}(x_0)] \\
 &= \text{var}[y \mid x_0] + \text{var}[\hat{f}(x_0)] + (f(x_0) - E[\hat{f}(x_0)])^2 \\
 &= \text{var}[y \mid x_0] + E[(\hat{f}(x_0) - E[\hat{f}(x_0)])^2] + (f(x_0) - E[\hat{f}(x_0)])^2 \\
 &= \sigma^2 + \text{variance of prediction} + \text{Bias of prediction}^2
 \end{aligned}$$

Irreducible

Can balance

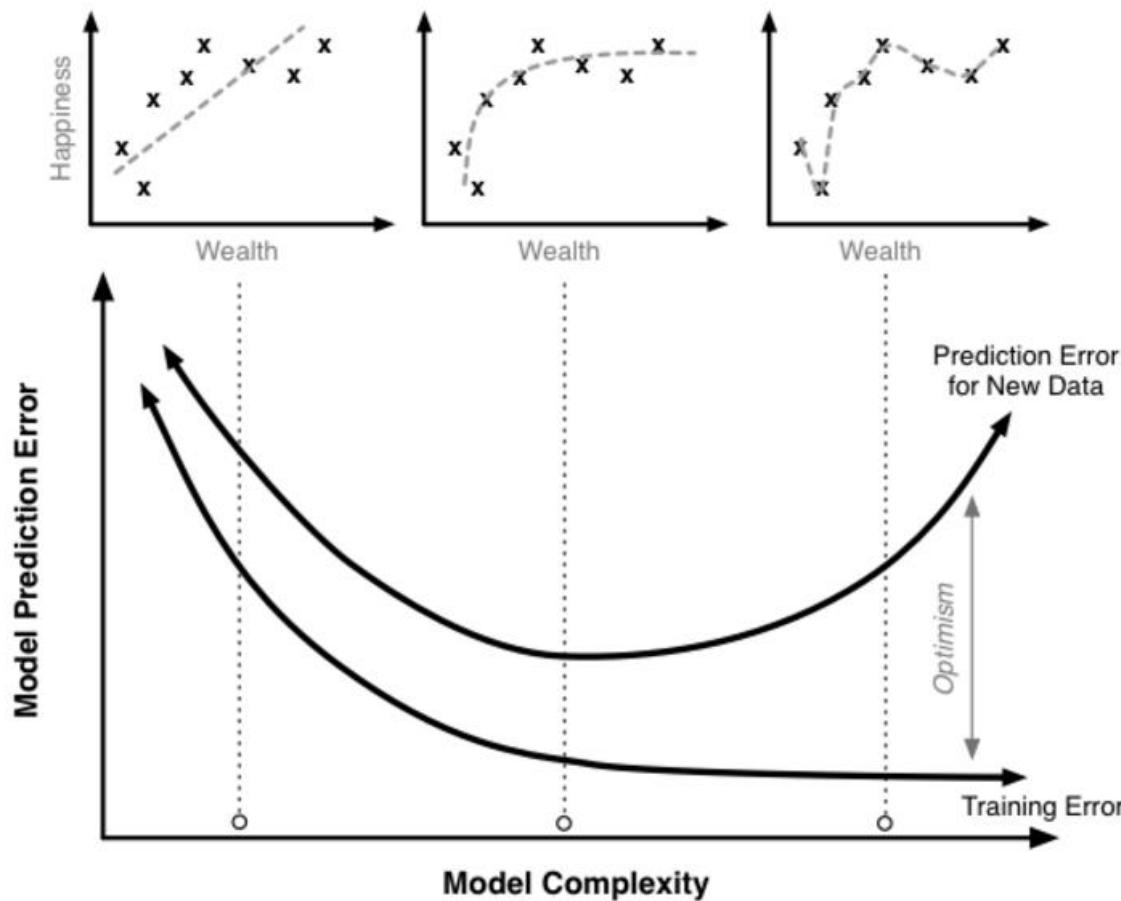
## The Bias and Variance Trade off



Model complexity is related with

- The number of model parameters
- The size of model parameters
- ...

## How to measure prediction error



## How to measure (estimate) model prediction error?

- Statistical measure (i.e.,  $R^2$  value)
- Information Theoretic Approaches (i.e., BIC, AIC measure)
- Holdout set or Cross Validation (Training data vs Test data set)

## Regularized linear regression

### Ordinary Least Square Estimation

- OLS estimates find the parameter that minimize the bias between the predicted and true values :

$$\hat{w} = \operatorname{argmin}_w \|y - wX\|_2^2$$

- OLS estimates often have low bias but large variance

→ Poor generalization toward unseen test data set

- All features have a weight

→ Smaller subset with strong effects is more interpretable

- $w_i'$ 's are unconstrained

→ They can explore and hence are susceptible to very high variance

We nee some shrinkage (or regulation) to constraint  $\hat{w}$

## Regularized linear regression

### Ridge regression

- Ridge regression introduces a regularization with the L-2 norm:

$$\hat{w} = \operatorname{argmin}_w \|y - wX\|_2^2 + \lambda_2 \|w\|_2^2 \quad \|w\|_2 = \sqrt{\sum_{i=1}^k w_i^2},$$

- Sacrifice a little of bias to reduce the variance of predicted values  
→ More stable and generalize better
- Keep all the repressors in the model  
→ Not easily interpretable model

### Lasso (Least Absolute Shrinkage and Selection Operator)

- Lasso regression introduces a regularization with the L-1 norm:

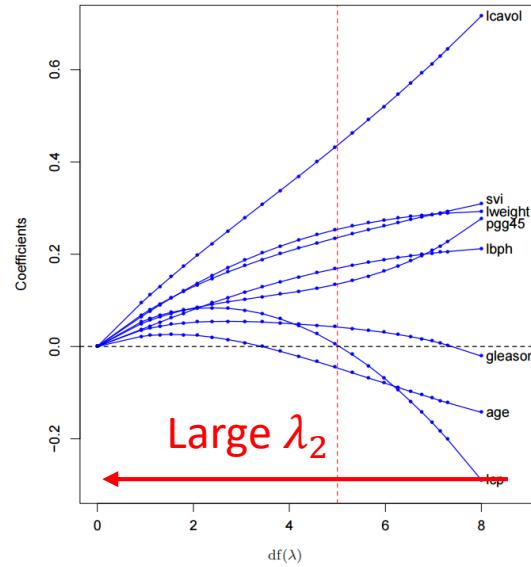
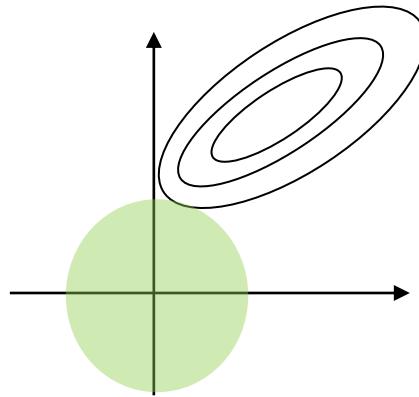
$$\hat{w} = \operatorname{argmin}_w \|y - wX\|_2^2 + \lambda_1 \|w\|_1 \quad \|w\|_1 = \sum_{i=1}^k |w_i|$$

- Only a small subset of features with  $\hat{w}_i \neq 0$  are selected  
→ Increases the interpretability
- More difficult to implement than Ridge Regression

## Regularized linear regression

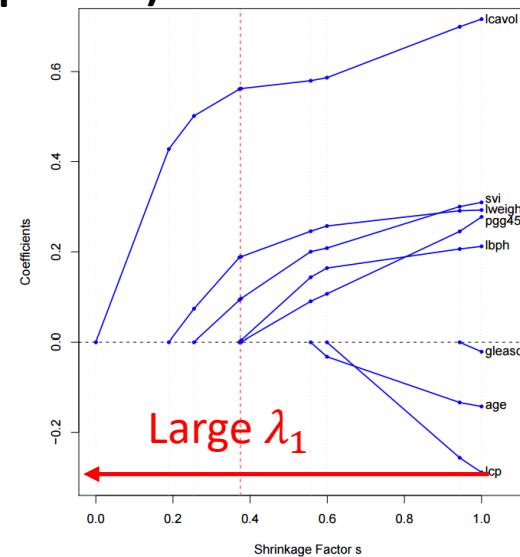
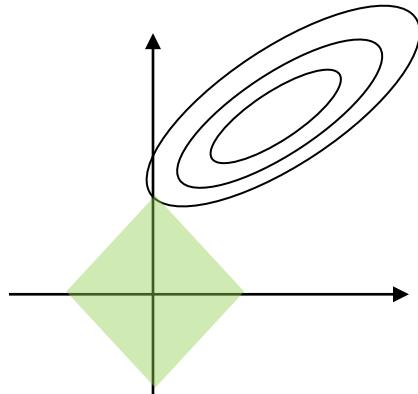
### Ridge regression

$$\hat{w} = \underset{w}{\operatorname{argmin}} \|y - wX\|_2^2 + \lambda_2 \|w\|_2^2$$



### Lasso (Least Absolute Shrinkage and Selection Operator)

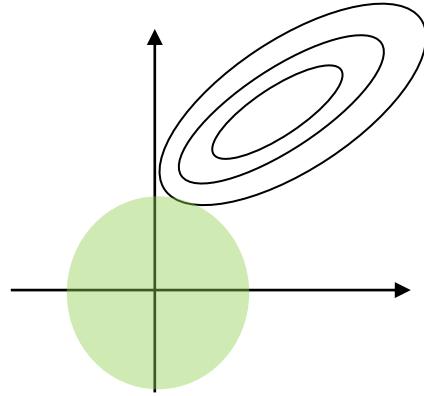
$$\hat{w} = \underset{w}{\operatorname{argmin}} \|y - wX\|_2^2 + \lambda_1 \|w\|_1$$



## Bayesian view on Ridge regression

### Ridge regression

$$\hat{w} = \operatorname{argmin}_w \|y - wX\|_2^2 + \lambda_2 \|w\|_2^2$$



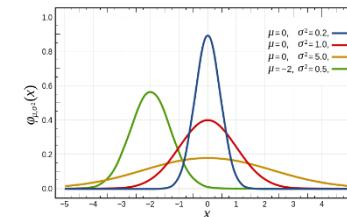
### MAP estimation view

$$\hat{w} = \operatorname{argmax}_w \log p(w|X, y) = p(y|X, w)p(w)$$

Gaussian prior

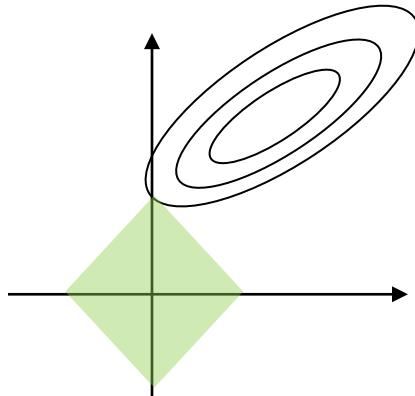
$$p(w) = \frac{1}{(2\pi\alpha^2)^{k/2}} \exp\left(-\frac{1}{2\alpha^2} w^T w\right)$$

MAP estimation view



### Lasso (Least Absolute Shrinkage and Selection Operator)

$$\hat{w} = \operatorname{argmin}_w \|y - wX\|_2^2 + \lambda_1 \|w\|_1$$

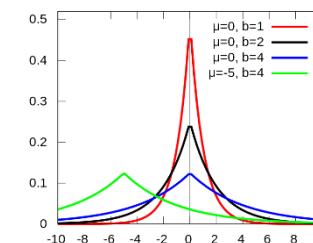


$$\hat{w} = \operatorname{argmax}_w \log p(w|X, y) = p(y|X, w)p(w)$$

Laplace prior

$$p(w) = \prod_{i=1}^k \frac{\lambda}{2\sqrt{\tau^2}} \exp\left(-\frac{\lambda|w_i|}{\sqrt{\tau^2}}\right)$$

MAP estimation view



## Bayesian view on Ridge regression

- $$p(y|X, w) = \prod_{i=1}^m \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left(-\frac{(y_i - w^T x_i)^2}{2\sigma^2}\right)$$
 $m$  : number of data points  
 $n$  : dimension of  $w$
- $$p(w) = N(w|\mathbf{0}, \tau^2 \mathbf{I}) = \prod_{i=1}^n \frac{1}{(2\pi\tau^2)^{1/2}} \exp\left(-\frac{(w_i - 0)^2}{2\tau^2}\right) = \frac{1}{(2\pi\tau^2)^{n/2}} \exp\left(-\frac{1}{2\tau^2} w^T w\right)$$
- $$\begin{aligned} p(w|X, y) &\propto p(y|X, w)p(w) \\ &= \prod_{i=1}^m \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left(-\frac{(y_i - w^T x_i)^2}{2\sigma^2}\right) \frac{1}{(2\pi\tau^2)^{n/2}} \exp\left(-\frac{1}{2\tau^2} w^T w\right) \\ &= \left(\frac{1}{(2\pi\sigma^2)^{1/2}}\right)^m \frac{1}{(2\pi\tau^2)^{n/2}} \exp\left(-\sum_{i=1}^m \frac{(y_i - w^T x_i)^2}{2\sigma^2} - \frac{1}{2\tau^2} w^T w\right) \end{aligned}$$
- $$\log p(w|X, y) = m \log \frac{1}{(2\pi\sigma^2)^{1/2}} + \log \frac{1}{(2\pi\tau^2)^{n/2}} - \frac{1}{2\sigma^2} \left( \sum_{i=1}^m (y_i - w^T x_i)^2 + \frac{\sigma^2}{\tau^2} w^T w \right)$$
- Maximum A Posteriori (MAP) estimation with Gaussian prior = **Ridge Regression**

$$(w^*) = \operatorname{argmax}_w \log p(w|X, y) = \operatorname{argmin}_w \|y - wX\|_2^2 + \lambda_2 \|w\|_2^2$$

## Bayesian view on Lasso regression

- $p(y|X, w) = \prod_{i=1}^m \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left(-\frac{(y_i - w^T x_i)^2}{2\sigma^2}\right)$
- $p(w) = \text{Lap}(w|\lambda, \tau) = \prod_{i=1}^n \frac{\lambda}{2\sqrt{\tau^2}} \exp\left(-\frac{\lambda|w_i|}{\sqrt{\tau^2}}\right)$
- $$\begin{aligned}
 p(w|X, y) &\propto p(y|X, w)p(w) \\
 &= \prod_{i=1}^m \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left(-\frac{(y_i - w^T x_i)^2}{2\sigma^2}\right) \left(\frac{\lambda}{2\sqrt{\tau^2}}\right)^n \exp\left(-\frac{\lambda}{\sqrt{\tau^2}} \sum_{i=1}^n |w_i|\right) \\
 &= \left(\frac{1}{(2\pi\sigma^2)^{1/2}}\right)^m \left(\frac{\lambda}{2\sqrt{\tau^2}}\right)^n \exp\left(-\sum_{i=1}^m \frac{(y_i - w^T x_i)^2}{2\sigma^2} - \frac{\lambda}{\sqrt{\tau^2}} \sum_{i=1}^n |w_i|\right)
 \end{aligned}$$
- $$\log p(w|X, y) = m \log \frac{1}{(2\pi\sigma^2)^{1/2}} + n \log \frac{\lambda}{2\sqrt{\tau^2}} - \frac{1}{2\sigma^2} \left( \sum_{i=1}^m (y_i - w^T x_i)^2 + \frac{2\sigma^2 \lambda}{\sqrt{\tau^2}} \sum_{i=1}^n |w_i| \right)$$
- Maximum A Posteriori estimation with Laplacian prior = **Lasso regression**

$$(w^*) = \operatorname{argmax}_w \log p(w|X, y) = \operatorname{argmin}_w \|y - wX\|_2^2 + \lambda_1 \|w\|_1$$

$m$  : number of data points

$n$  : dimension of  $w$

## Bayesian view on Lasso regression

Jupyter Demo Simulation  
Ridge and Lasso Simulation

# **Bayesian Model Selection**

## Bayesian Approach for Model Selection

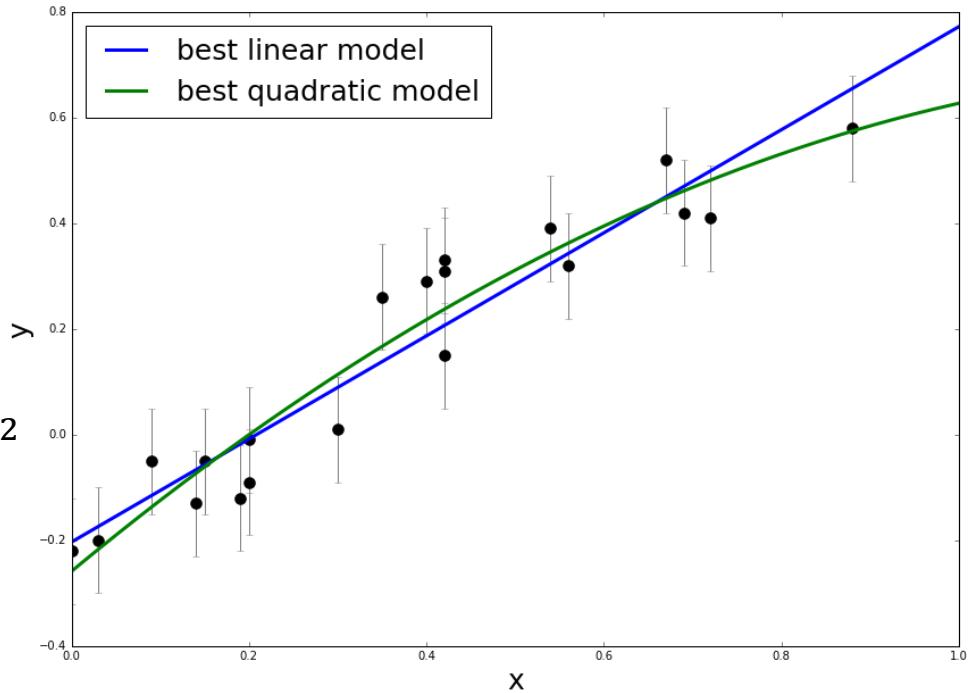
- **Model fitting** proceeds by assuming a particular model is true, and tuning the model so it provides the best possible fit to the data
- **Model selection**, on the other hand, asks the larger question of whether the assumptions of the model are compatible with the data.

Linear model:

$$y_{M1} = f_{M1}(x; w) = w_0 + w_1 x$$
$$y \sim N(y_{M1}, \sigma_y^2)$$

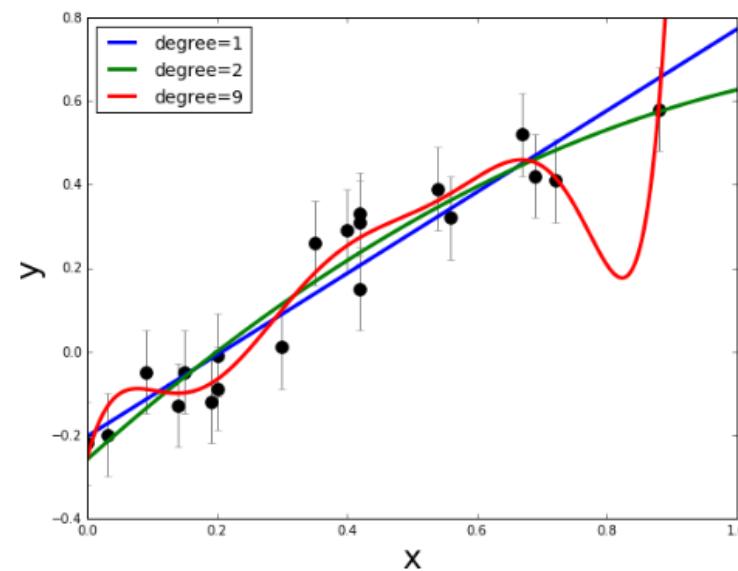
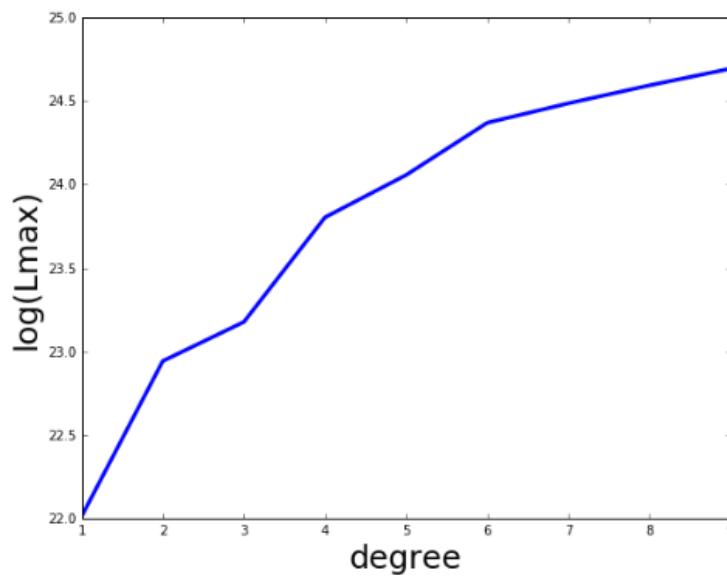
Quadratic model:

$$y_{M2} = f_{M2}(x; w) = w_0 + w_1 x + w_2 x^2$$
$$y \sim N(y_{M2}, \sigma_y^2)$$



Which model is better ?

## Model Complexity and Generality



- Comparing maximum likelihood  $p(D|w, M_1)$  and  $p(D|w, M_2)$  is not a good idea

$$\lim_w p(D|w, M_1) \quad v.s. \quad \lim_w p(D|w, M_2)$$

- As more complex model is used, model better fits the data, however, this model cannot predict well on unseen test data
- Balancing between model fitting and generalization is a fundamental question in ML
- In frequentist approach, a complex model is penalized by additional regularization term

The Bayesian approach addresses this by integrating over the model parameter space, which in effect acts to automatically penalize overly-complex models.

## Bayesian Approach for Model Selection

- The parameter posterior given the model  $M$  is expressed

$$p(w|D, M) = \frac{p(D|w, M)p(w|M)}{p(D|M)}$$

- The model posterior can be expressed

$$p(M|D) = \frac{p(D|M)p(M)}{p(D)}$$

where  $p(D|M) = \int_w p(D, w|M) = \int_{\Omega} p(D|w, M)p(w|M)d\theta$

(Integration over the entire parameter space  $w \in \Omega$ )

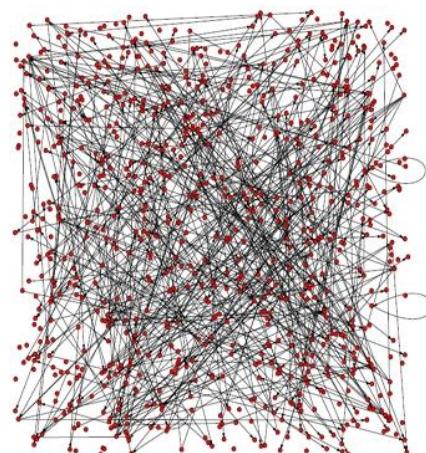
- The odd ratio between two models,  $M_1$  and  $M_2$ , can be expressed

$$O_{21} = \frac{p(M_2|D)}{p(M_1|D)} = \frac{p(D|M_2)}{p(D|M_1)} \frac{p(M_2)}{p(M_1)}$$

$O_{21} > \text{threshold}$   
Choose  $M_2$

$\frac{p(D|M_2)}{p(D|M_1)}$  : Bayes factors     $\frac{p(M_2)}{p(M_1)}$  : Prior odd ratio

## L7. Bayesian Network (Modeling)



Probability + Statistics + Graph Theory

## Degree of Belief and Probability

How to compare the plausibility of different statements?



$G$  : “we can be a billionaire if we go to graduate school”

vs

$S$  : “we can be a billionaire if we go to Samsung”



- If you believe  $G$  more than  $S$ , you can write  $G > S$
- If you believe  $S$  more than  $G$ , you can write  $G < S$
- If you have the same belief, you can write  $G \sim S$

Assumptions about relationships of  $>$  and  $\sim$

- *Universal comparability* : either  $G > S$ ,  $G < S$  or  $G \sim S$
- *Transitivity* : if  $G > S$  and  $S > V$ , then  $G > V$

Due to the two assumptions, the **degree of belief** can be represented by a real-valued function:

- $P(G) > P(S)$  if and only if  $G > S$
- $P(G) = P(S)$  if and only if  $G \sim S$

## Properties of probabilities for Bayesian Networks

We are going to use very simple probability theories to construct Probabilistic Graphical Model

- conditional probability :

$$P(A|B) = \frac{P(B|A)}{P(B)}$$

- Law of total probability :

$$P(A) = \sum_{B \in \mathcal{B}} P(A|B) P(B)$$

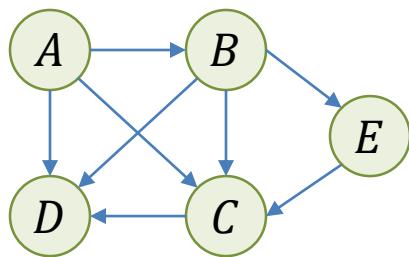
- Bayes' rule:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

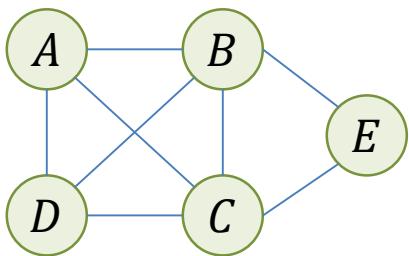
# Introduction to Graph Theory

## Graph

- A graph  $G$  consists of nodes (also called vertices) and edges (also called links) between the nodes.



A directed graph  $G$  consists of directed edges between nodes



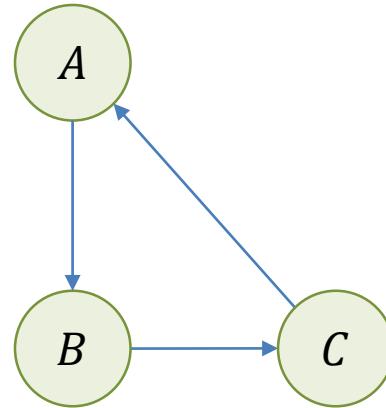
An undirected graph  $G$  consists of undirected edges between nodes

## Introduction to Graph Theory

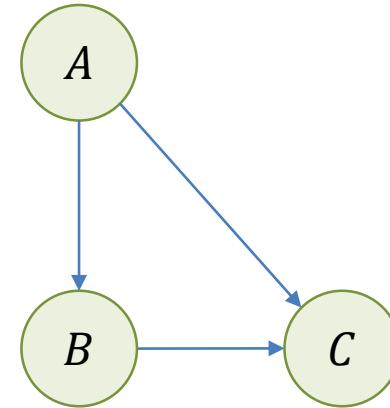
### Directed Acyclic Graph (DAG)

- A DAG is a graph  $G$  with directed edges (arrows on each link) between the nodes such that by following a path of nodes from one node to another along the direction of each edge no path will revisit a node.

Cyclic Graph



Acyclic Graph



- DAG will play a central role in modeling environments with many variables
  - will be used for the belief networks
  - can encode the direction dependence between the parent nodes and child nodes.

# Introduction to Graph Theory

## Path

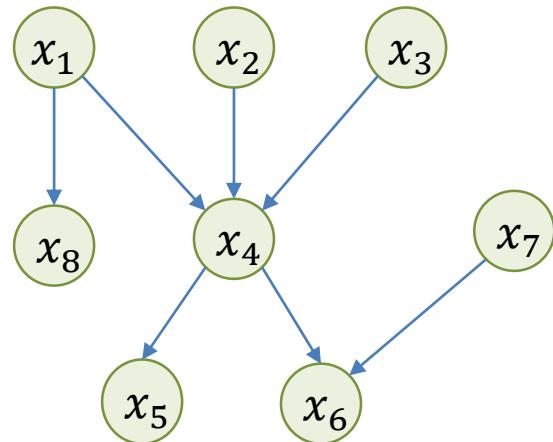
- A path  $A \rightarrow B$  from node  $A$  to node  $B$  is a sequence of nodes that connects  $A$  to  $B$

## Ancestors

- In directed graph, the nodes  $A$  such that  $A \rightarrow B$  and  $B \not\rightarrow A$  are the ancestors of  $B$

## Descendants

- In directed graph, the nodes  $B$  such that  $A \rightarrow B$  and  $B \not\rightarrow A$  are the descendants of  $A$



- ✓ A path  $x_1 \rightarrow x_6$  is  $x_1 \rightarrow x_4 \rightarrow x_6$
- ✓ The ancestors of  $x_6$  are  $\text{ac}(x_6) = \{x_1, x_2, x_3, x_4\}$
- ✓ The descendants of  $x_2$  are  $\text{dc}(x_2) = \{x_4, x_5, x_6\}$
- ✓ The parents of  $x_4$  are  $\text{pa}(x_4) = \{x_1, x_2, x_3\}$
- ✓ The children of  $x_4$  are  $\text{ch}(x_4) = \{x_5, x_6\}$

## Representations

- Edge list

$$L = \{(x_1, x_4), (x_2, x_4), (x_3, x_4), (x_1, x_8), (x_4, x_5), (x_4, x_6), (x_7, x_6)\}$$

- Adjacency matrix

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

## Motivation of Bayesian Network

### Full Joint Distribution

Example distribution			
$A$	$B$	$C$	$P(A, B, C)$
0	0	0	0.08
0	0	1	0.15
0	1	0	0.05
0	1	1	0.10
1	0	0	0.14
1	0	1	0.18
1	1	0	0.19
1	1	1	0.11

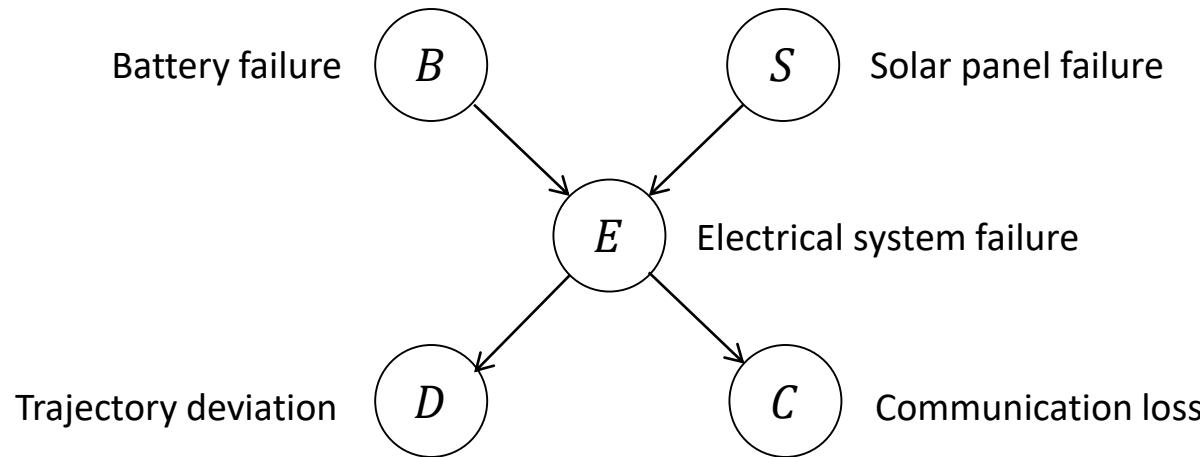
- Binary variables:  $A, B, C$  (e.g.,  $A = 1$  or 0)
- $2^3$  entities are required to construct the table
- $2^3 - 1$  independent parameters are required to fully specify the joint probability distribution
- $2^N - 1$  parameters are required for  $N$  binary variables
- If each variable has  $M$  different choices,  $M^N - (M - 1)$  parameters are required

The number of parameters grows exponentially

→ Difficult to represent Probability distribution and learn the parameters from data

## Motivation of Bayesian Network

### Full Joint Probability Distribution



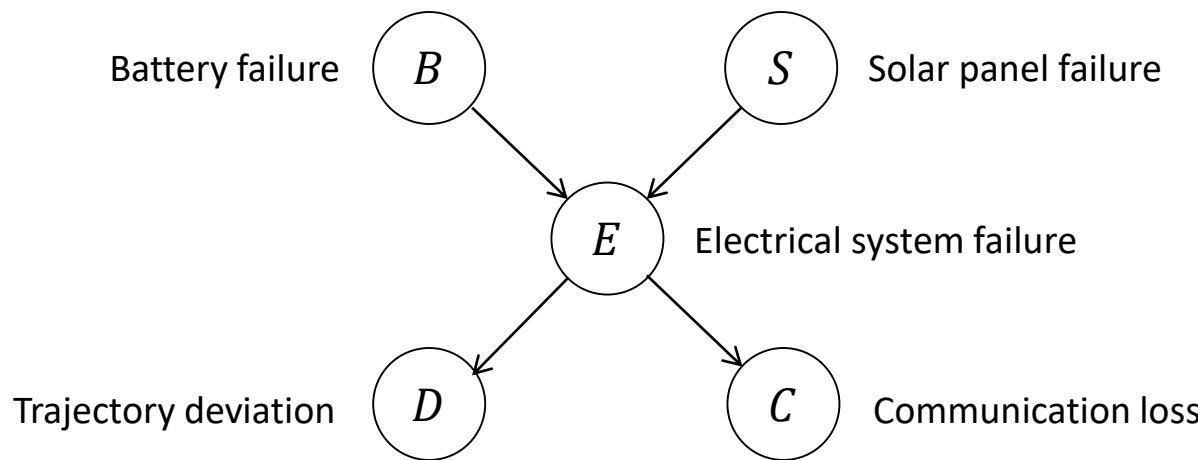
- Binary variables:  $B, S, E, D, C$  (e.g.,  $B = 1$  or  $0$ )
- $2^5$  entities are required to construct the table
- $2^5 - 1$  independent parameters are required to fully specify the joint probability distribution
- $2^N - 1$  parameters are required for  $N$  binary variables
- If each variable has  $M$  different choices,  
 $M^N - (M - 1)$  parameters are required

The number of parameters grows exponentially

→ Difficult to represent Probability distribution and learn the parameters from data

## Motivation of Bayesian Network

A Bayesian network is a compact representation of a joint distribution

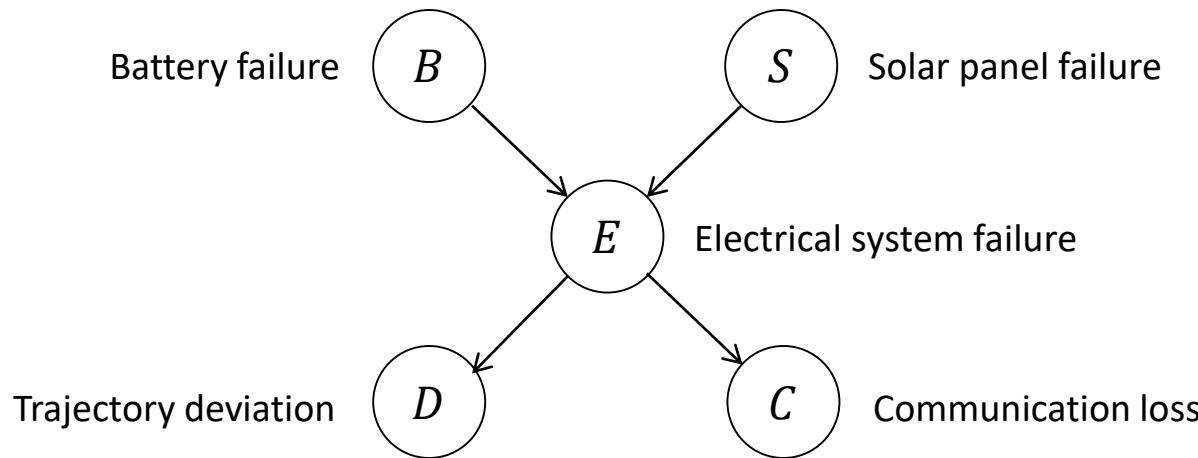


**Probability** + **Statistics** + **Graph Theory**

- A Bayesian Network introduces structure into a probabilistic model by using graphs to represent independence assumptions among the variables. For inferencing, it uses statistics
- Provide a good representation to model the probabilistic structures between random variables.
  - Nodes represent random variables
  - Edges represent probabilistic dependency, namely conditional probability among variables
- Conditional independence described by the graph, greatly reduce the computational effort to learn the model and inferencing random variables.

## Motivation of Bayesian Network

A Bayesian network is a compact representation of a joint distribution



- Each node corresponds to a random variable
- Directed edges connect pairs of nodes, indicating direct probabilistic relationships
- $P(x_i|\text{pa}_{x_i})$  represents the probability distribution of  $x_i$  conditional on the parent nodes  $\text{pa}_{x_i}$  of  $X_i$     e.g.,  $P(E|B,S)$  :  $B$  and  $S$  are the parent nodes of  $E$

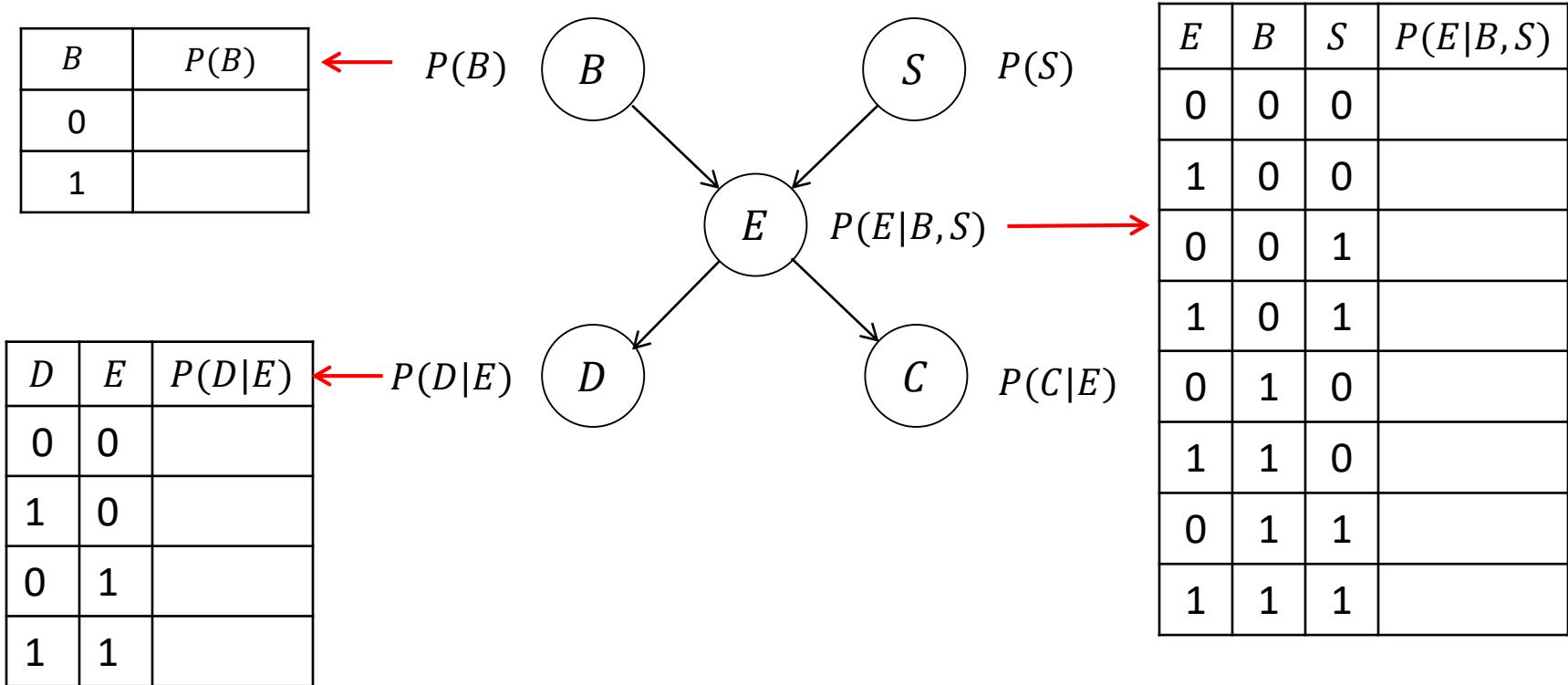
The chain rule for Bayesian networks specifies how to construct a joint distribution from the local conditional probability distribution

$$P(x_1, \dots, x_n) = \prod_{i=1}^n P(x_i | \text{pa}_{x_i})$$

local conditional probability distribution

## Motivation of Bayesian Network

A Bayesian network is a compact representation of a joint distribution



- Chain rule:  $P(B, S, E, D, C) = P(B)P(S)P(E|B, S)P(D|E)P(C|E)$
- Required independent parameters to fully specify the joint PDF  
 $P(B) : 1, P(S) : 1, P(E|B, S) : 4, P(D|E) : 2, P(C|E) : 2$  (total 10 compared to  $2^5 - 1 = 31$ )

Bayesian network can greatly reduce the number of parameters

## Formal Definition Bayesian Network

- A Bayesian network (BN) is a distribution of the form

$$p(x_1, \dots, x_n) = \prod_{i=1}^n p(x_i | \text{pa}_{x_i})$$

- ✓  $\text{pa}_{x_i}$  represents the parental variables of variable  $x_i$
- ✓ BN is represented as a directed acyclic graph with an arrow pointing from a parent variable to child variable
- Every probability distribution can be written as a BN:

$$\begin{aligned} p(x_1, \dots, x_n) &= p(x_n | x_1, \dots, x_{n-1}) p(x_1, \dots, x_{n-1}) \\ &= p(x_n | x_1, \dots, x_{n-1}) p(x_{n-1} | x_1, \dots, x_{n-2}) p(x_1, \dots, x_{n-2}) \\ &= p(x_1) \prod_{i=2}^n p(x_i | \text{pa}_{x_{i-1}}) \end{aligned}$$

- The particular role of BN is that the structure of the DAG corresponds to a set of **conditional independence assumptions**, namely which ancestral parental variables are sufficient to specify each conditional probability table

## Conditional Independence

### Definition : Independence

$$X \perp Y$$

$p(X, Y) = p(X)p(Y)$  for all states of  $X, Y$

or equivalently  $P(X|Y) = P(X)$

### Definition : Conditional Independence

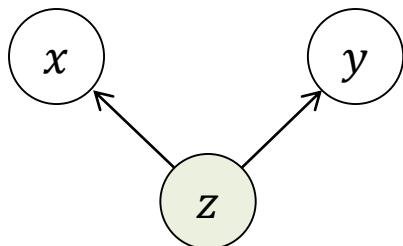
$$X \perp Y|Z$$

$p(X, Y|Z) = p(X|Z)p(Y|Z)$  for all states of  $X, Y, Z$

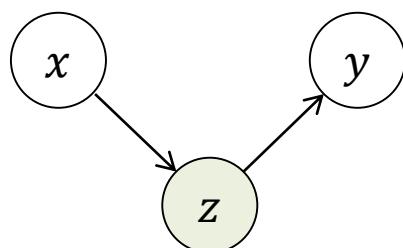
or equivalently  $P(X|Y, Z) = P(X|Z)$

- ✓ The two sets of variables  $X$  and  $Y$  are independent of each other provided we know the state of the set of variables  $Z$
- ✓ The information of  $Y$  does not give further information on  $X$

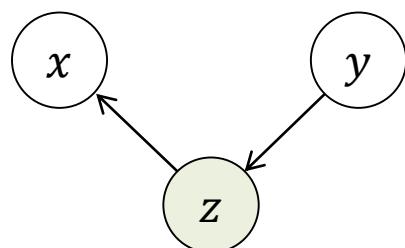
## V-structure (or collider)



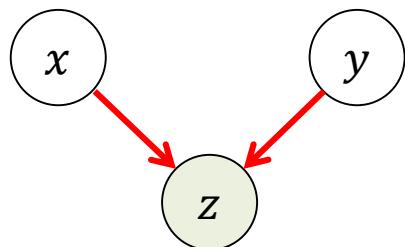
$$p(x, y|z) = \frac{p(x, y, z)}{p(z)} = \frac{p(z)p(x|z)p(y|z)}{p(z)} = p(x|z)p(y|z)$$



$$\begin{aligned} p(x, y|z) &= \frac{p(x, y, z)}{p(z)} = \frac{p(x)p(z|x)p(y|z)}{p(z)} \\ &= \frac{p(x, z)p(y|z)}{p(z)} = p(x|z)p(y|z) \end{aligned}$$



$$\begin{aligned} p(x, y|z) &= \frac{p(x, y, z)}{p(z)} = \frac{p(y)p(z|y)p(x|z)}{p(z)} \\ &= \frac{p(y, z)p(x|z)}{p(z)} = p(y|z)p(x|z) \end{aligned}$$



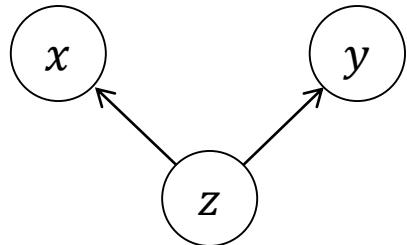
$$p(x, y|z) = \frac{p(x, y, z)}{p(z)} = \frac{p(x)p(y)\cancel{p(z|x, y)}}{p(z)} \neq p(y|z)p(x|z)$$

BN with  $x \rightarrow z \leftarrow y$

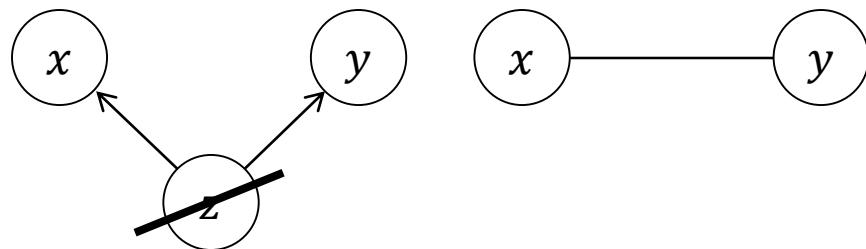
- ✓  $x$  and  $y$  are unconditionally independent
- ✓  $x$  and  $y$  are dependent conditional on  $z$

## V-structure (or collider)

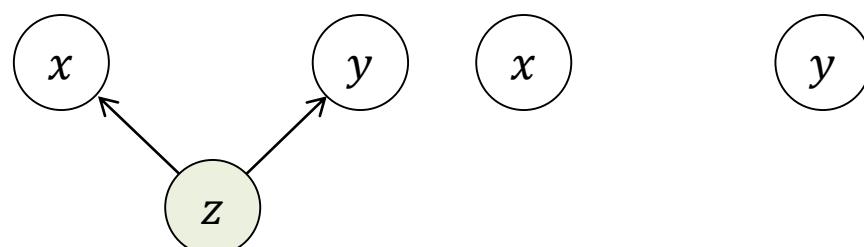
$$p(x, y, z) = p(x|z)p(y|z)$$



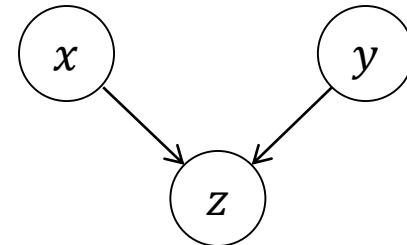
Marginalization over z



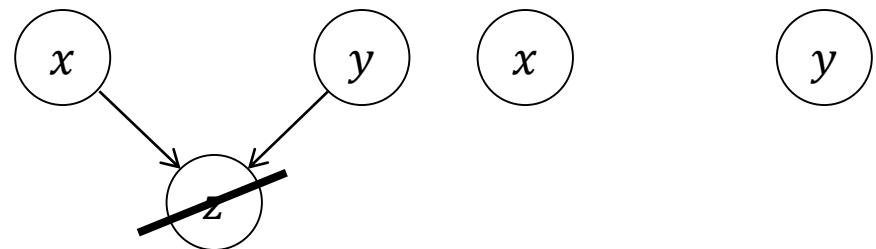
Conditionalization on z



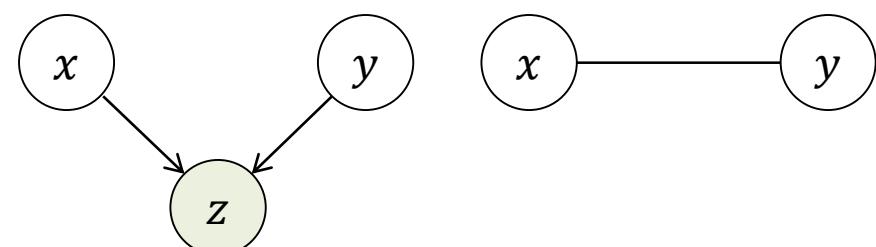
$$p(x, y, z) = p(z|x, y)p(x)p(y)$$



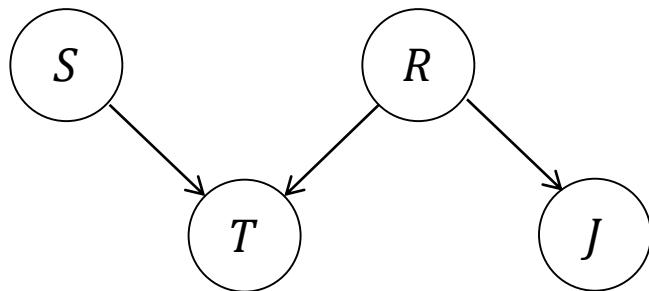
Marginalization over z



Conditionalization on z



## Example : Wet Grass



$R \in \{0,1\}$  :  $R = 1$  means that it has been raining  
 $S \in \{0,1\}$  :  $S = 1$  Sprinkler is turned on  
 $J \in \{0,1\}$  :  $J = 1$  Jack's grass is wet  
 $T \in \{0,1\}$  :  $T = 1$  Tracey's grass is wet

*Joint distribution based on chain rule*

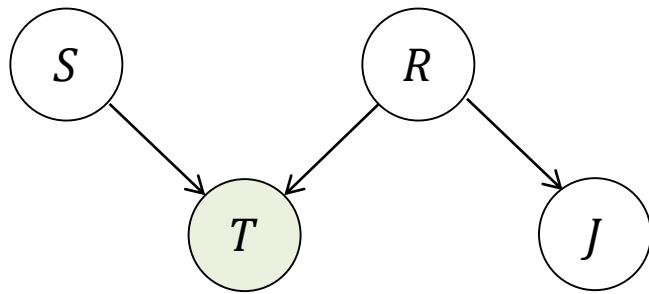
$$\begin{aligned} p(T, J, R, S) &= p(T|J, R, S)p(J, R, S) \\ &= p(T|J, R, S)p(J|R, S)p(R, S) \\ &= p(T|J, R, S)p(J|R, S)p(R|S)p(S) \\ &\quad 8 + \quad 4 + \quad 2 + 1 = 2^4 - 1 = 15 \text{ parameters are required} \end{aligned}$$

*Joint distribution conditional independence*

$$\begin{aligned} p(T, J, R, S) &= \color{blue}{p(T|J, R, S)} \color{red}{p(J|R, S)} \color{green}{p(R|S)} p(S) \\ &= \color{blue}{p(T|R, S)} \times \color{red}{p(J|R)} \times \color{green}{p(R)} \times p(S) \\ &= \color{blue}{p(T|R, S)} \color{red}{p(J|R)} \color{green}{p(R)} p(S) \end{aligned}$$

## Example : Wet Grass

### Modeling



$R \in \{0,1\}$  :  $R = 1$  means that it has been raining  
 $S \in \{0,1\}$  :  $S = 1$  Sprinkler is turned on  
 $J \in \{0,1\}$  :  $J = 1$  Jack's grass is wet  
 $T \in \{0,1\}$  :  $T = 1$  Tracey's grass is wet

$$p(T, J, R, S) = p(T|R, S)p(J|R)p(R)p(S)$$

$$p(T|S, E)$$

Tracey's Grass wet=1	Rain	Sprinkler
1	1	1
1	1	0
0.9	0	1
0	0	0

$$p(J|R)$$

Jack's Grass wet=1	Rain
1	1
0.2	0

$$p(S = 1) = 0.1$$

$$p(R = 1) = 0.2$$

The tables and graphical structure fully specify the distribution

## Example : Wet Grass

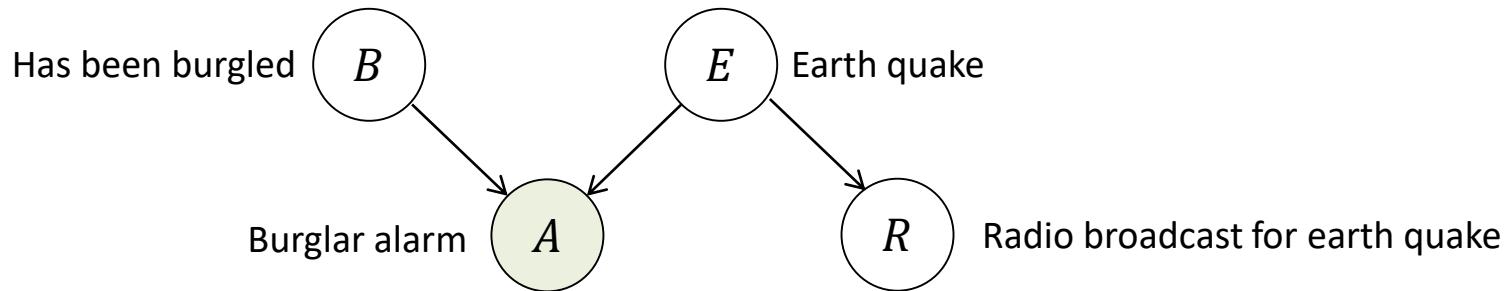
### Inference

$$\begin{aligned} p(S = 1|T = 1) &= \frac{p(S = 1, T = 1)}{p(T = 1)} = \frac{\sum_{J,R} p(T = 1, J, R, S = 1)}{\sum_{J,R,S} p(T = 1, J, R, S)} \\ &= \frac{\sum_{J,R} p(J|R)p(T = 1|R, S = 1)p(R)p(S = 1)}{\sum_{J,R,S} p(J|R)p(T = 1|R, S)p(R)p(S)} \\ &= \frac{\sum_R p(T = 1|R, S = 1)p(R)p(S = 1)}{\sum_{R,S} p(T = 1|R, S)p(R)p(S)} \quad \because \sum_J p(J|R) = 1 \\ &= \frac{0.9 \times 0.8 \times 0.1 + 1 \times 0.2 \times 0.1}{0.9 \times 0.8 \times 0.1 + 1 \times 0.2 \times 0.1 + 0 \times 0.8 \times 0.9 + 1 \times 0.2 \times 0.9} = 0.3382 \end{aligned}$$

$$\begin{aligned} p(S = 1|T = 1, J = 1) &= \frac{p(S = 1, T = 1, J = 1)}{p(T = 1, J = 1)} \\ &= \frac{\sum_R p(T = 1, J = 1, R, S = 1)}{\sum_{R,S} p(T = 1, J = 1, R, S)} \\ &= \frac{\sum_R p(J = 1|R)p(T = 1|R, S = 1)p(R)p(S = 1)}{\sum_{R,S} p(J = 1|R)p(T = 1|R, S)p(R)p(S)} \\ &= \frac{0.0344}{0.2144} = 0.1604 \end{aligned}$$

The fact that Jack's grass is also wet increases the chance that the rain has played a role in making Tracey's grass wet

## Example : Burglar Alarm



$$p(B, E, A, R) = p(A|B, E)p(R|E)p(E)p(B)$$

$$p(A|B, E)$$

Alarm=1	Burglar	Earthquake
0.9999	1	1
0.99	1	0
0.99	0	1
0.0001	0	0

$$p(R|E)$$

Radio=1	Earthquake
1	1
0	0

$$p(E = 1) = 0.01$$

$$p(E = 1) = 0.000001$$

$$\begin{aligned}
 p(B = 1|A = 1) &= \frac{p(B, A = 1)}{p(A = 1)} = \frac{\sum_{E,R} p(B = 1, E, A = 1, R)}{\sum_{B,E,R} p(B, E, A = 1, R)} \\
 &= \frac{\sum_{E,R} p(A = 1|B = 1, E)p(R|E)p(E)p(B = 1)}{\sum_{B,E,R} p(A = 1|B, E)p(R|E)p(E)p(B)} \approx 0.99
 \end{aligned}$$

$$p(B = 1|A = 1, R = 1) \approx 0.01$$

## Conditional Independence

Where causes the number of parameters to be reduced?

→ The conditional independence assumptions encoded by the structure of a Bayesian network

- $X$  and  $Y$  are independent if and only if

$$P(X, Y) = P(X)P(Y)$$

or equivalently  $P(X|Y) = P(X)$

$$\therefore P(X|Y) = \frac{P(X, Y)}{P(Y)} = \frac{P(X)P(Y)}{P(Y)} = P(X)$$

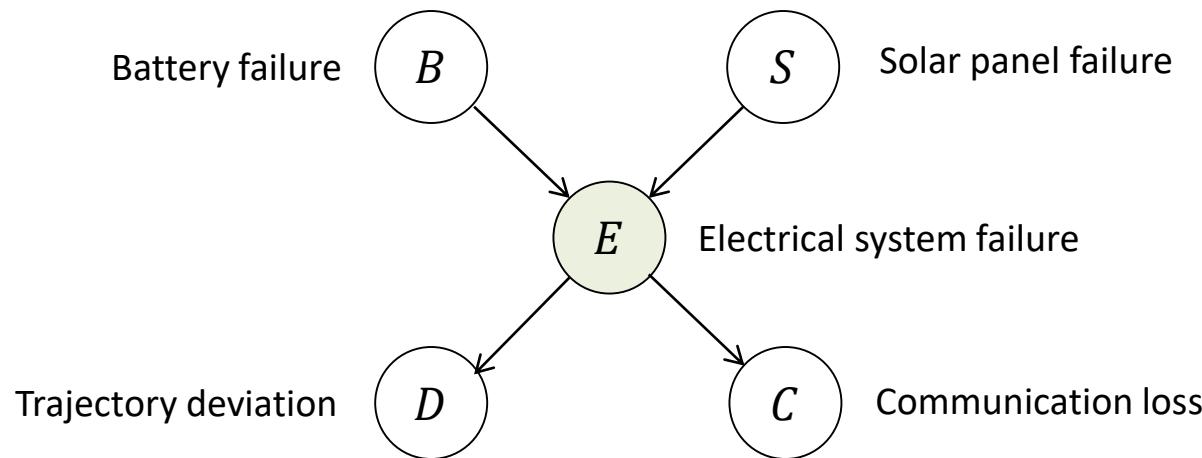
- $X$  and  $Y$  are conditionally independent given  $Z$  if and only if

$$P(X, Y|Z) = P(X|Z)P(Y|Z)$$

or equivalently  $P(X|Z) = P(X|Y, Z)$

Independence assumptions reduce the number of parameters used to represent a joint pdf

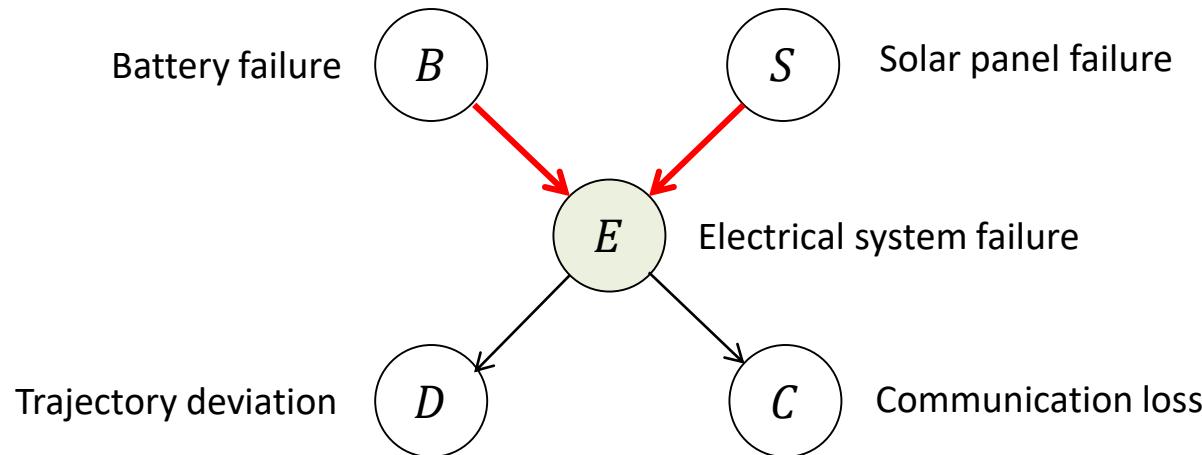
## Conditional Independence examples



- $C$  is independent of  $B$  given  $E$  :  $(C \perp B|E)$   
→ Information about Battery failure does not affect my belief on communication loss if I already know (observed) the status of electrical system failure
- $D$  is independent of  $S$  given  $E$  :  $(D \perp S|E)$   
→ Information about Solar failure does not affect my belief on a trajectory deviation if I already know (observed) the status of electrical system failure

## Conditional Independence examples

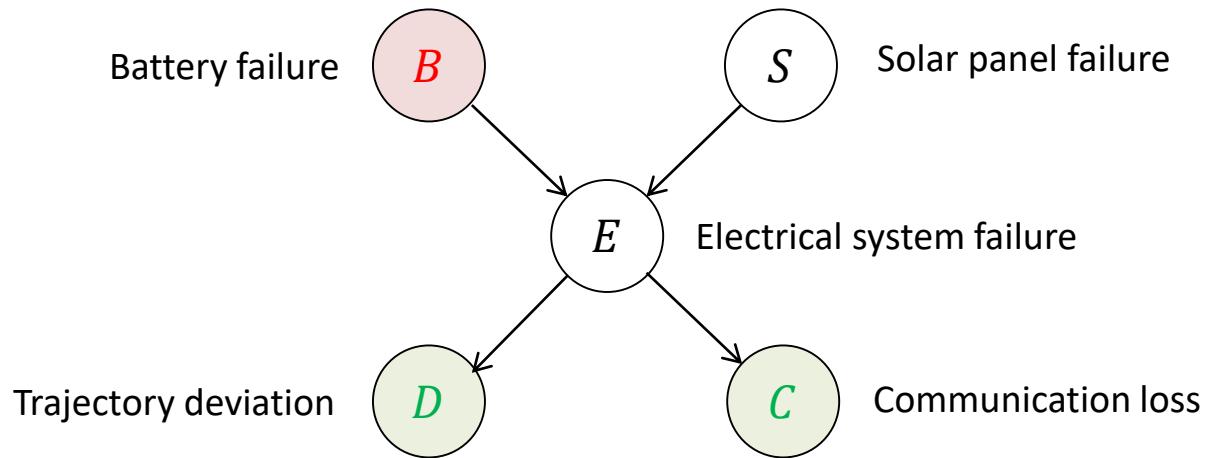
### V-structure



- $B$  is independent  $S$  ( $E$  is not observed)  
→ Knowing there is a battery failure does not affect my belief regarding solar panel failure
- $B$  is dependent  $S$  **given  $E$**   
→ If there was an electrical system failure (observed) and there was no battery failure, there it is likely that a solar panel fails
- Influence flows only through  $B \rightarrow E \leftarrow S$  when  $E$  is known

## Inference

Once a joint probability distribution is constructed, inference can be performed to determine the distribution over one or more unobserved variables given the values associated with a set of observed variables

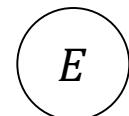


$P(B|d^1, c^1)$  Probability distribution of Battery failure

**Query variable**

given the trajectory deviation and the communication loss

**Evidence variable**



: Hidden variables

## Inference

How to compute  $P(B|d^1, c^1)$ ?

### Exact inference

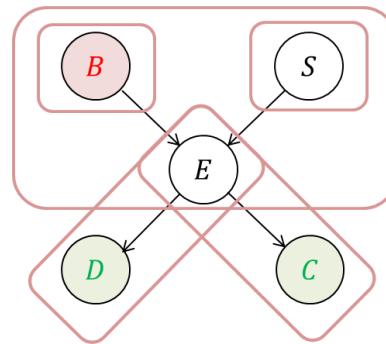
$$\begin{aligned} P(b^1|d^1, c^1) &\propto \sum_s \sum_e P(b^1, s, e, d^1, c^1) \\ &= \sum_s \sum_e P(b^1)P(s)P(e|b^1, s)P(d^1|e)p(c^1|e) \quad \text{By conditional independence} \\ &= P(b^1) \sum_e P(d^1|e)p(c^1|e) \sum_s P(s)P(e|b^1, s) \end{aligned}$$

The number of terms to be added together can grow exponentially with the number of hidden variables

## Inference

How to compute  $P(B|d^1, c^1)$ ?

### Variable Elimination



Conditional distributions are represented by the following tables

$$T_1(B)T_2(S)T_3(E, B, S)T_4(d^1, E)T_5(c^1, E)$$

$$T_1(B)T_2(S)T_3(E, B, S)T_6(E)T_7(E)$$

Observe evidence ( $d^1$  and  $c^1$ )

$$T_1(B)T_2(S)T_8(B, S)$$

$$T_8(B, S) = \sum_e T_3(e, B, S)T_6(e)T_7(e)$$

$$T_1(B)T_9(B)$$

$$T_9(B) = \sum_s T_2(s)T_8(B, s)$$

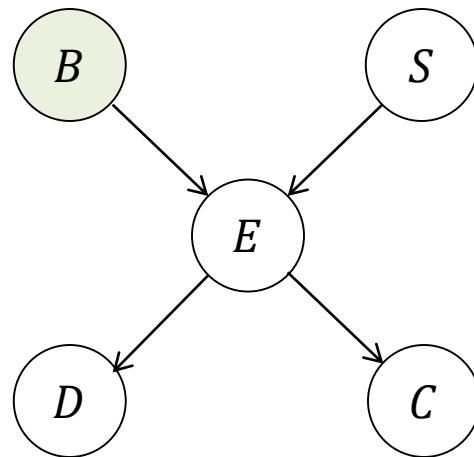
Normalizing the product of the two factors ( $T_1(B)$  and  $T_9(B)$ ) results in  $P(B|d^1, c^1)$

Variable elimination algorithm relies on [heuristic ordering](#) of variables to eliminate in sequence  
→ Often linear but sometimes exponential

## Inference

How to compute  $P(B|d^1, c^1)$ ?

Approximate inference (Sampling based methods)



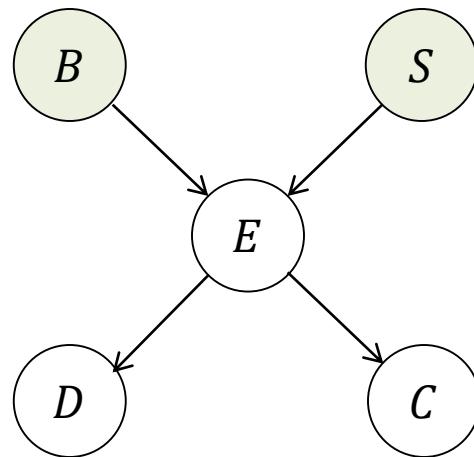
B	S	E	D	C
1				

Sample from  $P(B)$

## Inference

How to compute  $P(B|d^1, c^1)$ ?

Approximate inference (Sampling based methods)



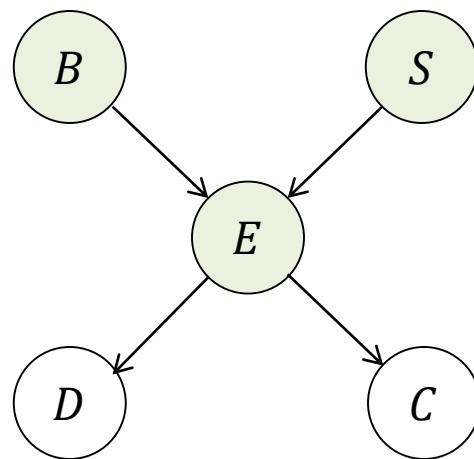
B	S	E	D	C
1	1			

Sample from  $P(S)$

## Inference

How to compute  $P(B|d^1, c^1)$ ?

Approximate inference (Sampling based methods)



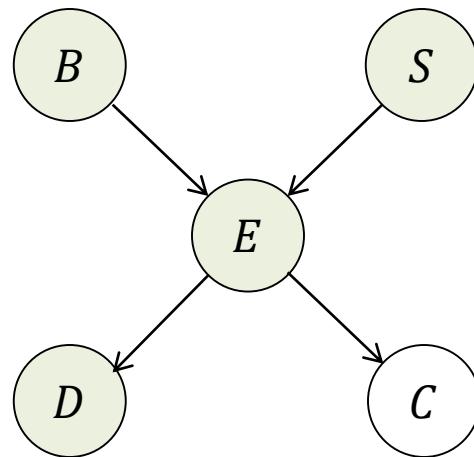
B	S	E	D	C
1	1	1		

Sample from  $P(E|B = 1, S = 1)$

## Inference

How to compute  $P(B|d^1, c^1)$ ?

Approximate inference (Sampling based methods)



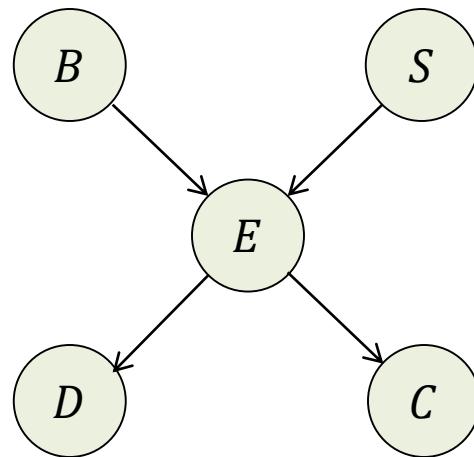
B	S	E	D	C
1	1	1	0	

Sample from  $P(D|E = 1)$

## Inference

How to compute  $P(B|d^1, c^1)$ ?

Approximate inference (Sampling based methods)



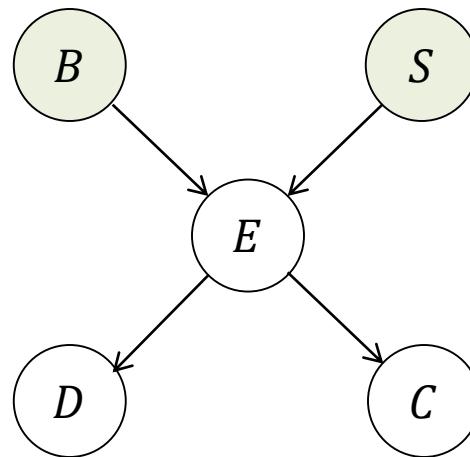
B	S	E	D	C
1	1	1	0	0

Sample from  $P(C|E = 1)$

## Inference

How to compute  $P(B|d^1, c^1)$ ?

Approximate inference (Sampling based methods)



$$P(b^1|d^1, c^1) = 1/3$$

$$P(b^0|d^1, c^1) = 2/3$$

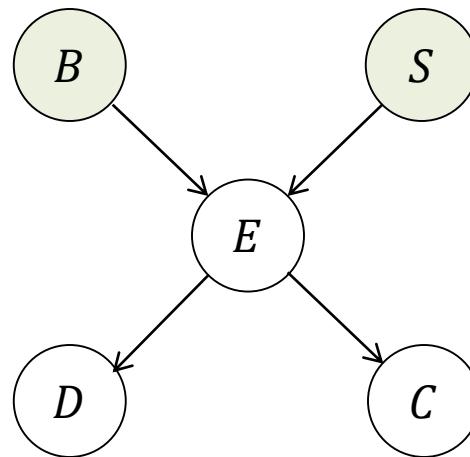
B	S	E	D	C
1	1	1	0	0
0	1	0	1	0
1	0	1	1	1
0	1	0	0	1
0	1	1	1	1
0	1	0	0	1
0	0	0	1	0
0	1	1	1	0
0	1	0	1	1

Three cases coincide observations  $d^1, c^1$

## Inference

How to compute  $P(B|d^1, c^1)$ ?

Approximate inference (Sampling based methods)



$$P(b^1|d^1, c^1) = 1/3$$

$$P(b^0|d^1, c^1) = 2/3$$

B	S	E	D	C
1	1	1	0	0
0	1	0	1	0
1	0	1	1	1
0	1	0	0	1
0	1	1	1	1
0	1	0	0	1
0	0	0	1	0
0	1	1	1	0
0	1	0	1	1

Three cases coincide observations  $d^1, c^1$

If likelihood of evidence is small, then many samples are required!!

## Inference

How to compute  $P(B|d^1, c^1)$ ?

### Likelihood sampling

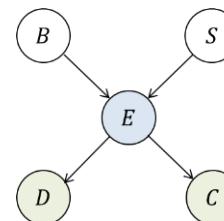
B	S	E	D	C
1	1	1	0	0
0	1	0	1	0
1	0	1	1	1
0	1	0	0	1
0	1	1	1	1
0	1	0	0	1
0	0	0	1	0
0	1	1	1	0
0	1	0	1	1

Algorithm 2.5 Likelihood-weighted sampling from a Bayesian network

```

1: function LIKELIHOODWEIGHTEDSAMPLE( $B, o_{1:n}$ )
2:    $X_{1:n} \leftarrow$  a topological sort of nodes in  $B$ 
3:    $w \leftarrow 1$ 
4:   for  $i \leftarrow 1$  to  $n$ 
5:     if  $o_i = \text{NIL}$ 
6:        $x_i \leftarrow$  a random sample from  $P(X_i | \text{pa}_{x_i})$ 
7:     else
8:        $x_i \leftarrow o_i$ 
9:        $w \leftarrow w \times P(x_i | \text{pa}_{x_i})$ 
10:  return  $(x_{1:n}, w)$ 

```



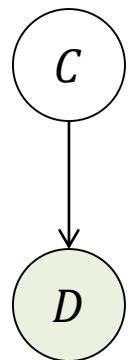
B	S	E	D	C	weight
1	0	1	1	1	$P(d^1 e^1) P(c^1 e^1)$
0	1	1	1	1	$P(d^1 e^1) P(c^1 e^1)$
0	1	0	1	1	$P(d^1 e^0) P(c^1 e^0)$

$$P(b^1|d^1, c^1) = \frac{P(d^1|e^1) P(c^1|e^1)}{P(d^1|e^1) P(c^1|e^1) + P(d^1|e^1) P(c^1|e^1) + P(d^1|e^0) P(c^1|e^0)}$$

## Inference

How to compute  $P(B|d^1, c^1)$ ?

**Likelihood sampling has a still problem!**



$$P(c^1) = 0.001$$

$$P(d^1|c^0) = 0.001$$

$$P(d^1|c^1) = 0.999$$

**Bayesian approach :**

$$\begin{aligned} P(c^1|d^1) &= \frac{P(d^1|c^1)P(c^1)}{P(d^1|c^1)P(c^1) + P(d^1|c^0)P(c^0)} \\ &= \frac{0.999 \times 0.001}{0.999 \times 0.001 + 0.001 \times 0.999} \\ &= 0.5 \end{aligned}$$

**To use likelihood weighting sampling approach:**

$$c^0, c^0, \dots, c^1$$

$P(d^1|c^1) = 0$  because  $c^1$  is not sampled due to the low prior

## Inference

How to compute  $P(B|d^1, c^1)$ ?

### Gibbs sampling, a kind of Markov chain Monte Carlo technique

- The sequence of samples forms a Markov chain
- In the limit, samples are drawn exactly from the joint distribution over the unobserved variables given the observations
- Simulate samples by sweeping through all the posterior conditionals, one random variables at a time

---

Algorithm : Gibbs sampler

---

Initialize  $X^{(0)} \sim q(x)$

**for** iteration  $i = 1, \dots$  **do**

$$x_1^{(i)} \sim P(X_1 = x_1 | X_2 = x_2^{(i-1)}, X_3 = x_3^{(i-1)}, \dots, X_D = x_D^{(i-1)})$$

$$x_2^{(i)} \sim P(X_2 = x_2 | X_1 = x_1^{(i)}, X_3 = x_3^{(i-1)}, \dots, X_D = x_D^{(i-1)})$$

$$x_3^{(i)} \sim P(X_3 = x_3 | X_1 = x_1^{(i)}, X_2 = x_2^{(i)}, \dots, X_D = x_D^{(i-1)})$$

⋮

$$x_D^{(i)} \sim P(X_D = x_D | X_1 = x_1^{(i)}, X_2 = x_2^{(i)}, \dots, X_{D-1} = x_{D-1}^{(i)})$$

**end for**

---

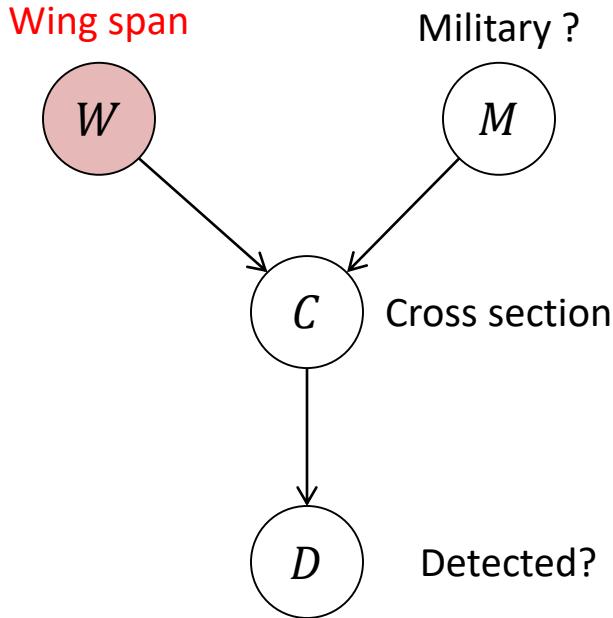
Because samples from the early iterations are not from the target posterior, it is common to discard these samples “burn-in” period”

### Sampling method comparisons

Jupyter Demo Simulation  
Wet grass (PyMC)

## Hybrid Bayesian Networks

Bayesian networks can contain a mixture of both discrete and continuous variables

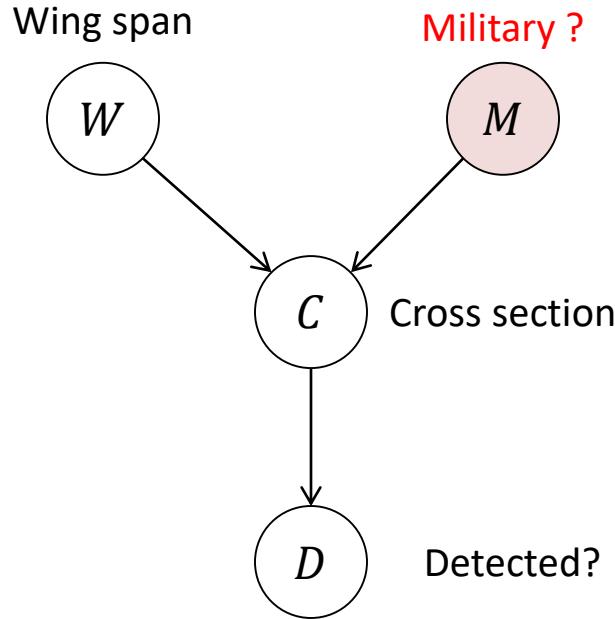


Wing span is a continuous variable and modeled as a Gaussian distribution

$$P(w) = N(w|\mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{w-\mu}{\sigma}\right)^2}$$

## Hybrid Bayesian Networks

Bayesian networks can contain a mixture of both discrete and continuous variables



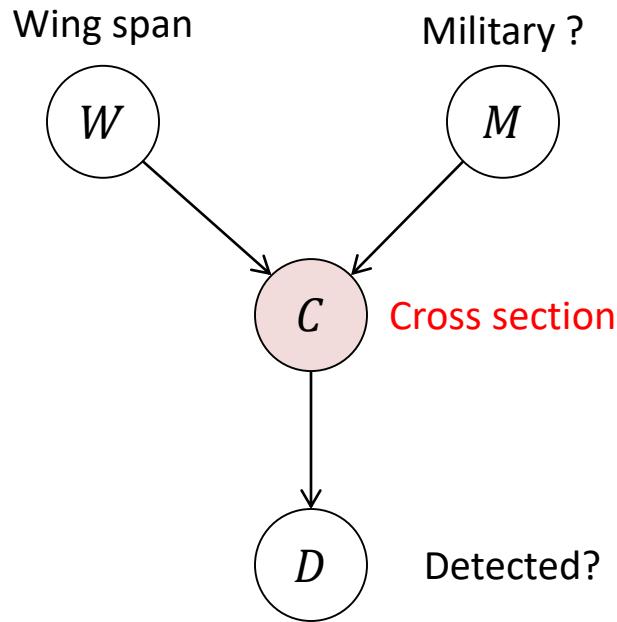
Whether a target is a military vehicle can be modeled with a single parameter  $\theta$

$$P(m^1) = \theta$$

$$P(m^0) = 1 - \theta$$

## Hybrid Bayesian Networks

Bayesian networks can contain a mixture of both discrete and continuous variables



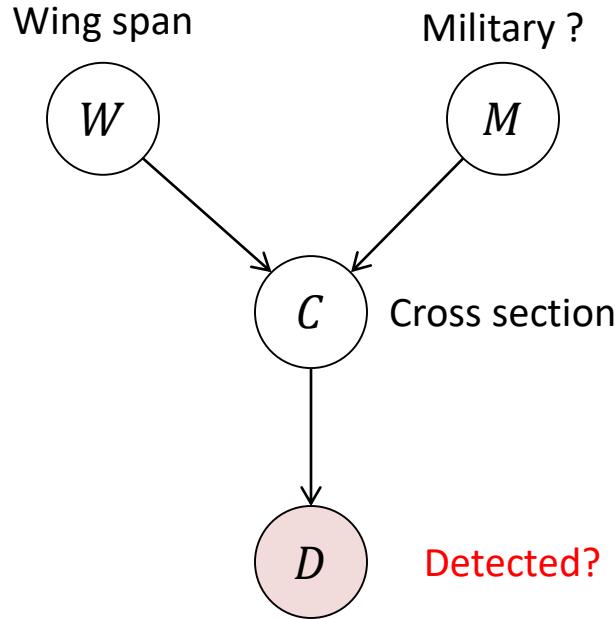
Radar cross section can be modeled as a conditional Gaussian

$$P(c|w, m) = \begin{cases} N(c|a_0 w + b_0, \sigma_0^2) & \text{if } m = m^0 \\ N(c|a_1 w + b_1, \sigma_1^2) & \text{if } m = m^1 \end{cases}$$

(Conditional linear Gaussian)

## Hybrid Bayesian Networks

Bayesian networks can contain a mixture of both discrete and continuous variables

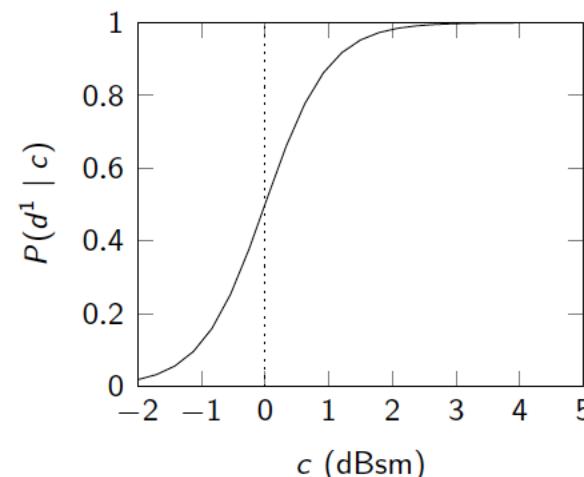


- Logit model:

$$P(d^1|c) = \frac{1}{1 + \exp\left(-2\frac{c - \alpha}{\beta}\right)}$$

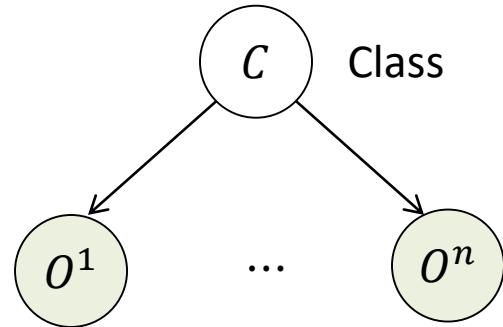
- Probit model:

$$P(d^1|c) = \Phi\left(\frac{c - \alpha}{\beta}\right)$$



## Bayesian Network for Classification

### Naïve Bayes Model



Prior:  $P(C)$

Class conditional distribution  $P(O^i|C)$

$$P(C|O^{1:n}) = \frac{P(C, O^{1:n})}{P(O^{1:n})} = \frac{P(C) \prod_{i=1}^n P(O^i|C)}{P(O^{1:n})}$$

$$P(O^{1:n}) = \sum_c P(c, O^{1:n})$$

$$P(C|O^{1:n}) \propto P(C) \prod_{i=1}^n P(O^i|C)$$

## Parameter Learning

We already know how to estimate the parameters for probability distributions

MLE or Bayesian approach

## Structure learning

- Bayesian Score  $P(G|D)$  for a certain graph  $G$  given data  $D$  is defined as

$$\begin{aligned} P(G|D) &= \frac{P(G)P(D|G)}{P(D)} \\ &= \frac{P(G) \int_{\theta} P(D|\theta, G)P(\theta|G)d\theta}{P(D)} \end{aligned}$$

- A Bayesian approach to structure learning involves finding the graph  $G$  that maximizes the Bayesian Score  $P(G|D)$  as

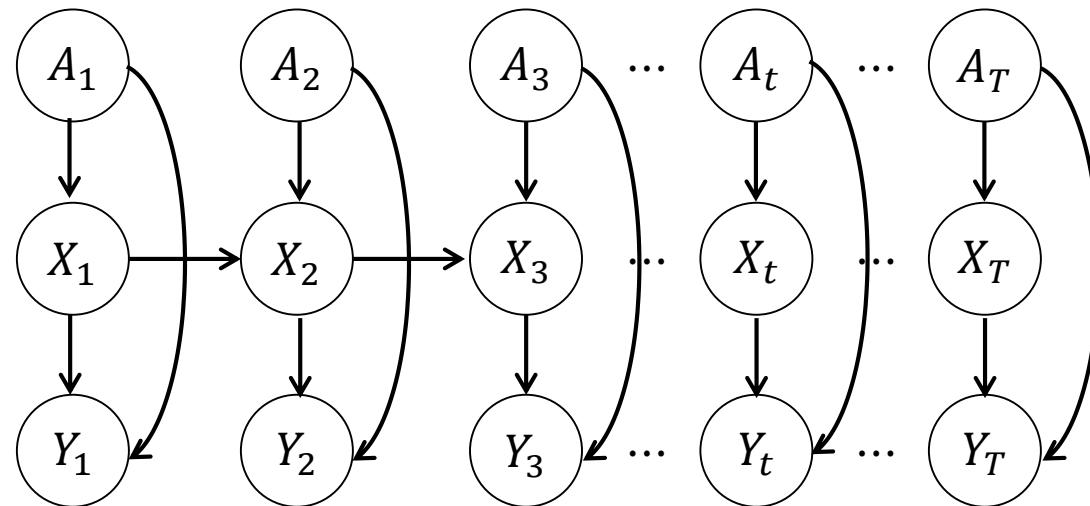
$$G^* = \operatorname{argmax}_G P(G|D)$$

- Not feasible to enumerate every possible structure, so use local search for graph with largest Bayesian score

## **Reference**

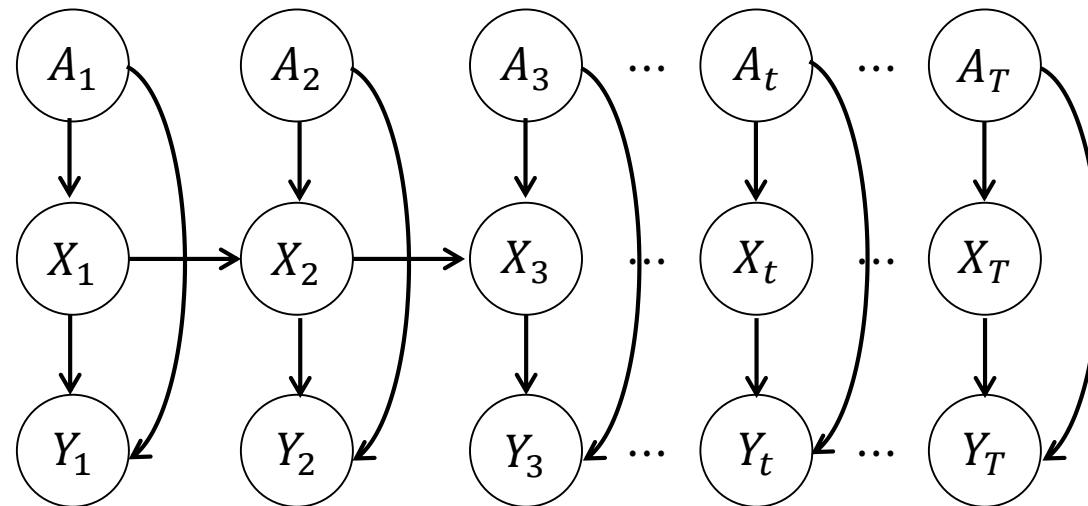
## L8. Dynamic Bayesian Network

Dynamic Bayesian Network relates variables to each other **over adjacent time steps**.



## Dynamic Bayesian Networks (temporal model)

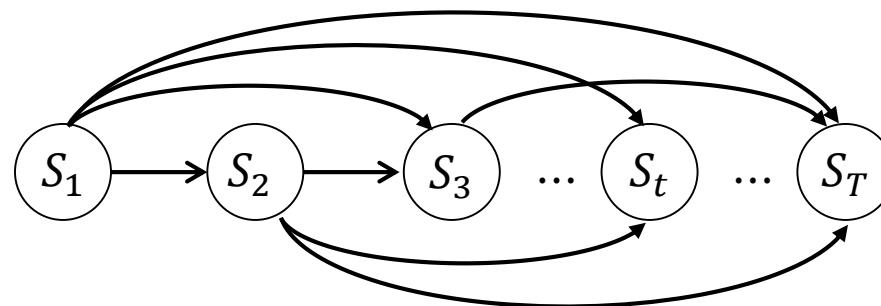
- We are interested in reasoning about the state of the world as it evolves over time
- **System state**  $S_t$  is a snapshot of the relevant attributes of the system at time  $t$
- Trajectory of states  $S_1, \dots, S_t$  represents the evolution of the target system
- $P(S_1, \dots, S_t)$  is very complex probability space
  - we need a series of simplifying assumptions



## Discrete-State Markov models

Consider a distribution over trajectories sampled over a prefix of time  $t = 1, \dots, T$

$$P(S_1, S_2, \dots, S_T) = P(S_1) \prod_{t=2}^T P(S_t | S_{1:t-1}) \quad \text{Cascade decomposition}$$



## Discrete-State Markov models

Consider a distribution over trajectories sampled over a prefix of time  $t = 1, \dots, T$

$$P(S_1, S_2, \dots, S_T) = P(S_1) \prod_{t=2}^T P(S_t | S_{1:t-1}) \quad \text{Cascade decomposition}$$

### 1. Markov Chain:

A Markov chain is defined on either discrete or continuous variables  $S_{1:t}$  is one in which the following **conditional independence** assumption holds:

$$P(S_t | S_1, \dots, S_{t-1}) = P(S_t | S_{t-L}, \dots, S_{t-1})$$

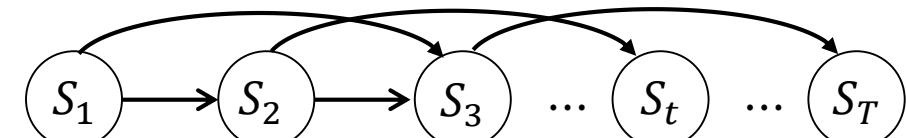
where  $L \geq$  is the order of the Markov chain. First order Markov chain can be represented

$$P(S_1, S_2, \dots, S_T) = P(S_1) \prod_{t=2}^T P(S_t | S_{t-1})$$

the future is conditionally independent of the past given the present:  $(S^{t+1} \perp S^{(1:t-1)} | S^t)$



First Order Markov Chain



Second Order Markov Chain

## Discrete-State Markov models

Consider a distribution over trajectories sampled over a prefix of time  $t = 1, \dots, T$

$$P(S_1, S_2, \dots, S_T) = P(S_1) \prod_{t=2}^T P(S_t | S_{1:t-1}) \quad \text{Cascade decomposition}$$

### 1. Markov Chain:

A Markov chain is defined on either discrete or continuous variables  $S_{1:t}$  is one in which the following **conditional independence** assumption holds:

$$P(S_t | S_1, \dots, S_{t-1}) = P(S_t | S_{t-L}, \dots, S_{t-1})$$

where  $L \geq$  is the order of the Markov chain. First order Markov chain can be represented

$$P(S_1, S_2, \dots, S_T) = P(S_0) \prod_{t=1}^T P(S_t | S_{t-1})$$

### 2. Stationary assumption

The state transition probability  $P(S^{t+1}|S^t)$  is the same for all  $t$

$$P(S^{t+1} = s' | S^t = s) = P(S' = s' | S = s) \text{ for any } t$$

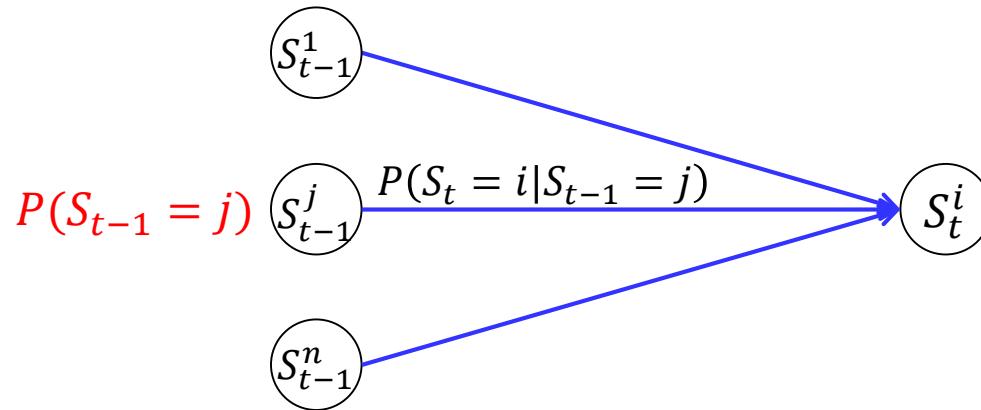
→The number of parameters are reduced substantially

## Equilibrium and stationary distribution of a Markov chain

- The marginal  $P(S_t)$  evolves through time. For discrete time,

$$P(S_t = i) = \sum_j P(S_t = i | S_{t-1} = j) P(S_{t-1} = j)$$

- ✓  $P(S_t = i)$ : the frequency that we visit state  $i$  at time  $t$ , given we started with a sample from  $P(S_1)$  and subsequently repeatedly drew samples from the transition  $P(S_\tau | S_{\tau-1})$



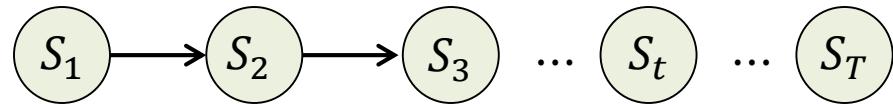
- Denoting  $(\mathbf{p}_t)_i = P(S_t = i)$ ,

$$\mathbf{p}_t = \mathbf{M}\mathbf{p}_{t-1} = \mathbf{M}^{t-1}\mathbf{p}_1$$

- If, for  $t \rightarrow \infty$ ,  $\mathbf{p}_t$  is independent of the initial distribution  $\mathbf{p}_1$ , then  $\mathbf{p}_\infty$  is called the equilibrium distribution of the chain, that is

$$\mathbf{p}_\infty = \mathbf{M}\mathbf{p}_\infty$$

## Fitting Markov Models



Given a sequence ( $S_1 = s_1, S_2 = s_2, \dots, S_T = s_T$ ), how to construct the transition matrix?

$$\theta_{i|j} = P(S_\tau = i | S_{\tau-1} = j) \propto \sum_{t=2}^T \mathbb{I}[S_\tau = i, S_{\tau-1} = j]$$

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_{1|1} \theta_{1|2} \theta_{1|3} \theta_{1|4} \theta_{1|5} \\ \theta_{2|1} \theta_{2|2} \theta_{2|3} \theta_{2|4} \theta_{2|5} \\ \theta_{3|1} \theta_{3|2} \theta_{3|3} \theta_{3|4} \theta_{3|5} \\ \theta_{4|1} \theta_{4|2} \theta_{4|3} \theta_{4|4} \theta_{4|5} \\ \theta_{5|1} \theta_{5|2} \theta_{5|3} \theta_{5|4} \theta_{5|5} \end{bmatrix}$$

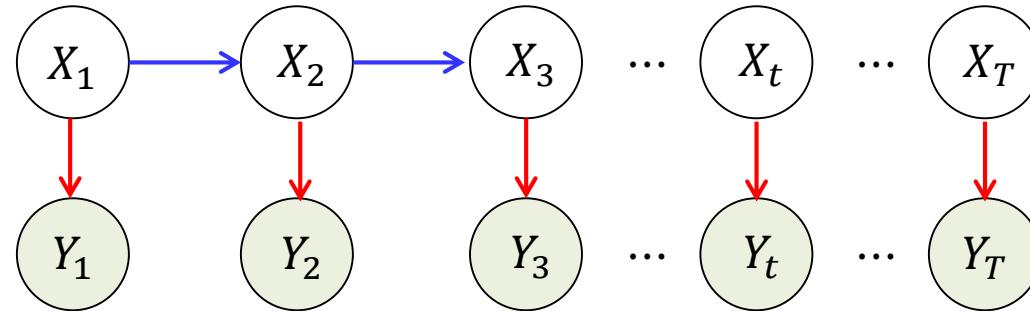
State transition matrix

$$\sum_i \theta_{i|j} = 1$$

If we have  $s_{1:t} = 1, 3, 2, 4, 1, 4, 3, 5, 1, 3, 4, 2, 1, 4, 4, 2, 4, 5, 1, 3, 3, 4, \dots$   $\rightarrow \theta_{3|1} = \frac{3}{5}$

## Definition of Hidden Markov Models

- The Hidden Markov Model (HMM) defines a Markov chain on hidden variables  $X_{1:t}$
- The observed variables are dependent on the hidden variables through an emission  $P(Y_t|X_t)$



- The joint distribution on the hidden variables and observations are

$$P(X_{1:t}, Y_{1:t}) = P(X_1)P(Y_1|X_1) \prod_{t=2}^T P(X_t|X_{t-1})P(Y_t|X_t)$$

- **Transition distribution:** For a stationary HMM the transition distribution  $P(X_t|X_{t-1})$  is defined as the  $H \times H$  matrix

$$M_{i,j} = P(X_t = i | X_{t-1} = j)$$

- **Emission distribution:** For a stationary HMM and emission distribution with discrete states  $Y_t \in \{1, \dots, V\}$ , we define  $V \times H$  matrix

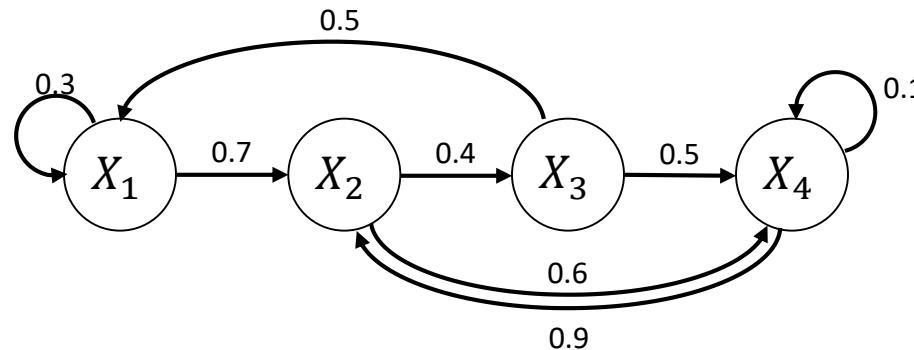
$$O_{i,j} = P(Y_t = i | X_t = j)$$

## Hidden Markov Model

The state variable  $X_t$  is discrete

- The state transition model  $P(X'|X)$  is usually sparse,  
→ can be represented as a directed graph

$$X = (X_1, X_2, X_3, X_4) : 4 \text{ discretized states}$$

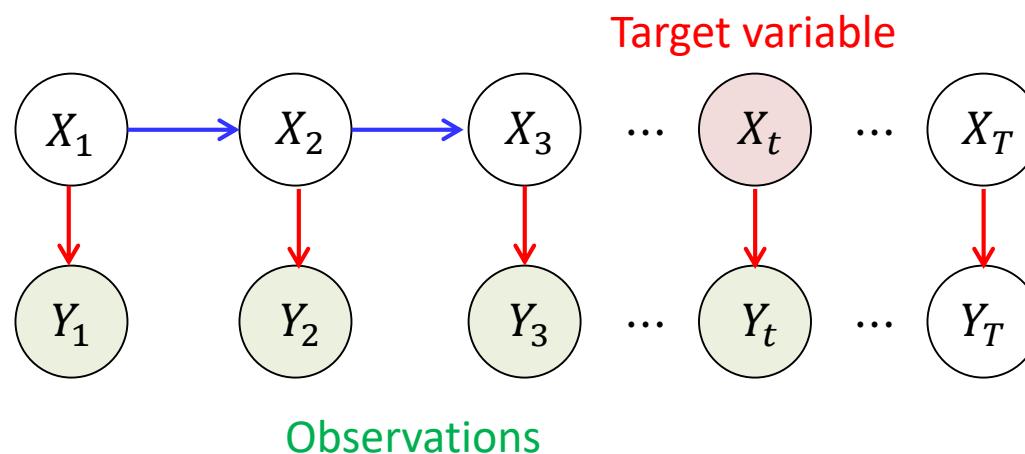


$$\sum_i P(X'_i|X) = 1$$

- The observation model :  $P(Y | X)$  can be deterministic or random

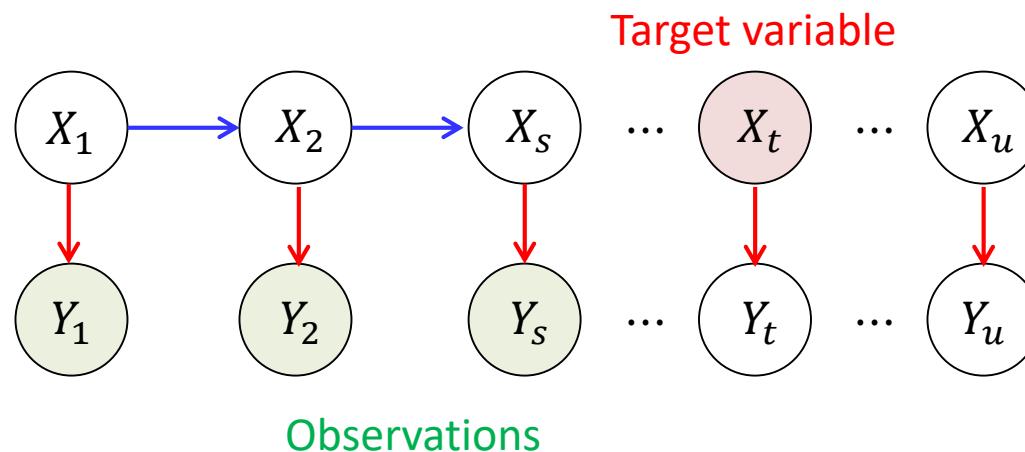
## Inferencing of Hidden Markov Models

- Filtering (inferencing the present)  $P(x_t|y_{1:t})$
- Prediction (inferencing the future)  $P(x_t|y_{1:s}) \quad t > s$
- Smoothing (inferencing the past)  $P(x_t|y_{1:u}) \quad t < u$
- Likelihood (inferencing the past)  $P(y_{1:t})$
- Most likely hidden path (Viterbi alignment)  $\operatorname{argmax}_{x_{1:t}} P(x_{1:t}|y_{1:t})$



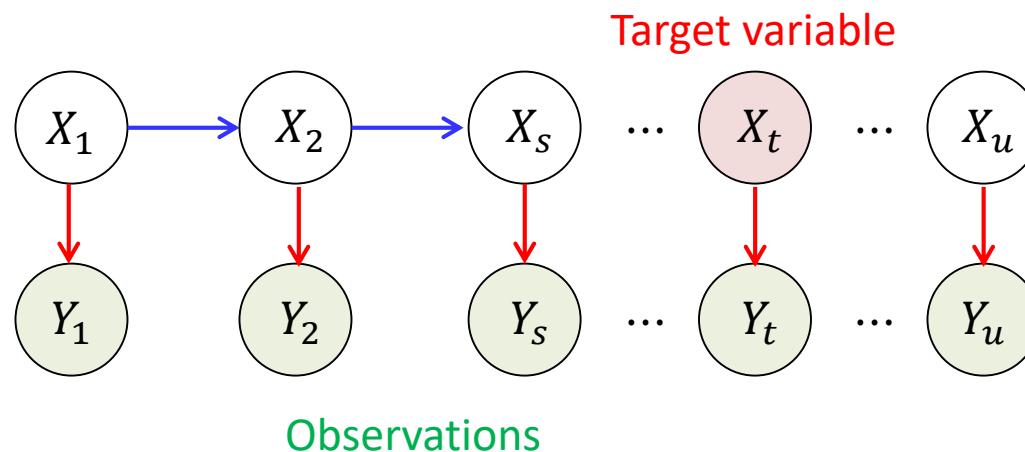
## Inferencing of Hidden Markov Models

- Filtering (inferencing the present)  $P(x_t|y_{1:t})$
- Prediction (inferencing the future)  $P(x_t|y_{1:s}) \quad t > s$
- Smoothing (inferencing the past)  $P(x_t|y_{1:u}) \quad t < u$
- Likelihood (inferencing the past)  $P(y_{1:t})$
- Most likely hidden path (Viterbi alignment)  $\operatorname{argmax}_{x_{1:t}} P(x_{1:t}|y_{1:t})$



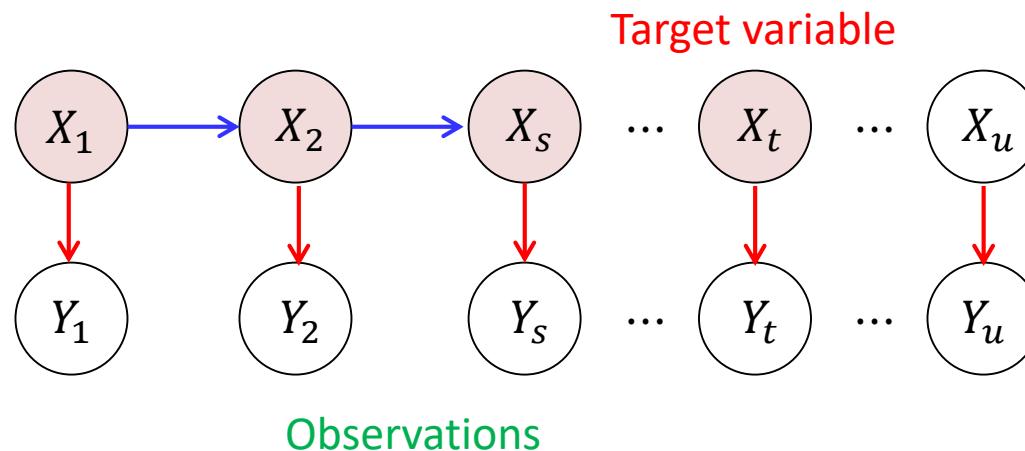
## Inferencing of Hidden Markov Models

- Filtering (inferencing the present)  $P(x_t|y_{1:t})$
- Prediction (inferencing the future)  $P(x_t|y_{1:s}) \quad t > s$
- Smoothing (inferencing the past)  $P(x_t|y_{1:u}) \quad t < u$
- Likelihood (inferencing the past)  $P(x_{1:t})$
- Most likely hidden path (Viterbi alignment)  $\operatorname{argmax}_{x_{1:t}} P(x_{1:t}|y_{1:t})$



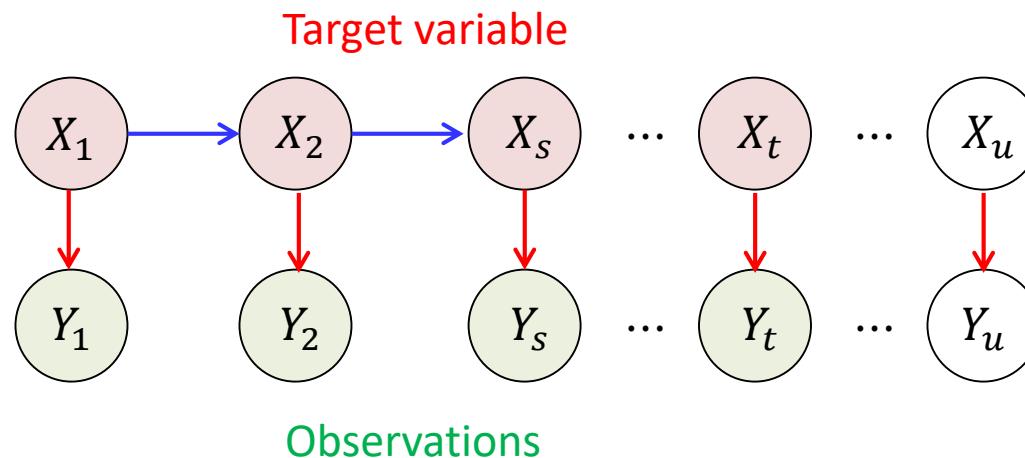
## Inferencing of Hidden Markov Models

- Filtering (inferencing the present)  $P(x_t|y_{1:t})$
- Prediction (inferencing the future)  $P(x_t|y_{1:s}) \quad t > s$
- Smoothing (inferencing the past)  $P(x_t|y_{1:u}) \quad t < u$
- Likelihood (inferencing the past)  $P(y_{1:t})$
- Most likely hidden path (Viterbi alignment)  $\operatorname{argmax}_{x_{1:t}} P(x_{1:t}|y_{1:t})$



## Inferencing of Hidden Markov Models

- Filtering (inferencing the present)  $P(x_t|y_{1:t})$
- Prediction (inferencing the future)  $P(x_t|y_{1:s}) \quad t > s$
- Smoothing (inferencing the past)  $P(x_t|y_{1:u}) \quad t < u$
- Likelihood (inferencing the past)  $P(y_{1:t})$
- Most likely hidden path (Viterbi alignment)  $\operatorname{argmax}_{x_{1:t}} P(x_{1:t}|y_{1:t})$



## Inferencing of Hidden Markov Models

- Filtering (inferencing the present)  $P(x_t|y_{1:t})$

$$P(x_t|y_{1:t}) = \frac{P(x_t, y_{1:t})}{P(y_{1:t})} \propto P(x_t, y_{1:t})$$

$$P(x_t, y_{1:t}) = \sum_{x_{t-1}} P(x_t, \cancel{x_{t-1}}, y_{1:t-1}, y_t)$$

$$\sum_{x_{t-1}} P(y_t|y_{1:t-1}, x_t, \cancel{x_{t-1}}) P(x_t|y_{1:t-1}, x_{t-1}) \cancel{P(x_{t-1}, y_{1:t-1})}$$

$$\sum_{x_{t-1}} P(y_t|x_t) P(x_t|x_{t-1}) \cancel{P(x_{t-1}, y_{1:t-1})} \quad ::\text{Conditional independence}$$

$$P(y_t|x_t) \sum_{x_{t-1}} P(x_t|x_{t-1}) \cancel{P(x_{t-1}, y_{1:t-1})}$$

corrector

predictor

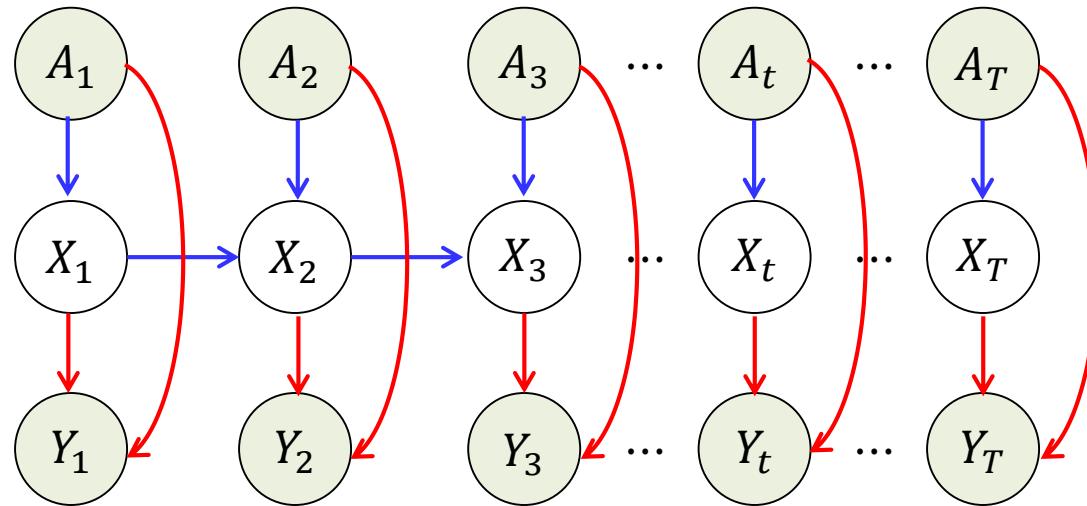
### Bayesian view

$$P(x_t|y_{1:t}) \propto \sum_{x_{t-1}} \underline{P(y_t|x_t) P(x_t|x_{t-1})} \cancel{P(x_{t-1}|y_{1:t-1})}$$

Regarded as likelihood

Modified prior distribution has an effect of  
removing all nodes in the graph before time  $t - 1$

## Input and output hidden Markov Model (IOHMM)



The state transition model :  $P(X_t|X_{t-1}, A_t)$

The observation model :  $P(Y_t|X_t, A_t)$

**Continuous domain**

## Continuous-state Markov models

- In many practical time series applications, the data is naturally continuous (i.e., variables are not discretized), particularly for models of the physical environment
- Restrict the form of the continuous transition  $p(X_t|X_{t-1})$
- A simple yet powerful class of such transitions are the [linear dynamical systems](#)
- A *deterministic linear dynamical system* defines the temporal evolution of a vector  $x_t$  according to the discrete-time update equation

$$x_t = A_t x_{t-1}$$

where  $A_t$  is the transition matrix at time  $t$

- If  $A_t$  is invariant with  $t$ , the process is called stationary or time-invariant

## Observed linear dynamic system

- A *stochastic linear dynamical system* defines the temporal evolution of a vector  $x_t$  according to the discrete-time update equation

$$x_t = A_t x_{t-1} + \eta_t$$

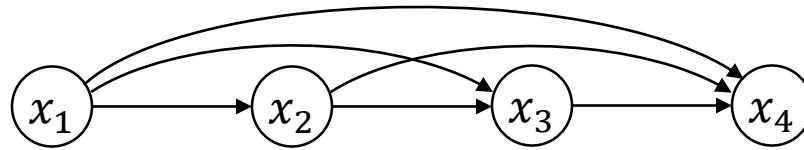
where  $\eta_t$  is a noise vector sampled from a Gaussian distribution

$$\eta_t \sim N(\mu_t, \Sigma_t)$$

- This is equivalent to a first-order Markov model with transition

$$p(x_t | x_{t-1}) = N(x_t | A_t x_{t-1} + \eta_t, \Sigma_t)$$

## Auto-regressive models



- A scalar time-invariant Auto-Regressive (AR) model is defined by

$$x_t = \sum_{l=1}^L a_l x_{t-l} + \eta_t, \quad \eta_t \sim N(\mu, \sigma^2)$$

where  $a = (a_1, a_2, \dots, a_L)^T$  are AR coefficients and  $\sigma^2$  is innovation noise.

- As a belief network, the AR model can be written as an  $L$ th-order Markov model:

$$p(x_{1:T}) = \prod_{t=1}^T p(x_t | x_{t-1}, \dots, x_{t-L}), \quad \text{with } x_i = 0 \text{ for } i \leq 0$$

$\hat{x}_{t-1} = (x_{t-1}, \dots, x_{t-L})$

with  $p(x_t | x_{t-1}, \dots, x_{t-L}) = N(x_t | \sum_{l=1}^L a_l x_{t-l} + \eta_t, \sigma^2) = N(x_t | a^T \hat{x}_{t-1}, \sigma^2)$

Similar to Bayesian Regression

- Heavily used in financial time series prediction, being able to capture simple trends in the data
- The AR coefficients form a compressed representation of the signal

## Training Auto-regressive model

- Maximum likelihood training of the AR coefficients is straightforward based on

$$\begin{aligned}\log p(x_{1:T}) &= \log \prod_{t=1}^T p(x_t | x_{t-1}, \dots, x_{t-L}) \\ &= \sum_{t=1}^T \log(x_t | \hat{x}_{t-1}) \\ &= -\frac{1}{2\sigma^2} \sum_{t=1}^T (x_t - a^T \hat{x}_{t-1})^2 - \frac{T}{2} \log(2\pi\sigma^2)\end{aligned}$$

- Differentiating w.r.t.  $a$  and equating to zero we arrive at

$$\begin{aligned}\sum_{t=1}^T (\textcolor{red}{x}_t - a^T \hat{x}_{t-1}) \hat{x}_{t-1} &= 0 \\ \rightarrow a &= [\sum_t \hat{x}_{t-1} \hat{x}_{t-1}^T]^{-1} \sum_t \textcolor{red}{x}_t \hat{x}_{t-1} \quad \textcolor{red}{x}_t : \text{target output (scalar)}\end{aligned}$$

- Similarly,

$$\sigma^2 = \frac{1}{T} \sum_{t=1}^T (x_t - a^T \hat{x}_{t-1})^2$$

## Time-varying Auto-regressive model

- Learning the AR coefficients as a problem in inference in a latent linear dynamical system (LDS):

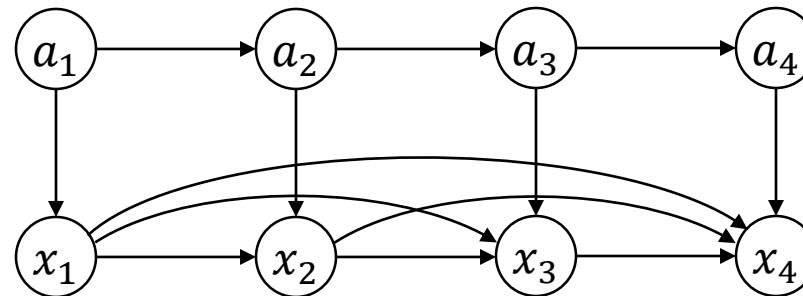
$$x_t = \hat{x}_{t-1}^T a_t + \eta_t, \quad \eta_t \sim N(0, \sigma^2)$$

which can be viewed as the emission distribution of a latent LDS in which the hidden variable is  $a_t$  and the time dependent emission matrix is given by  $\hat{x}_{t-1}^T$

- By placing a simple latent transition

$$a_t = a_{t-1} + \eta_t^a, \quad \eta_t^a \sim N(0, \sigma_a^2 \mathbf{I})$$

which encourages the AR coefficients to change slowly with time



## Time-varying Auto-regressive model

- Learning the AR coefficients as a problem in inference in a latent linear dynamical system (LDS):

$$x_t = \hat{x}_{t-1}^T a_t + \eta_t, \quad \eta_t \sim N(0, \sigma^2)$$

which can be viewed as the emission distribution of a latent LDS in which the hidden variable is  $a_t$  and the time dependent emission matrix is given by  $\hat{x}_{t-1}^T$

- By placing a simple latent transition

$$a_t = a_{t-1} + \eta_t^a, \quad \eta_t^a \sim N(0, \sigma_a^2 \mathbf{I})$$

which encourages the AR coefficients to change slowly with time

- The joint distribution between the observation  $x_{1:T}$  and the coefficients  $a_{1:T}$

$$p(a_{1:T}|x_{1:T}) \propto p(x_{1:T}, a_{1:T}) = \prod_{t=2}^T p(x_t|a_t, \hat{x}_{t-1}) p(a_t|a_{t-1})$$

then we can compute

$$a_{1:T}^* = \operatorname{argmax}_{a_{1:T}} p(a_{1:T}|x_{1:T})$$

from which the MAP estimates for the AR coefficients can be determined

## Time-varying variance Auto-regressive model

- For some applications, particularly in finance, the variance can change with time due to volatility

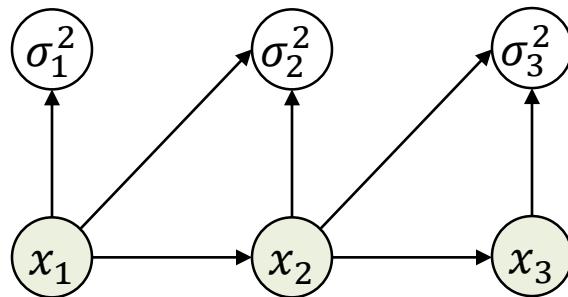
$$x_t = \sum_{l=1}^L a_l x_{t-l} + \eta_t, \quad \eta_t \sim N(\mu, \sigma_t^2)$$

→ Time varying variance

$$\bar{x}_t = \sum_{l=1}^L a_l x_{t-l}$$

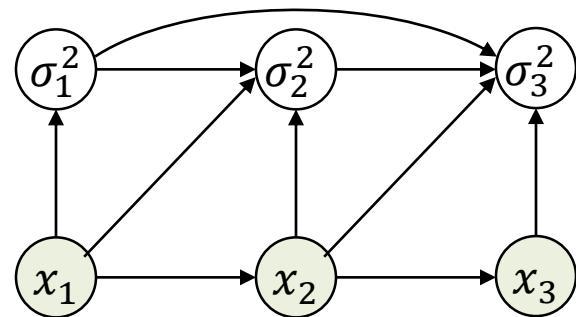
- The estimated time varying variance of noise can be computed

Auto Regressive Conditional Heteroscedasticity (ARCH)



$$\sigma_t^2 = \sigma_0 + \sum_{i=1}^q \alpha_i (x_{t-i} - \bar{x}_{t-i})^2$$

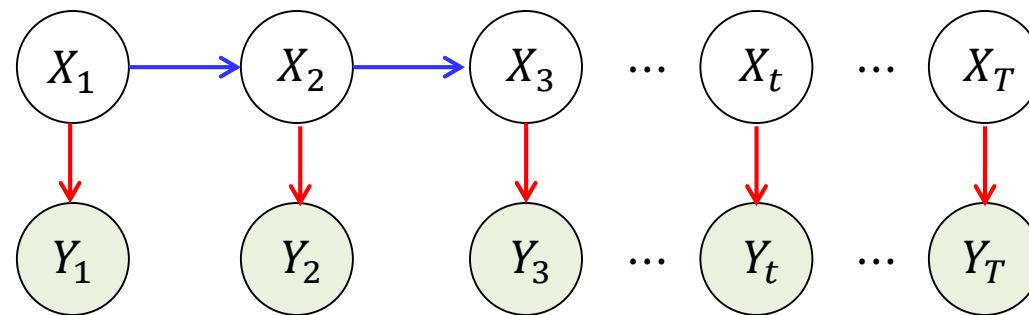
Generalized ARCH model (GARCH)



$$\sigma_t^2 = \sigma_0 + \sum_{i=1}^q \alpha_i (x_{t-i} - \bar{x}_{t-i})^2 + \sum_{i=1}^q \beta_i \sigma_{t-i}^2$$

## Linear Gaussian State Space Model

- The latent LDS defines a stochastic linear dynamical system in a latent space on a sequence of states  $x_{1:T}$
- Observations  $y_{1:T}$  are used to infer the hidden states that tracks or explains the system evolution



Transition model :  $x_t = A_t x_{t-1} + \eta_t^x,$

$$\eta_t^x \sim N(\bar{x}_t, \Sigma_t^x)$$

Emission model :  $y_t = B_t x_t + \eta_t^y,$

$$\eta_t^y \sim N(\bar{y}_t, \Sigma_t^y)$$

$A_t$ : transition matrix

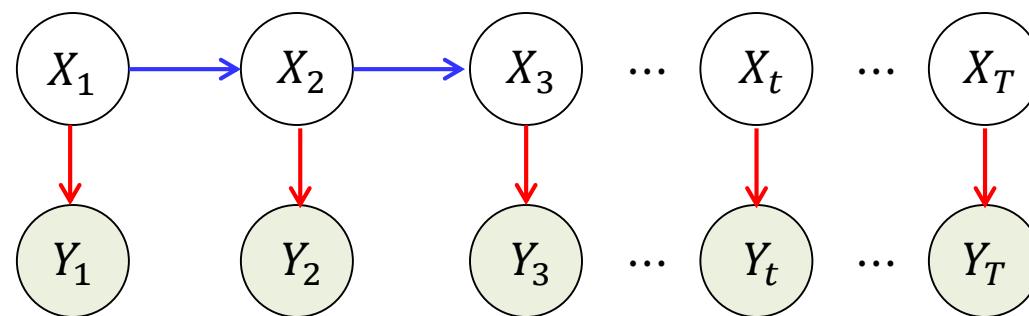
$B_t$ : emission matrix

$\eta_t^x$  transition noise vector with a hidden bias  $\bar{x}_t$

$\eta_t^y$  emission noise vector with a hidden bias  $\bar{y}_t$

## Linear Gaussian State Space Model

- The latent LDS defines a stochastic linear dynamical system in a latent space on a sequence of states  $x_{1:T}$
- Observations  $y_{1:T}$  are used to infer the hidden states that tracks or explains the system evolution



Transition model :  $p(x_t|x_{t-1}) = N(x_t|A_t x_{t-1} + \bar{x}_t, \Sigma_t^x)$ ,       $p(x_1) = N(x_1|\mu_\pi, \Sigma_\pi)$

Emission model :  $p(y_t|x_t) = N(y_t|B_t x_t + \bar{y}_t, \Sigma_t^y)$

- The first order Markov model is then defined as

$$p(x_{1:T}, y_{1:T}) = p(x_1)p(y_1|x_1) \prod_{t=2}^T p(x_t|x_{t-1})p(y_t|x_t)$$

## Kalman Filter

- Recall the filtering recursion for HMM:

$$P(x_t|y_{1:t}) \propto \sum_{x_{t-1}} P(y_t|x_t)P(x_t|x_{t-1})P(x_{t-1}|y_{1:t-1})$$

- For linear Gaussian State-space model, the recursion becomes

$$P(x_t|y_{1:t}) \propto \int_{x_{t-1}} P(y_t|x_t)P(x_t|x_{t-1})P(x_{t-1}|y_{1:t-1}) \text{ for } t > 1$$

- Since the product of two Gaussians is another Gaussian, and the integral of a Gaussian is another Gaussian,  $P(x_t|y_{1:t})$  is Gaussian:

$$P(x_t|y_{1:t}) = N(x_t|f_t, F_t)$$

- Thus the recursion is for computing the mean  $f_t$  and the variance  $F_t$  for  $P(x_t|y_{1:t})$  using  $f_{t-1}$  and the variance  $F_{t-1}$  for  $P(x_{t-1}|y_{1:t-1})$



$$P(x_{t-1}|y_{1:t-1}) = N(x_{t-1}|f_{t-1}, F_{t-1})$$

$$P(x_t|y_{1:t}) = N(x_t|f_t, F_t)$$

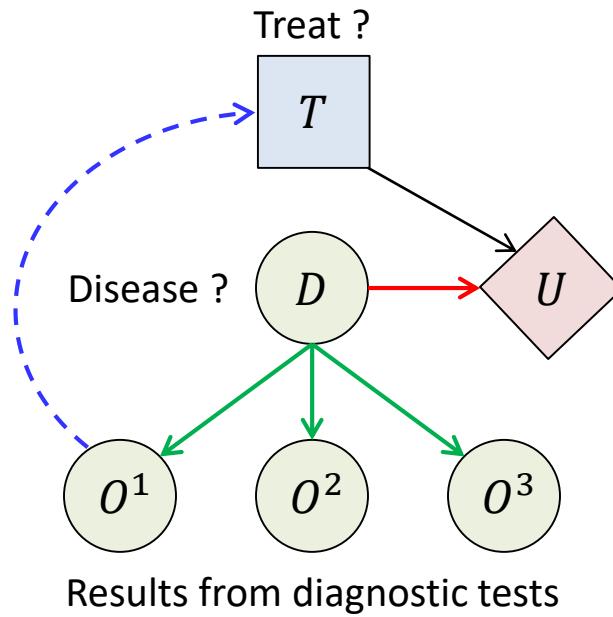
## **Supplements**

## L11. Influential Diagram

## Introduction

**Bayesian Network + Decision node + Utility node = Decision network (Influential Diagram)**

- make rational decisions based on a probabilistic model and utility function



- A chance node** corresponds to a random variable
- A decision node** corresponds to each decision to be made
- A utility node** corresponds to an additive utility component

## Utility theory

How to compare the plausibility of different statements?

$A$  : “we can be a millinery if we go to graduate school”

vs

$B$  : “we can be a millenary if we go to Samsung”

- If you believe  $A$  more than  $B$ , you can write  $A > B$
- If you believe  $B$  more than  $A$ , you can write  $A < B$
- If you have the same belief, you can write  $A \sim B$

## Constraints on Rational Preference

Assumptions about relationships of  $\succ$  and  $\sim$

- Completeness (comparability) : either  $A \succ B$ ,  $A \prec B$  or  $A \sim B$
- Transitivity : if  $A \succ B$  and  $B \succ C$ , then  $A \succ C$
- Continuity : if  $A \succ B \succ C$ , there exists a probability  $p$  such that  $[A:p; C:1-p] \sim B$
- Monotonicity: if  $A \succ B$ , then for any  $C$  and probability  $p$ ,  $[A:p; C:1-p] \succ [B:p; C:1-p]$

The degree of belief can be represented by a real-valued function:

- $P(A) > P(B)$  if and only if  $A \succ B$
- $P(A) = P(B)$  if and only if  $A \sim B$

## Utility functions

- Just as beliefs can be subjective, so can preferences.
- Preference operators can be used to compare preferences over uncertain outcomes. That is, a lottery is a set of probabilities  $p_{1:n}$  associated with a set of outcomes  $S_{1:n}$ .

$$[S_1:p_1; S_2:p_2; \dots; S_n:p_n]$$

- The utility of a lottery is given by

$$U([S_1:p_1; S_2:p_2; \dots; S_n:p_n]) = \sum_{i=1}^n p_i U(S_i)$$

## Maximum Expected Utility Principle

- Reach rational decisions with imperfect knowledge of the state of the world.
- Expected utility of taking action  $a$  given that we observe  $o$  and take action  $a$ :

$$\text{Probabilistic model} \quad \text{Utility function}$$
$$P(s'|o, a) \quad U(s')$$
$$\text{EU}(a|o) = \sum_{s'} P(s'|o, a)U(s')$$

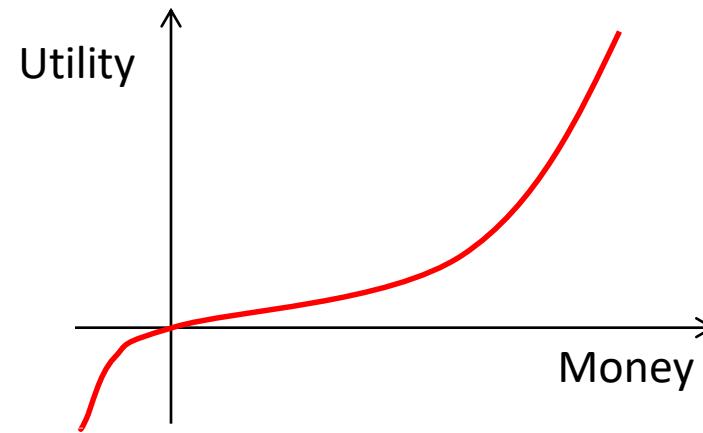
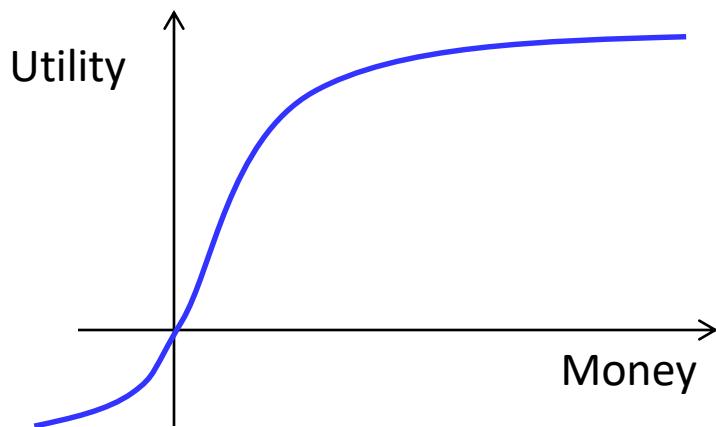
The diagram illustrates the components of expected utility. On the left, a green box contains the expression  $P(s'|o, a)$ . On the right, a red box contains the expression  $U(s')$ . A blue arrow points from the green box down to the formula  $\text{EU}(a|o) = \sum_{s'} P(s'|o, a)U(s')$ . Another blue arrow points from the red box to the same term  $U(s')$  in the formula.

- The principle of maximum expected utility : a rational agent should choose the action that maximizes expected utility:

$$a^* = \operatorname{argmax}_a \text{EU}(a|o)$$

## Utility of Money

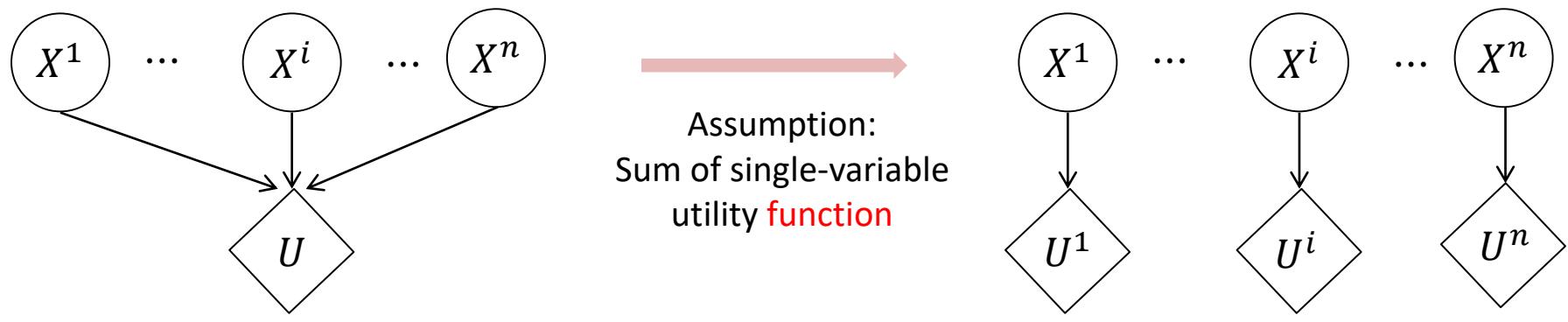
- Monetary values are often used to infer utility function.  
For example, cost of property and human loss damage caused by natural disasters
- It is well known that the relationship between utility and money is not linear as shown below



**A: Wining 1\$ with a probability 1** vs **B: Wining 100\$ with a probability 0.01**

- **Risk averse** : Preference for A (Utility function is concave)
- **Risk neutral** : There is no difference between A and B (Utility function is a linear)
- **Risk seeking** : Preference for B (Utility function is convex)

## Multiple Variable Utility Function



$$U(X^{1:n})$$

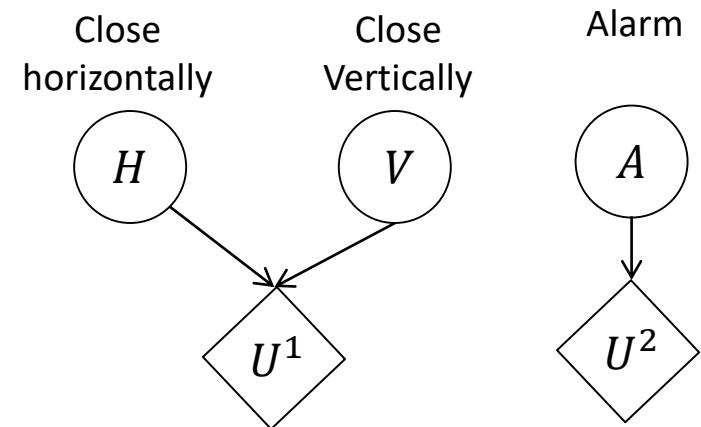
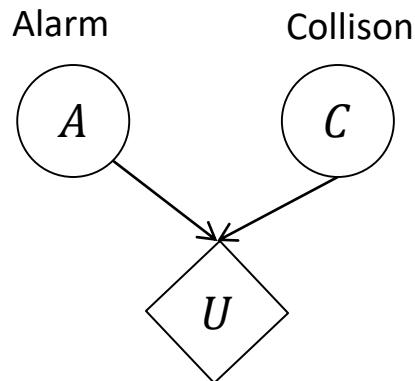
$$U(X^{1:n}) = \sum_{i=1}^n U(X^i)$$

- If  $X^i$  is a binary variable,  
 $2^n$  parameters are required to specify  $U(X^{1:n})$
- If  $X^i$  is a binary variable,  
 $2n$  parameters are required to specify  $U(X^{1:n})$

Different additive decomposition can be explicitly imposed on the network structure!

## Multiple Variable Utility Function

Example : Collision avoidance system



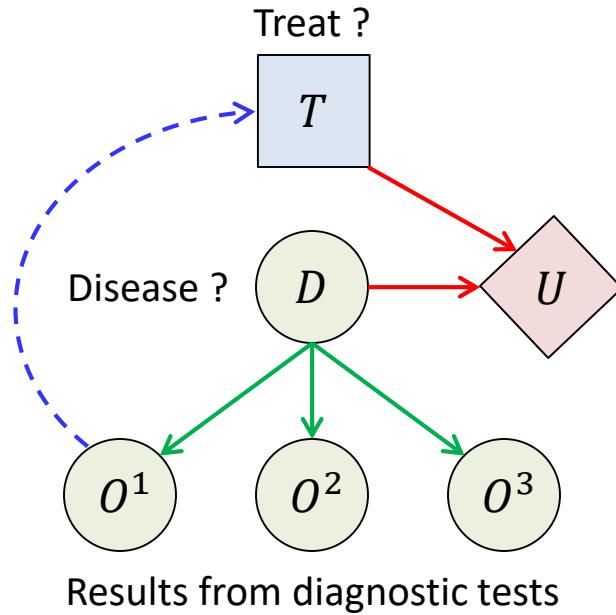
A	C	U
$a^0$	$c^0$	$U(a^0, c^0)$
$a^0$	$c^1$	$U(a^0, c^1)$
$a^1$	$c^0$	$U(a^1, c^0)$
$a^1$	$c^1$	$U(a^1, c^1)$

Additive decomposition of utility function

$$U(h, v, a) = U^1(h, v) + U^2(a)$$

## Decision Network

Bayesian Network + Decision node + Utility node = Decision network (Influential Diagram)



$T$	$D$	$U(T, D)$
0	0	0
0	1	-10
1	0	-1
1	1	-1

- Conditional edge
- Functional edge
- Information edge  
(often omitted)



A chance node corresponds to a random variable



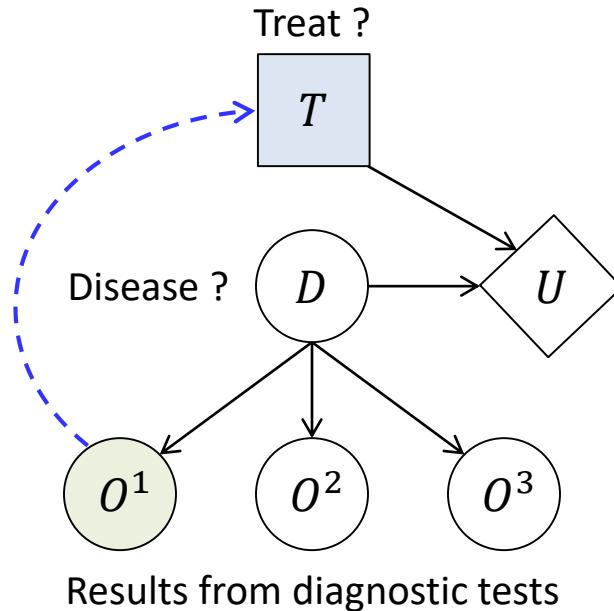
A decision node corresponds to each decision to be made



A utility node corresponds to an additive utility component

## Decision Network

Assume we only have a single observation  $O^1 = 1 (= o_1^1)$  from test 1



$$\begin{aligned} EU(t^1 | o_1^1) &= \sum_{o_3} \sum_{o_2} \sum_d P(d, o_2, o_3 | t^1, o_1^1) U(t^1, d, o_1^1, o_2, o_3) \\ &= \sum_d \underbrace{P(d | t^1, o_1^1)}_{\text{Can be computed using many inference methods}} U(t^1, d) \end{aligned}$$

Compare  $EU(t^0 | o_1^1)$  and  $EU(t^1 | o_1^1)$  and chose the treatment that lead maximum EU

## Value of Information

- It may be beneficial to administer additional diagnostic tests to reduce the uncertainty about the decease. Then, how to choose a test type to be conducted?
- Expected utility of optimal action given observation  $o$  :

$$EU^*(o) = \operatorname{argmax}_a EU(a|o)$$

- The value of information (VPI) about new variable  $O^{new}$  (**unobserved**) given the current observation  $o$  (**observed**):

$$\text{VOI}(O^{new}|o) = \left( \sum_{o^{new}} P(o^{new}|o) EU^*(o^{new}, o) \right) - EU^*(o)$$

- The value of information about a variable is the increase in expected utility with the observation of that variable
- VPI can only captures the increase in expected utility → need to consider the cost associated with observing the new information

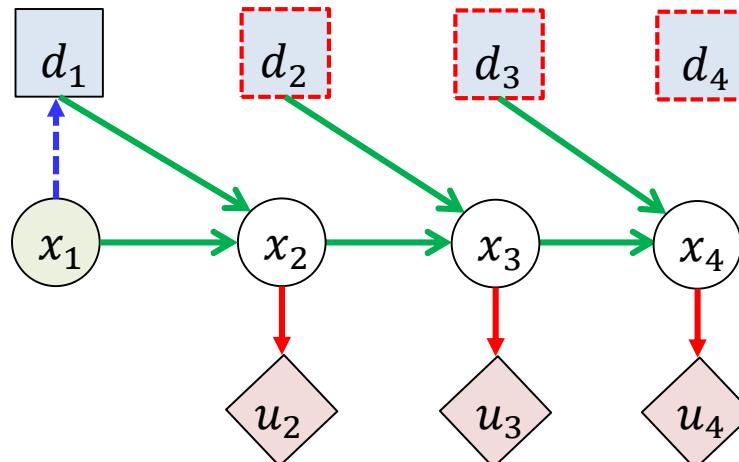
## Sequential Decision Making : Partial Ordering

- The sequential decision making problems can be solved by exploiting structure in the problem based on Bayesian Network and the corresponding inference routines
- The sequential decision making problem will be extended later to problems in **control theory** and **reinforcement learning**
- Influential Diagram defines a partial ordering of the nodes:

$$\mathcal{X}_0 \prec D_1 \prec \mathcal{X}_1 \prec D_2, \dots, \prec \mathcal{X}_{n-1} \prec D_n \prec \mathcal{X}_n$$

with  $\mathcal{X}_k$  being the variables revealed between decision  $D_k$  and  $D_{k+1}$

## Sequential Decision Making : Partial Ordering



- Transition probability :

$$p(x_{t+1}|x_t, d_t)$$

- Utility is

$$U(x_{1:4}) = \sum_{t=2}^4 u(x_t)$$

- The probability of the sequence

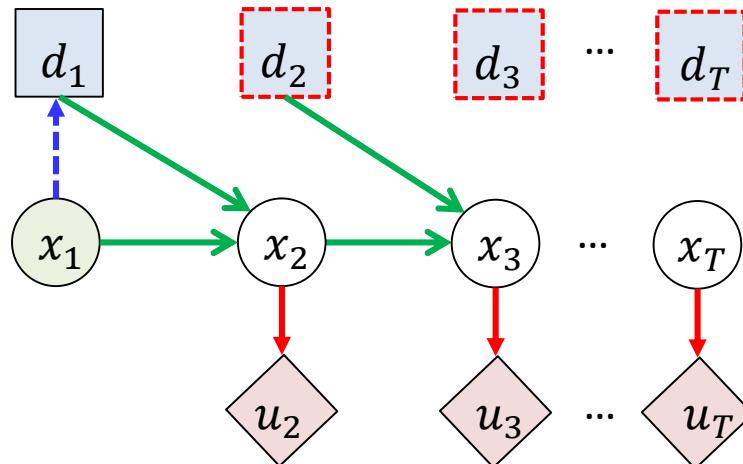
$$p(x_{2:4}|x_1, d_{1:3}) = \prod_{t=1}^3 p(x_{t+1}|x_t, d_t) = p(x_2|x_1, d_1)p(x_3|x_2, d_2)p(x_4|x_3, d_3)$$

- At time  $t = 1$ , we want to make the decision  $d_1$  that will lead to maximized expected total utility

$$U(d_1|x_1) = \sum_{x_2} \max_{d_2} \sum_{x_3} \max_{d_3} p(x_{2:4}|x_1, d_{1:3}) U(x_{2:4})$$

$$U(d_1|x_1) = \sum_{x_2} \max_{d_2} \sum_{x_3} \max_{d_3} \prod_{t=1}^3 p(x_{t+1}|x_t, d_t) \sum_{t=2}^4 u(x_t)$$

## Sequential Decision Making : Partial Ordering



- Transition probability :

$$p(x_{t+1}|x_t, d_t)$$

- Utility is

$$U(x_{1:T}) = \sum_{t=2}^T u(x_t)$$

- The probability of the sequence

$$p(x_{2:T}|x_1, d_{1:T-1}) = \prod_{t=1}^{T-1} p(x_{t+1}|x_t, d_t)$$

- At time  $t = 1$ , we want to make the decision  $d_1$  that will lead to maximized expected total utility

$$U(d_1|x_1) = \sum_{x_2} \max_{d_2} \sum_{x_3} \max_{d_3} \sum_{x_4} \dots \max_{d_{T-1}} \sum_{x_T} p(x_{2:T}|x_1, d_{1:T-1}) U(x_{2:T})$$

$$U(d_1|x_1) = \sum_{x_2} \max_{d_2} \sum_{x_3} \max_{d_3} \sum_{x_4} \dots \max_{d_{T-1}} \sum_{x_T} \prod_{t=1}^{T-1} p(x_{t+1}|x_t, d_t) \sum_{t=2}^T u(x_t)$$

## Sequential Decision Making : Partial Ordering

- The sequential decision making problems can be solved by exploiting structure in the problem based on Bayesian Network and the corresponding inference routines
- The sequential decision making problem will be extended later to problems in **control theory** and **reinforcement learning**
- Influential Diagram defines a partial ordering of the nodes:

$$\mathcal{X}_0 \prec D_1 \prec \mathcal{X}_1 \prec D_2, \dots, \prec \mathcal{X}_{n-1} \prec D_n \prec \mathcal{X}_n$$

with  $\mathcal{X}_k$  being the variables revealed between decision  $D_k$  and  $D_{k+1}$

- The optimal first decision  $D_1 = d_1$  is determined by computing

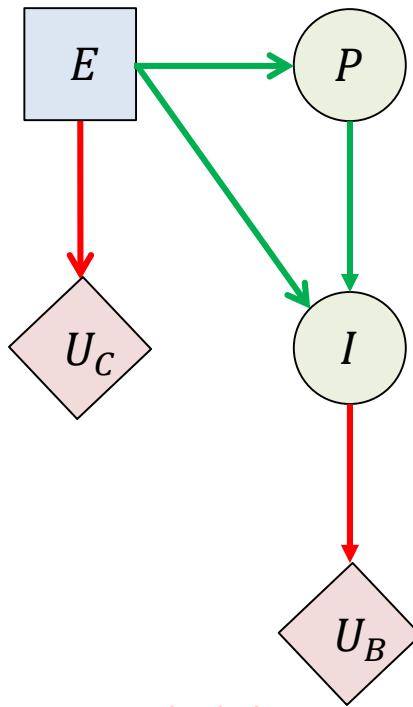
$$U(d_1|x_0) \equiv \sum_{\mathcal{X}_1} \max_{D_2} \dots \sum_{\mathcal{X}_{n-1}} \max_{D_n} \sum_{\mathcal{X}_n} \prod_{i \in \mathcal{L}} p(x_i | \text{pa}(x_i)) \sum_{j \in \mathcal{T}} U_j(\text{pa}(u_j))$$

$\mathcal{L}$  is a set of random variables and  $\mathcal{T}$  is a set of utility variables

- The optimal first decision  $D_1^*$  is determined as

$$d_1^* = \operatorname{argmax}_{d_1} U(d_1|x_0)$$

## Example: Should I do a PhD?

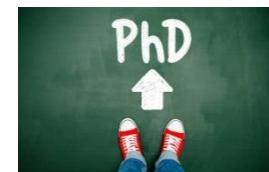


### Do PhD to wind a Novel Prize?

The ordering :  $E^* < \{I, P\}$

#### Domains

- $\text{dom}(E) = \{\text{do PhD, no PhD}\}$
- $\text{dom}(P) = \{\text{prize, no prize}\}$
- $\text{dom}(I) = \{\text{low, average, high}\}$



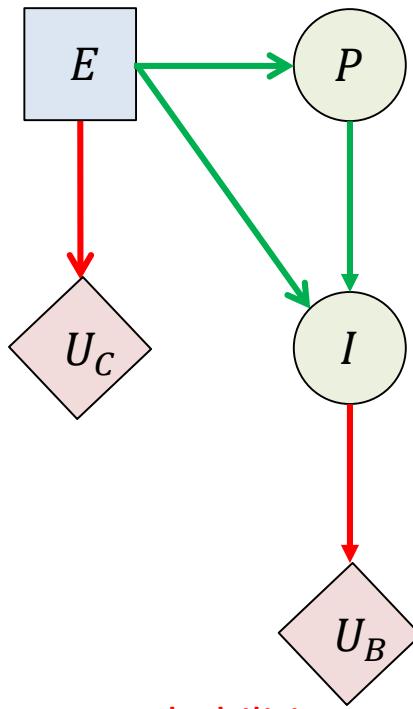
#### Utilities

- $U_C(\text{do PhD}) = -50000, U_C(\text{no PhD}) = 0$
- $U_B(\text{low}) = 100000, U_B(\text{average}) = 200000, U_B(\text{high}) = 500000$

#### Probabilities

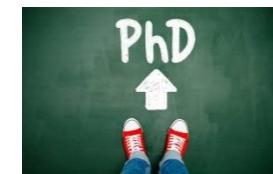
- $p(\text{win Novel Prize}|\text{no PhD}) = 0.0000001, p(\text{win Novel Prize}|\text{do PhD}) = 0.001$
- $p(\text{low}|\text{do PhD, no prize}) = 0.1, p(\text{average}|\text{do PhD, no prize}) = 0.5, p(\text{high}|\text{do PhD, no prize}) = 0.4$
- $p(\text{low}|\text{no PhD, no prize}) = 0.2, p(\text{average}|\text{no PhD, no prize}) = 0.6, p(\text{high}|\text{no PhD, no prize}) = 0.2$
- $p(\text{low}|\text{do PhD, prize}) = 0.01, p(\text{average}|\text{do PhD, prize}) = 0.04, p(\text{high}|\text{do PhD, prize}) = 0.95$
- $p(\text{low}|\text{no PhD, prize}) = 0.01, p(\text{average}|\text{no PhD, prize}) = 0.04, p(\text{high}|\text{no PhD, prize}) = 0.95$

## Example: Should I do a PhD?



### Do PhD to wind a Novel Prize?

The ordering :  $E^* < \{I, P\}$



#### Domains

- $\text{dom}(E) = \{\text{do PhD, no PhD}\}$
- $\text{dom}(P) = \{\text{prize, no prize}\}$
- $\text{dom}(I) = \{\text{low, average, high}\}$

#### Utilities

- $U_C(\text{do PhD}) = -50000, U_C(\text{no PhD}) = 0$
- $U_B(\text{low}) = 100000, U_B(\text{average}) = 200000, U_B(\text{high}) = 500000$

#### Probabilities

- $p(\text{win Novel Prize}|\text{no PhD}) = 0.0000001, p(\text{win Novel Prize}|\text{do PhD}) = 0.001$
- $p(\text{low}|\text{do PhD, no prize}) = 0.1, p(\text{average}|\text{do PhD, no prize}) = 0.5, p(\text{high}|\text{do PhD, no prize}) = 0.4$
- $p(\text{low}|\text{no PhD, no prize}) = 0.2, p(\text{average}|\text{no PhD, no prize}) = 0.6, p(\text{high}|\text{no PhD, no prize}) = 0.2$
- $p(\text{low}|\text{do PhD, prize}) = 0.01, p(\text{average}|\text{do PhD, prize}) = 0.04, p(\text{high}|\text{do PhD, prize}) = 0.95$
- $p(\text{low}|\text{no PhD, prize}) = 0.01, p(\text{average}|\text{no PhD, prize}) = 0.04, p(\text{high}|\text{no PhD, prize}) = 0.95$

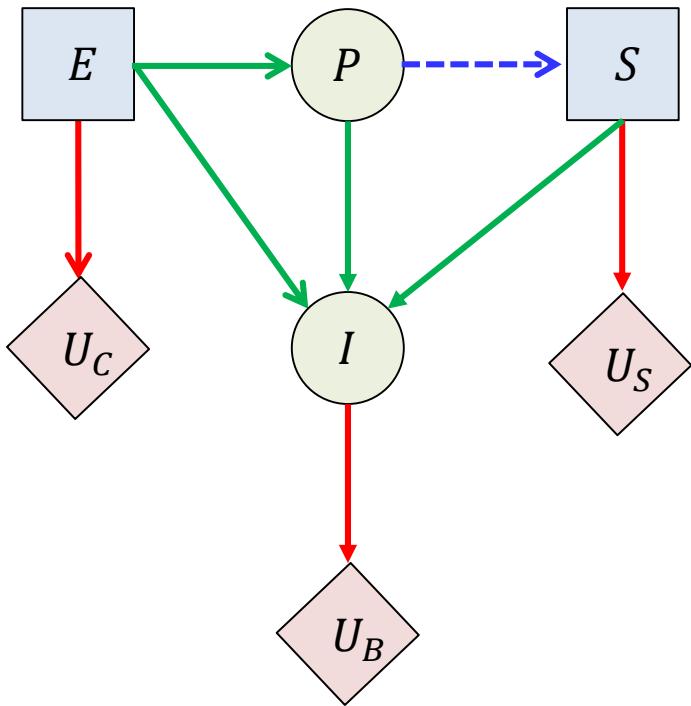
The expected utility of Education is

$$U(E) = \sum_{I,P} p(I|E, P)p(P|E)[U_C(E) + U_B(I)]$$

$$U(\text{do PhD}) = 260174$$

$$U(\text{no PhD}) = 240000$$

## Example: PhD and start-up companies



**Do PhD to wind a Novel Prize and start-up?**

The ordering :  $E^* \prec P \prec S^* \prec I$

**Domains**

- $\text{dom}(E) = \{\text{do PhD, no PhD}\}$
- $\text{dom}(P) = \{\text{prize, no prize}\}$
- $\text{dom}(I) = \{\text{low, average, high}\}$
- $\text{dom}(S) = \{\text{yes, no}\}$



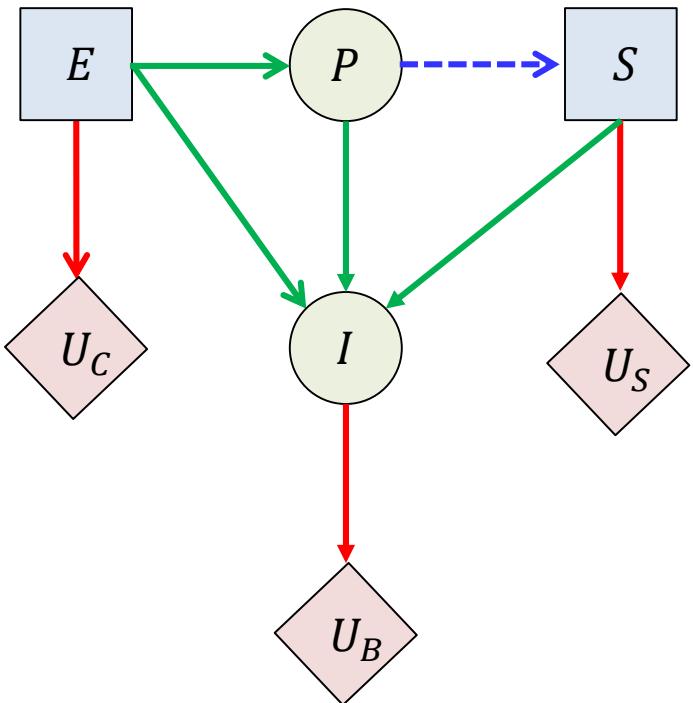
**Utilities**

- $U_C(\text{do PhD}) = -50000, U_C(\text{no PhD}) = 0$
- $U_B(\text{low}) = 100000, U_B(\text{average}) = 200000, U_B(\text{high}) = 500000$
- $U_S(\text{start up}) = -200000, U_S(\text{no start up}) = 0$

**Probabilities**

- $p(\text{win Novel Prize|no PhD}) = 0.0000001, p(\text{win Novel Prize|do PhD}) = 0.001$
- $p(\text{low|do PhD, no prize}) = 0.1, p(\text{average|do PhD, no prize}) = 0.5, p(\text{high|do PhD, no prize}) = 0.4$
- $p(\text{low|no PhD, no prize}) = 0.2, p(\text{average|no PhD, no prize}) = 0.6, p(\text{high|no PhD, no prize}) = 0.2$
- $p(\text{low|do PhD, prize}) = 0.01, p(\text{average|do PhD, prize}) = 0.04, p(\text{high|do PhD, prize}) = 0.95$
- $p(\text{low|no PhD, prize}) = 0.01, p(\text{average|no PhD, prize}) = 0.04, p(\text{high|no PhD, prize}) = 0.95$
- $p(\text{low|start up, no prize}) = 0.1, p(\text{average|start up, no prize}) = 0.5, p(\text{high|start up, no prize}) = 0.4$
- $p(\text{low|no start up, no prize}) = 0.2, p(\text{average|no start up, no prize}) = 0.6, p(\text{high|no start up, no prize}) = 0.2$
- $p(\text{low|start up, prize}) = 0.005, p(\text{average|start up, prize}) = 0.005, p(\text{high|start up, prize}) = 0.99$
- $p(\text{low|no start up, prize}) = 0.05, p(\text{average|no start up, prize}) = 0.15, p(\text{high|no start up, prize}) = 0.8$

## Example: PhD and start-up companies



**Do PhD to wind a Novel Prize and start-up?**

The ordering :  $E^* < P < S^* < I$

Domains

- $\text{dom}(E) = \{\text{do PhD, no PhD}\}$
- $\text{dom}(P) = \{\text{prize, no prize}\}$
- $\text{dom}(I) = \{\text{low, average, high}\}$
- $\text{dom}(S) = \{\text{yes, no}\}$

Utilities

- $U_C(\text{do PhD}) = -50000, U_C(\text{no PhD}) = 0$
- $U_B(\text{low}) = 100000, U_B(\text{average}) = 200000, U_B(\text{high}) = 500000$
- $U_S(\text{start up}) = -200000, U_S(\text{no start up}) = 0$

- Our interest is to advise whether or not it is desirable to take a PhD, bearing mind that later one may or may not win the Novel Prize, and may or may not form a start-up company
- The expected optimal utility for any state  $E$  is

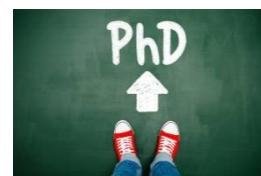
$$U(E) = \sum_P \max_S \sum_I p(I|S, P)p(P|E)[U_C(E) + U_B(I) + U_S(S)]$$

(where we assume that the optimal decisions are taken in the future)

$U(\text{do PhD}) = 190195$
-----------------------------

$U(\text{no PhD}) = 240002$
-----------------------------

Google



## Bayesian View on Bandit Problem (MDP formulation)

The posterior of the success probability given  $w_t$  winnings and  $l_t$  loss :

$$\theta_t | w_t, l_t \sim \text{Beta}(\theta | \alpha = 1 + w_t, \beta = 1 + l_t)$$

The mean probability of success :

$$\rho_i = \int_0^1 \theta \text{Beta}(\theta | \alpha = 1 + w_i, \beta = 1 + l_i) d\theta = \frac{w_i + 1}{w_i + l_i + 2}$$



data

$w$

MLE

$$\theta = \frac{1}{1}$$

Bayesian

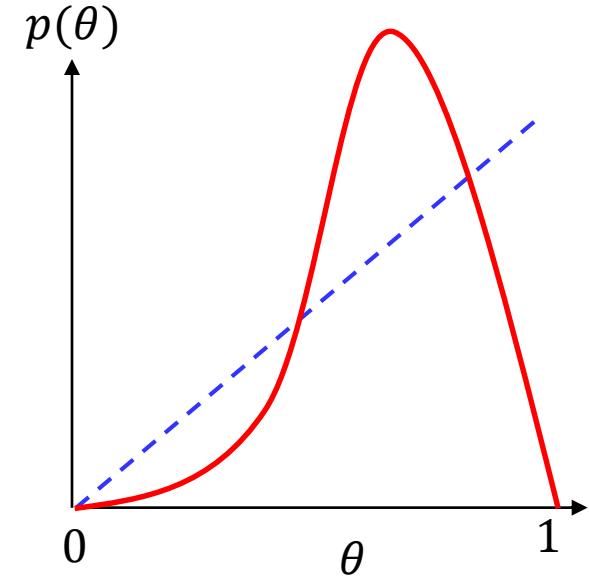
$$\theta | \text{data} \sim \text{Beta}(2,1)$$



$w, w, w, l, w$

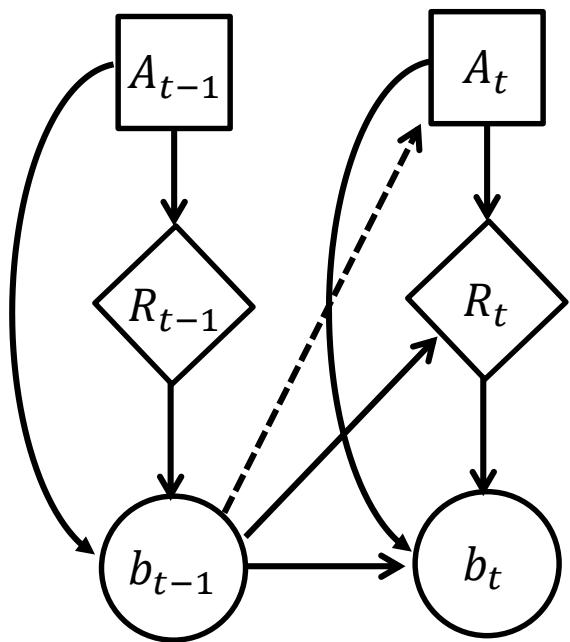
$$\theta = \frac{4}{5}$$

$$\theta | \text{data} \sim \text{Beta}(5,2)$$



## Bayesian View on Bandit Problem (MDP formulation)

### MDP over belief state and finding optimal policy using Dynamic Programming



- $h_t = [(a_1, r_1), (a_1, r_1), \dots, (a_t, r_t)]$
- $\theta = (\theta_i, \dots, \theta_n)$  : Unknown machine parameters
- Belief state  $b_t(\theta) = P(\theta|h_t)$  : probability dist. on para.
- Updating **belief state**  $b_t(\theta)$  for Binary bandit with prior Beta( $\theta_i|\alpha, \beta$ ) : **Deterministic**

$$b_t(\theta_i) = b_{t-1}[a_t, r_t] = \text{Beta}(\theta_i|\alpha + w_{t,i}, \beta + l_{t,i})$$

$w_{t,i}$ : Accumulated wins with arm  $i$  up to time  $t$

$l_{t,i}$ : Accumulated loses with arm  $i$  up to time  $t$

### Dynamic programming on the value function

$$V_{t-1}(b_{t-1}) = \max_{\pi} E \left[ \sum_{t=t}^T r_t \right]$$

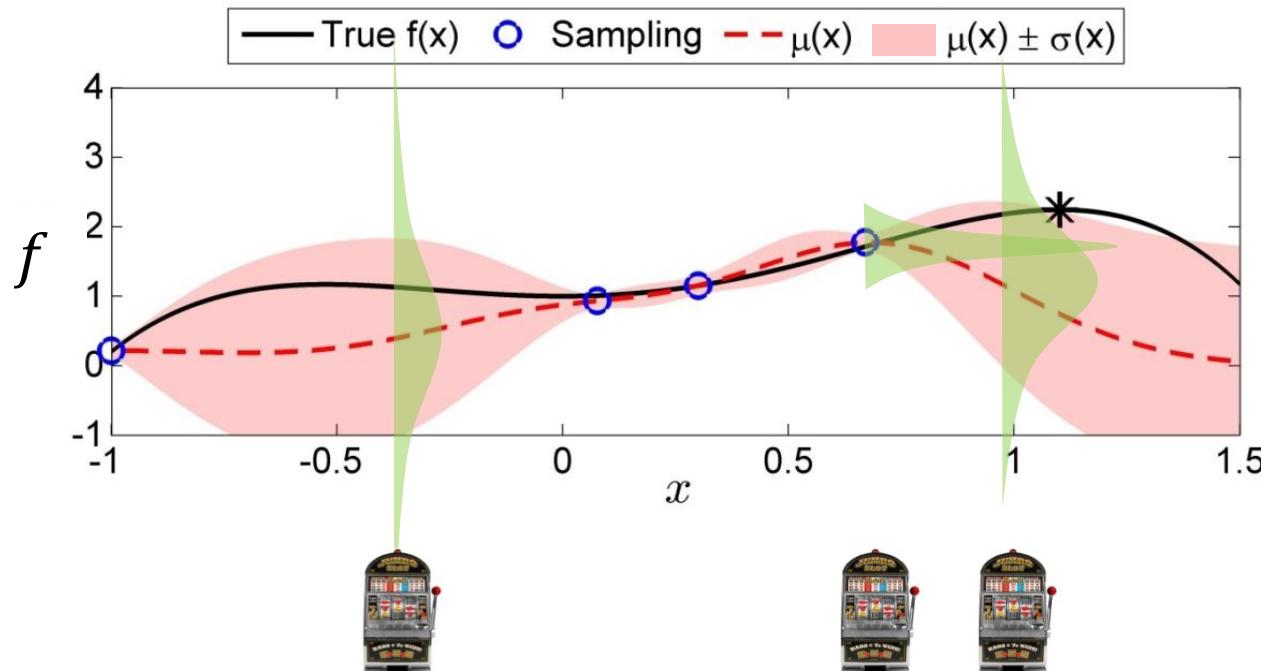
$$= \max_{a_t} \sum_{r_t} P(r_t|a_t, b_{t-1}) [r_t + V_t(b_{t-1}[a_t, r_t])]$$

$$P(r|a, b) = \int_{\theta_a} b(\theta_a) P(r|\theta_a) d\theta_a$$

# $\infty$ – armed Bandit Problem

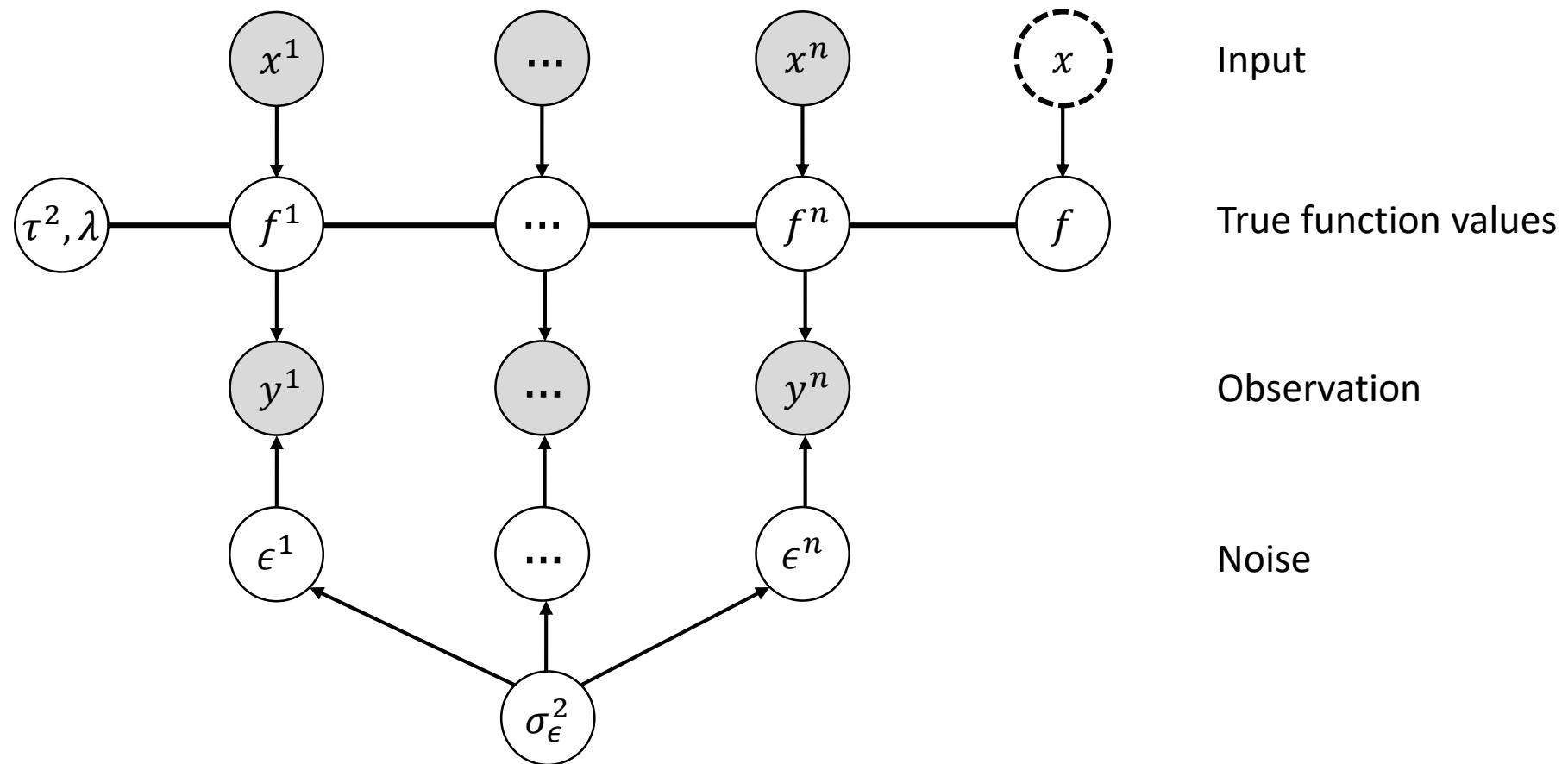
## How to solve?

- Learn target function (exploration)
- Improve target value (exploitation)



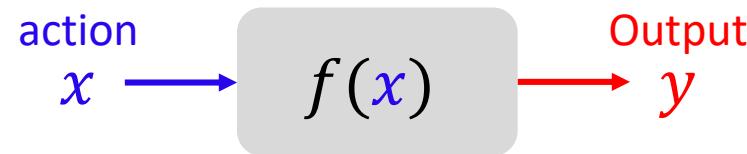
Which machine should be selected?

## Gaussian Process



## Bayesian Optimization

- **Bayesian Optimization (BO)** is a method to maximize (or minimize) a target value using measurement data from the **unknown target system**.
- BO is composed of three iterative steps:
  - (1) Learning : construct a probabilistic model function for a target system
  - (2) Optimization : select the next trial input to improve a target
  - (3) Observation : measure the output of a target system



- Policy  $\pi$  maps all the history to new action:  
$$\pi: [(\textcolor{blue}{x^1}, \textcolor{red}{y^1}), (\textcolor{blue}{x^2}, \textcolor{red}{y^2}), \dots, (\textcolor{blue}{x^{n-1}}, \textcolor{red}{y^{n-1}})] \rightarrow \textcolor{blue}{x^n}$$
- Find the optimal policy  $\pi^*$  that maximizes  $E[\sum_{t=1}^T y^t]$  or  $E[y^T]$

$$x^* = \operatorname{argmax}_x f(x)$$

### Learning phase : Gaussian Process (GP) regression

Construct the distribution on unknown target value  $f = f(x)$  corresponding  $x$

1. Given the data at  $n$ th iteration

Inputs	Latent function values	$y^i = f^i + \epsilon^i$
$x^{1:n} = \{x^1, \dots, x^i, \dots, x^n\}$	$f^{1:n} = \{f^1, \dots, f^i, \dots, f^n\}$	<b>Observations</b> $y^{1:n} = \{y^1, \dots, y^i, \dots, y^n\}$

## Bayesian Optimization

### Learning phase : Gaussian Process (GP) regression

Construct the distribution on unknown target value  $f = f(\mathbf{x})$  corresponding  $\mathbf{x}$

1. Given the data at  $n$ th iteration

Inputs	Latent function values	$y^i = f^i + \epsilon^i$ Observations
$x^{1:n} = \{x^1, \dots, x^i, \dots, x^n\}$	$f^{1:n} = \{f^1, \dots, f^i, \dots, f^n\}$	$y^{1:n} = \{y^1, \dots, y^i, \dots, y^n\}$

2. Prior on the function values  $f^{1:n}$  is represented as Gaussian Process (GP)

$$p(f^{1:n}) = GP(m(\cdot), k(\cdot, \cdot)) : \begin{bmatrix} f^1 \\ \vdots \\ f^n \end{bmatrix} \sim N \left( \begin{bmatrix} m(x^1) \\ \vdots \\ m(x^n) \end{bmatrix}, \begin{bmatrix} k(x^1, x^1) & \cdots & k(x^1, x^n) \\ \vdots & \ddots & \vdots \\ k(x^n, x^1) & \cdots & k(x^n, x^n) \end{bmatrix} \right) \quad \begin{array}{l} m(\cdot) : \text{mean function} \\ k(\cdot, \cdot) : \text{kernel function} \end{array}$$

$$m(\mathbf{x}) = \mathbf{0} \quad k(\mathbf{x}, \mathbf{x}') = \gamma \exp \left( -\frac{1}{2} (\mathbf{x} - \mathbf{x}')^T \text{diag}(\boldsymbol{\lambda})^{-2} (\mathbf{x} - \mathbf{x}') \right)$$

$$\text{cov}(f^i, f^j) = E[(f^i - m(x^i))(f^j - m(x^j))] \approx k(\mathbf{x}^i, \mathbf{x}^j)$$

Exponential Square kernel function:  $k(\mathbf{x}^i, \mathbf{x}^j) = \gamma \exp \left( -\frac{1}{2} (\mathbf{x}^i - \mathbf{x}^j)^T \text{diag}(\boldsymbol{\lambda})^{-2} (\mathbf{x}^i - \mathbf{x}^j) \right)$

## Kernel function?

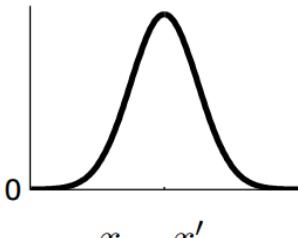
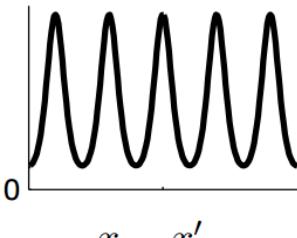
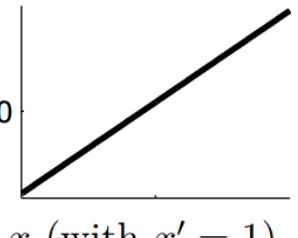
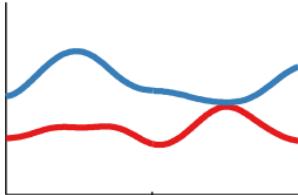
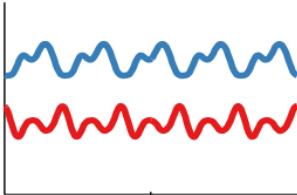
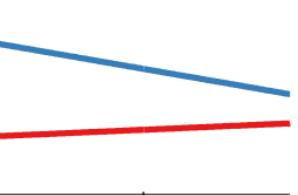
Kernel name:	Squared-exp (SE)	Periodic (Per)	Linear (Lin)
$k(x, x') =$	$\sigma_f^2 \exp\left(-\frac{(x-x')^2}{2\ell^2}\right)$	$\sigma_f^2 \exp\left(-\frac{2}{\ell^2} \sin^2\left(\pi \frac{x-x'}{p}\right)\right)$	$\sigma_f^2(x - c)(x' - c)$
Plot of $k(x, x')$ :			
Functions $f(x)$ sampled from GP prior:			
Type of structure:	local variation	repeating structure	linear functions

Figure form <http://www.cs.toronto.edu/~duvenaud/>

## Bayesian Optimization

### Learning phase : Gaussian Process (GP) regression

Construct the distribution on unknown target value  $f = f(\mathbf{x})$  corresponding  $\mathbf{x}$

1. Given the data at  $n$ th iteration

Inputs	Latent function values	$y^i = f^i + \epsilon^i$ Observations
$x^{1:n} = \{x^1, \dots, x^i, \dots, x^n\}$	$f^{1:n} = \{f^1, \dots, f^i, \dots, f^n\}$	$y^{1:n} = \{y^1, \dots, y^i, \dots, y^n\}$

2. Prior on the function values  $f^{1:n}$  is represented as Gaussian Process (GP)

$$p(f^{1:n}) = GP(m(\cdot), k(\cdot, \cdot)) : \begin{bmatrix} f^1 \\ \vdots \\ f^n \end{bmatrix} \sim N \left( \begin{bmatrix} m(x^1) \\ \vdots \\ m(x^n) \end{bmatrix}, \begin{bmatrix} k(x^1, x^1) & \cdots & k(x^1, x^n) \\ \vdots & \ddots & \vdots \\ k(x^n, x^1) & \cdots & k(x^n, x^n) \end{bmatrix} \right) \quad \begin{array}{l} m(\cdot) : \text{mean function} \\ k(\cdot, \cdot) : \text{kernel function} \end{array}$$

$$m(\mathbf{x}) = \mathbf{0} \quad k(\mathbf{x}, \mathbf{x}') = \gamma \exp \left( -\frac{1}{2} (\mathbf{x} - \mathbf{x}')^T \text{diag}(\boldsymbol{\lambda})^{-2} (\mathbf{x} - \mathbf{x}') \right)$$

3. Likelihood is constructed base the assumption on the noise, i.e., i.i.d. Gaussian noise

$$p(y^{1:n} | f^{1:n}) = N(f^{1:n}, \sigma_\epsilon^2 \mathbf{I})$$

The hyper-parameters  $\boldsymbol{\theta} = (\sigma_\epsilon, \sigma_s, \boldsymbol{\lambda})$  for the noise model and the kernel function are determined as ones maximizing the marginal log-likelihood of the training data  $\mathbf{D}^n = \{(\mathbf{x}^i, y^i) | i = 1, \dots, n\}$  as

$$\begin{aligned} \boldsymbol{\theta}^* &= \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \log p(y^{1:n} | \boldsymbol{\theta}) \\ &= \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \left( -\frac{1}{2} (\mathbf{y}^{1:n})^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n} - \frac{1}{2} \log |\mathbf{K} + \sigma_\epsilon^2 \mathbf{I}| - \frac{n}{2} \log 2\pi \right) \end{aligned}$$

## Bayesian Optimization

### Learning phase : Gaussian Process (GP) regression

Construct the distribution on unknown target value  $f = f(\mathbf{x})$  corresponding  $\mathbf{x}$

1. Given the data at  $n$ th iteration

Inputs	Latent function values	$y^i = f^i + \epsilon^i$ Observations
$x^{1:n} = \{x^1, \dots, x^i, \dots, x^n\}$	$f^{1:n} = \{f^1, \dots, f^i, \dots, f^n\}$	$y^{1:n} = \{y^1, \dots, y^i, \dots, y^n\}$

2. Prior on the function values  $f^{1:n}$  is represented as Gaussian Process (GP)

$$p(f^{1:n}) = GP(m(\cdot), k(\cdot, \cdot)) : \begin{bmatrix} f^1 \\ \vdots \\ f^n \end{bmatrix} \sim N \left( \begin{bmatrix} m(x^1) \\ \vdots \\ m(x^n) \end{bmatrix}, \begin{bmatrix} k(x^1, x^1) & \cdots & k(x^1, x^n) \\ \vdots & \ddots & \vdots \\ k(x^n, x^1) & \cdots & k(x^n, x^n) \end{bmatrix} \right) \quad \begin{array}{l} m(\cdot) : \text{mean function} \\ k(\cdot, \cdot) : \text{kernel function} \end{array}$$

$$m(x) = \mathbf{0} \quad k(x, x') = \gamma \exp \left( -\frac{1}{2} (x - x')^T \text{diag}(\lambda)^{-2} (x - x') \right)$$

3. Likelihood is constructed base the assumption on the noise, i.e., i.i.d. Gaussian noise

$$p(y^{1:n} | f^{1:n}) = N(f^{1:n}, \sigma_\epsilon^2 \mathbf{I})$$

4. Joint distribution based on Bayes' rule :

$$p(f, y^{1:n}) = \int p(f, f^{1:n}) p(y^{1:n} | f^{1:n}) df^{1:n} \quad \rightarrow \quad \begin{bmatrix} y^{1:n} \\ f \end{bmatrix} \sim N \left( \begin{bmatrix} \mathbf{0} \\ \mathbf{k} \end{bmatrix}, \begin{bmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k} \\ \mathbf{k}^T & k(x, x) \end{bmatrix} \right)$$

## Bayesian Optimization

### Learning phase : Gaussian Process (GP) regression

Construct the distribution on unknown target value  $f = f(\mathbf{x})$  corresponding  $\mathbf{x}$

**Property 4 :** Conditionals of a GRV are Gaussians, more specifically, if

$$Z = \begin{bmatrix} Y_1 \\ - \\ Y_2 \end{bmatrix} \sim N \left( \begin{bmatrix} Y_1 \\ - \\ Y_2 \end{bmatrix}, \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \hline \Sigma_{21} & \Sigma_{22} \end{bmatrix} \right)$$

where  $Y_1$  is  $k$ -dim RV and  $Y_2$  is an  $n - k$  dim RV, then

$$Y_2 | \{Y_1 = y\} \sim N(\Sigma_{21}\Sigma_{11}^{-1}(y - \mu_1) + \mu_2, \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12})$$

4. Joint distribution based on Bayes' rule :

$$p(f, \mathbf{y}^{1:n}) = \int p(f, \mathbf{f}^{1:n})p(\mathbf{y}^{1:n}|\mathbf{f}^{1:n}) d\mathbf{f}^{1:n} \quad \rightarrow \quad \begin{bmatrix} \mathbf{y}^{1:n} \\ f \end{bmatrix} \sim N \left( \mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k} \\ \mathbf{k}^T & k(\mathbf{x}, \mathbf{x}) \end{bmatrix} \right)$$

5. Conditionalization → Posterior distribution on the function value  $f = f(\mathbf{x})$  for  $\mathbf{x}$  given  $\mathcal{D}^n = \{\mathbf{x}^{1:n}, \mathbf{y}^{1:n}\}$

$$p(f|\mathcal{D}^n) = N(\mu(\mathbf{x}|\mathcal{D}^n), \sigma^2(\mathbf{x}|\mathcal{D}^n))$$

**Mean** :  $\mu(\mathbf{x}|\mathcal{D}^n) = \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$

**Variance** :  $\sigma^2(\mathbf{x}|\mathcal{D}^n) = k(\mathbf{x}, \mathbf{x}) - \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$

## Bayesian Optimization

### Learning phase : Gaussian Process (GP) regression

Construct the distribution on unknown target value  $f = f(x)$  corresponding  $x$

1. Given the data at  $n$ th iteration

Inputs	Latent function values	$y^i = f^i + \epsilon^i$ Observations
$x^{1:n} = \{x^1, \dots, x^i, \dots, x^n\}$	$f^{1:n} = \{f^1, \dots, f^i, \dots, f^n\}$	$y^{1:n} = \{y^1, \dots, y^i, \dots, y^n\}$

2. Prior on the function values  $f^{1:n}$  is represented as Gaussian Process (GP)

$$p(f^{1:n}) = GP(m(\cdot), k(\cdot, \cdot)) : \begin{bmatrix} f^1 \\ \vdots \\ f^n \end{bmatrix} \sim N \left( \begin{bmatrix} m(x^1) \\ \vdots \\ m(x^n) \end{bmatrix}, \begin{bmatrix} k(x^1, x^1) & \cdots & k(x^1, x^n) \\ \vdots & \ddots & \vdots \\ k(x^n, x^1) & \cdots & k(x^n, x^n) \end{bmatrix} \right)$$

$m(\cdot)$  : mean function  
 $k(\cdot, \cdot)$  : kernel function

$$m(x) = \mathbf{0} \quad k(x, x') = \gamma \exp \left( -\frac{1}{2} (x - x')^T \text{diag}(\lambda)^{-2} (x - x') \right)$$

3. Likelihood is constructed base the assumption on the noise, i.e., i.i.d. Gaussian noise

$$p(y^{1:n} | f^{1:n}) = N(f^{1:n}, \sigma_\epsilon^2 \mathbf{I})$$

4. Joint distribution based on Bayes' rule :

$$p(f, y^{1:n}) = \int p(f, f^{1:n}) p(y^{1:n} | f^{1:n}) df^{1:n} \quad \rightarrow \quad \begin{bmatrix} y^{1:n} \\ f \end{bmatrix} \sim N \left( \begin{bmatrix} \mathbf{0} \\ \mathbf{k} \end{bmatrix}, \begin{bmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k} \\ \mathbf{k}^T & k(x, x) \end{bmatrix} \right)$$

5. Conditionalization → Posterior distribution on the function value  $f = f(x)$  for  $x$  given  $D^n = \{x^{1:n}, y^{1:n}\}$

$$p(f | D^n) = N(\mu(x | D^n), \sigma^2(x | D^n))$$

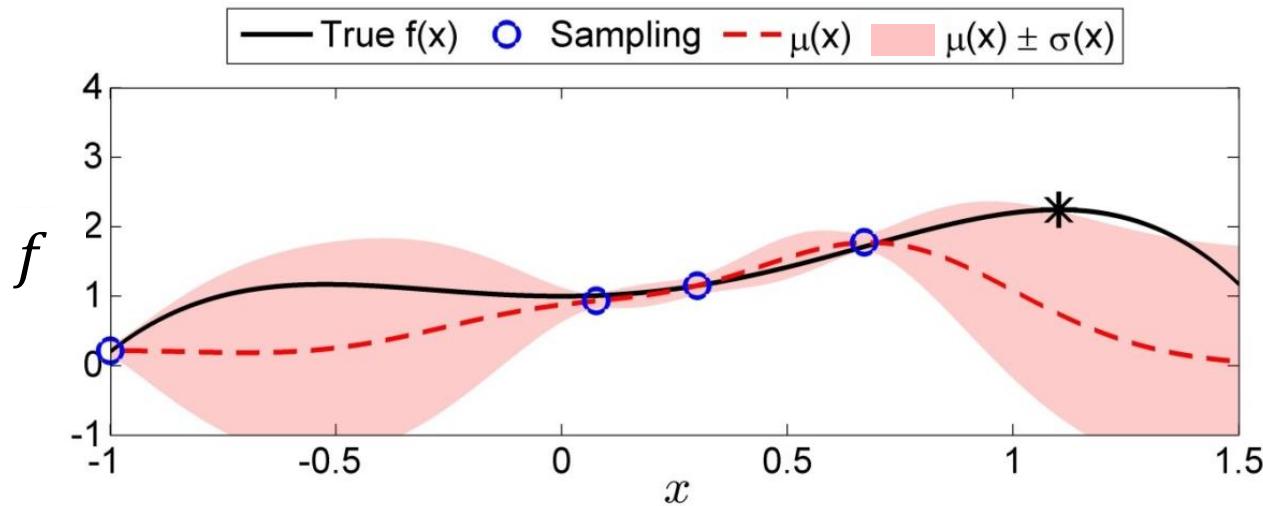
Mean :  $\mu(x | D^n) = \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$

Variance :  $\sigma^2(x | D^n) = k(x, x) - \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$

### Learning phase : Gaussian Process (GP) regression

Construct the distribution on unknown target value  $f = f(x)$  corresponding  $x$

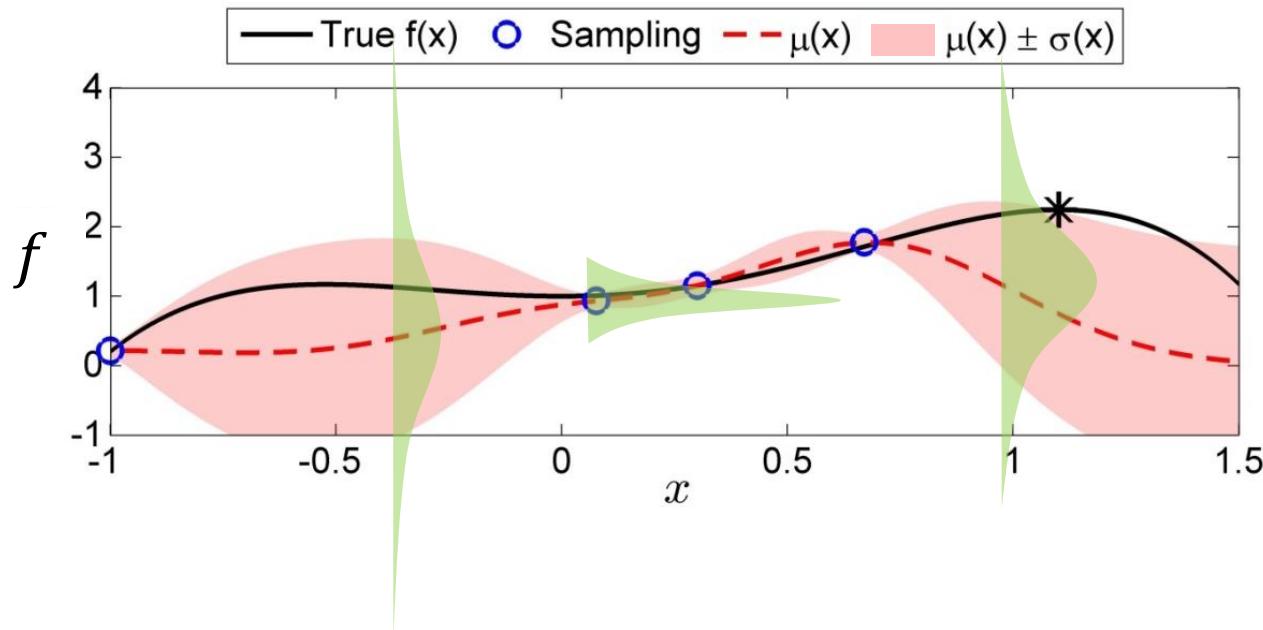
$$p(f|\mathcal{D}^n) = N(\mu(x|\mathcal{D}^n), \sigma^2(x|\mathcal{D}^n))$$



### Learning phase : Gaussian Process (GP) regression

Construct the distribution on unknown target value  $f = f(x)$  corresponding  $x$

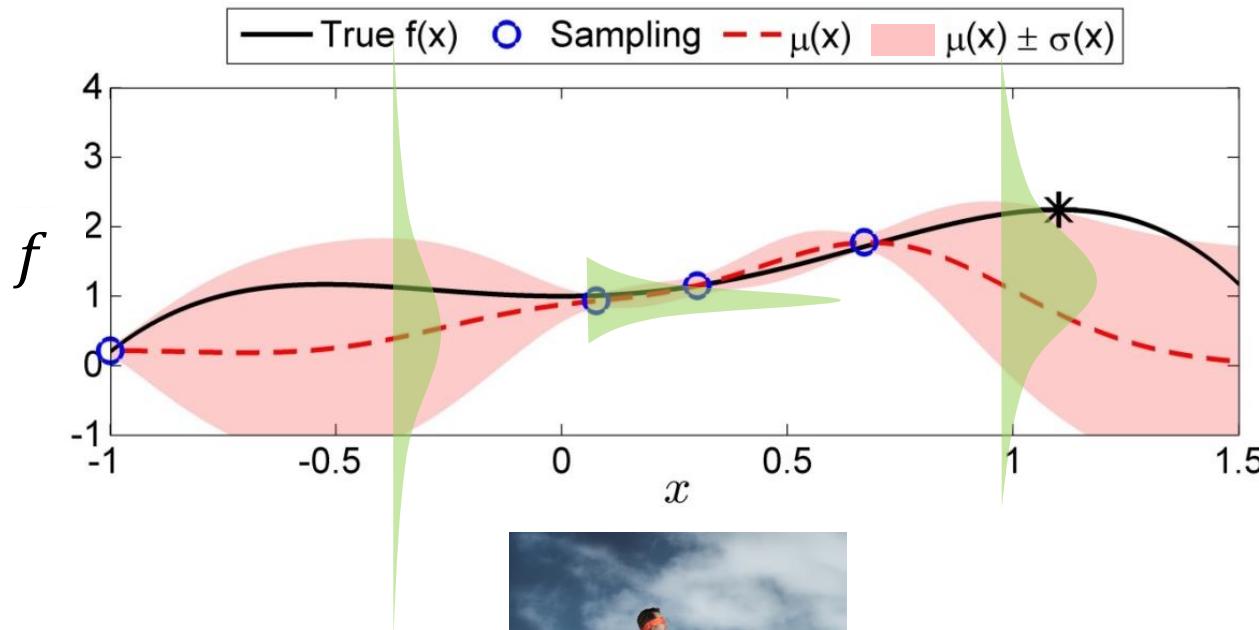
$$p(f|\mathcal{D}^n) = N(\mu(x|\mathcal{D}^n), \sigma^2(x|\mathcal{D}^n))$$



### Optimization (Sampling) phase

How to select the next input ?

- Learn target function (exploration)
- Improve target value (exploitation)



Go to a graduate school to explore  
my intellectual capability?

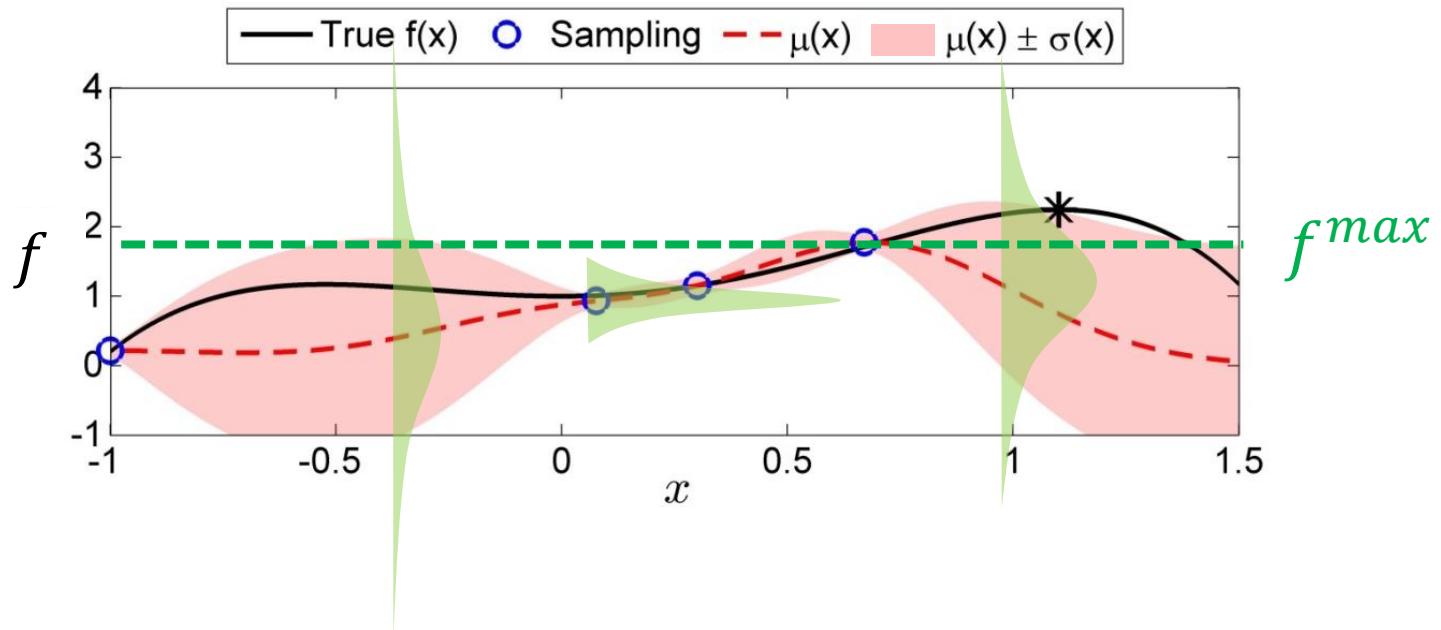


Go to a company to make money?

## Optimization (Sampling) phase

How to select the next input ?

- Learn target function (exploration)
- Improve target value (exploitation)



Next sampling point  $x^{n+1}$  is determined by solving (Mockus *et al*, 1978)

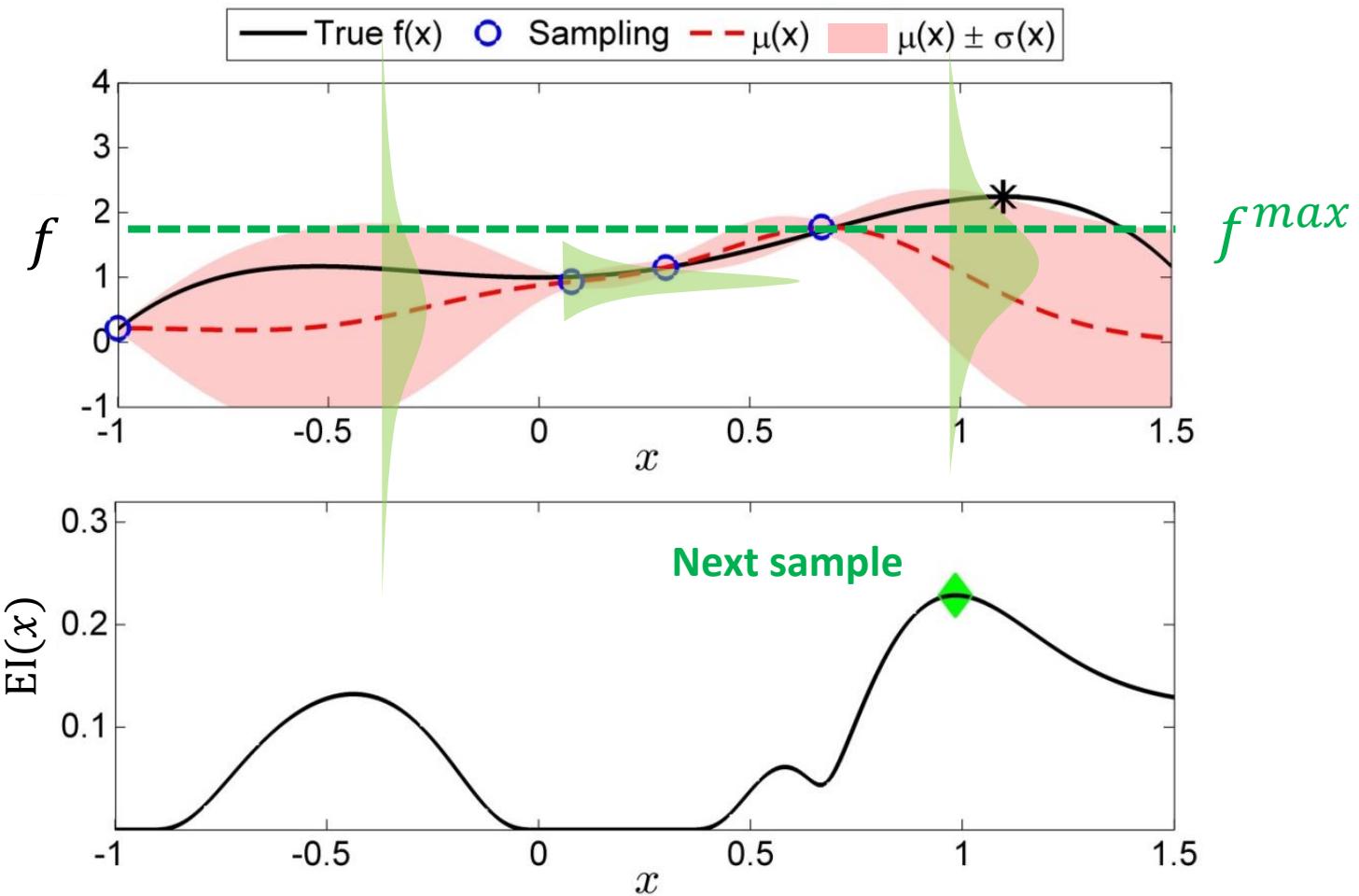
$$x^{n+1} = \arg \max_x \text{EI}(x) \triangleq E[\max\{0, f - f^{max}\} | \mathcal{D}^n]$$

*DIRECT* (Finkel, 2003), a gradient free optimization code, is used to solve this problem

## Optimization (Sampling) phase

How to select the next input ?

- Learn target function (exploration)
- Improve target value (exploitation)



# Bayesian Optimization

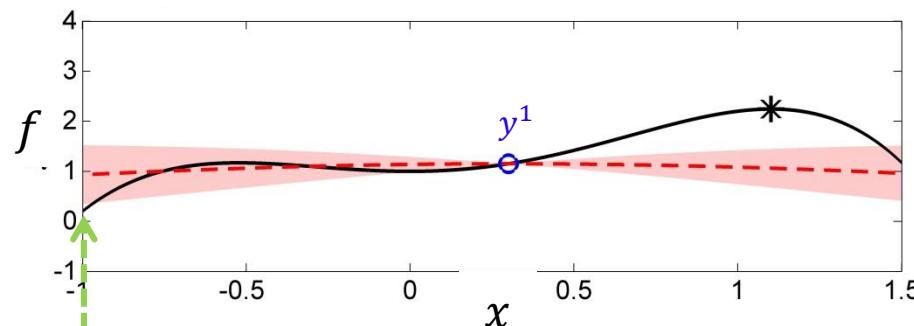
## Illustrative example

$$\underset{x}{\text{maximize}} \quad f(x) = -1.3x^4 + 1x^3 + 1.5x^2 + 1 + \epsilon$$

subject to  $-1 \leq x \leq 1.5$

$$\epsilon \sim N(0, 0.01^2)$$

$$\begin{matrix} \color{blue}{y^1} \\ \color{blue}{f} \end{matrix} \begin{bmatrix} 1.15 \\ f \end{bmatrix} \sim N \left( \mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k} \\ \mathbf{k}^T & k(\mathbf{x}, \mathbf{x}) \end{bmatrix} \right)$$



**Construct probability distribution on the unknown function value**

$$p(f|\mathcal{D}^{1:n}) = N(\mu(x|\mathcal{D}^{1:n}), \sigma^2(x|\mathcal{D}^{1:n}))$$

$$\mu(x|\mathcal{D}^{1:n}) = \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$$

$$\sigma^2(x|\mathcal{D}^{1:n}) = k(x, x) - \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$$

**Select the next trial action that maximizes**

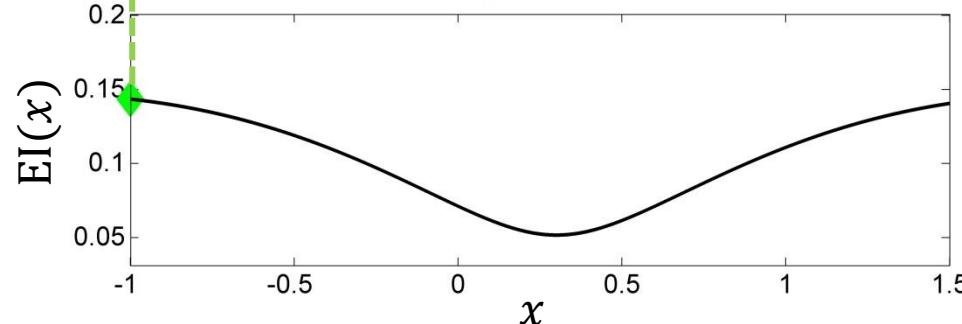
$$x^2 = \arg \max_x \text{EI}(x) \triangleq E[\max\{0, f - f^{\max}\} | \mathcal{D}^{1:n}]$$

$$x^2 = -1.00$$

**Query the function value**

$$y^2 = f(x^2) + \epsilon, \epsilon \sim N(0, 0.01^2)$$

$$y^2 = 0.18$$



# Bayesian Optimization

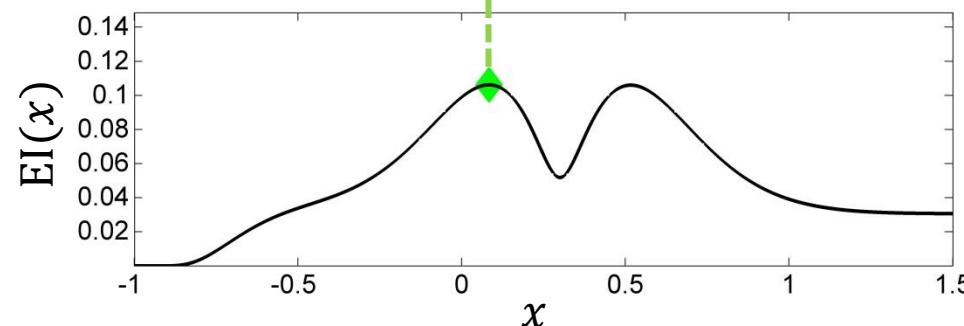
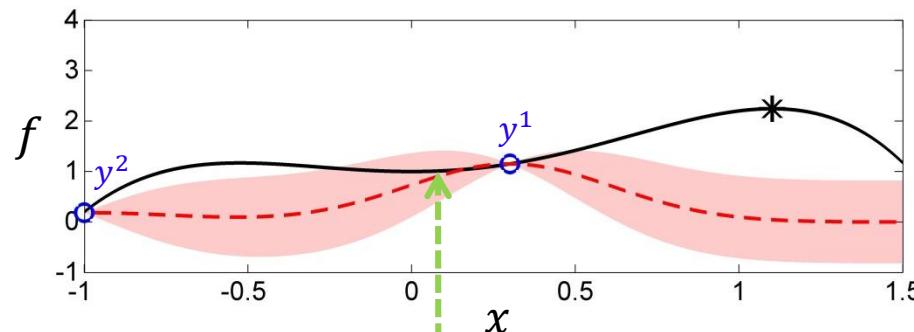
## Illustrative example

$$\underset{x}{\text{maximize}} \ f(x) = -1.3x^4 + 1x^3 + 1.5x^2 + 1 + \epsilon$$

subject to  $-1 \leq x \leq 1.5$

$$\epsilon \sim N(0, 0.01^2)$$

$$\begin{matrix} \color{blue}{y^1} \\ \color{blue}{y^2} \\ \color{blue}{f} \end{matrix} \begin{bmatrix} 1.15 \\ 0.18 \end{bmatrix} \sim N \left( \mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k} \\ \mathbf{k}^T & k(\mathbf{x}, \mathbf{x}) \end{bmatrix} \right)$$



**Construct probability distribution on the unknown function value**

$$p(f|\mathcal{D}^{1:n}) = N(\mu(x|\mathcal{D}^{1:n}), \sigma^2(x|\mathcal{D}^{1:n}))$$

$$\mu(x|\mathcal{D}^{1:n}) = \mathbf{k}^T(\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$$

$$\sigma^2(x|\mathcal{D}^{1:n}) = k(x, x) - \mathbf{k}^T(\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$$

**Select the next trial action that maximizes**

$$x^3 = \arg \max_x \text{EI}(x) \triangleq E[\max\{0, f - f^{\max}\} | \mathcal{D}^{1:n}]$$

$$x^3 = 0.03$$

**Query the function value**

$$x^3 = f(x^3) + \epsilon, \epsilon \sim N(0, 0.01^2)$$

$$x^3 = 1.02$$

# Bayesian Optimization

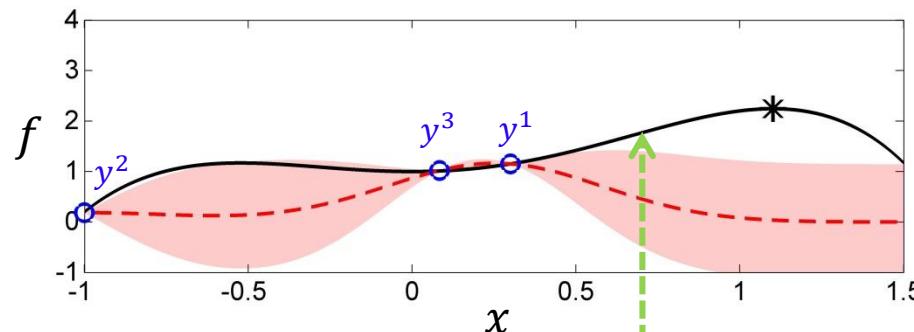
## Illustrative example

$$\underset{x}{\text{maximize}} \ f(x) = -1.3x^4 + 1x^3 + 1.5x^2 + 1 + \epsilon$$

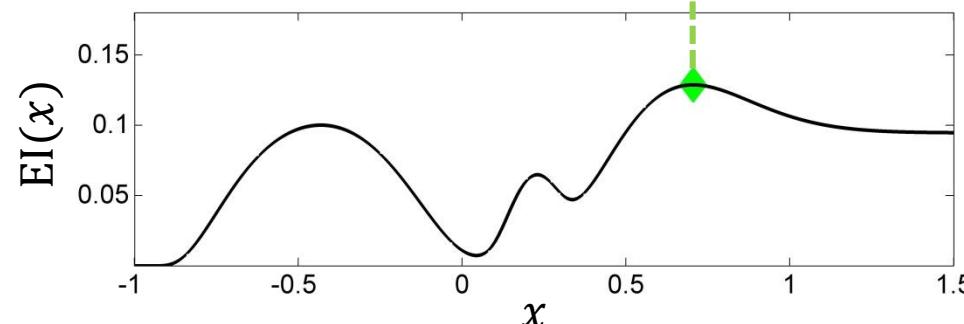
$$\text{subject to} \quad -1 \leq x \leq 1.5$$

$$\epsilon \sim N(0, 0.01^2)$$

$$\begin{bmatrix} y^1 \\ y^2 \\ y^3 \\ f \end{bmatrix} \sim N\left(\mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k} \\ \mathbf{k}^T & k(x, x) \end{bmatrix}\right)$$



Next sample



Construct probability distribution on the unknown function value

$$p(f|\mathcal{D}^{1:n}) = N(\mu(x|\mathcal{D}^{1:n}), \sigma^2(x|\mathcal{D}^{1:n}))$$

$$\mu(x|\mathcal{D}^{1:n}) = \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$$

$$\sigma^2(x|\mathcal{D}^{1:n}) = k(x, x) - \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$$

Select the next trial action that maximizes

$$x^4 = \arg \max_x \text{EI}(x) \triangleq E[\max\{0, f - f^{\max}\} | \mathcal{D}^{1:n}]$$

$$x^4 = 0.70$$

Query the function value

$$y^4 = f(x^4) + \epsilon, \epsilon \sim N(0, 0.01^2)$$

$$y^4 = 1.76$$

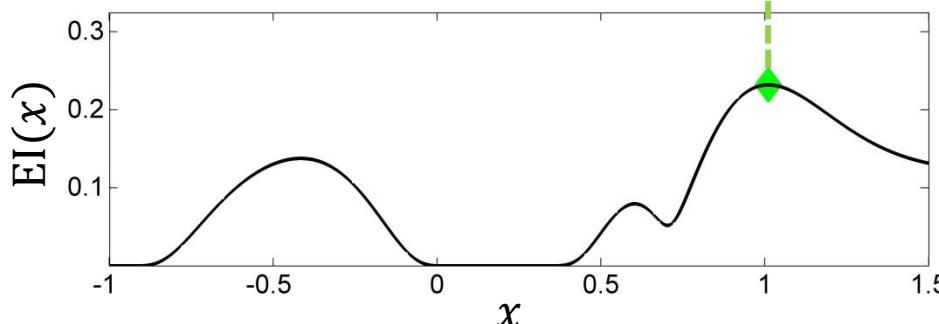
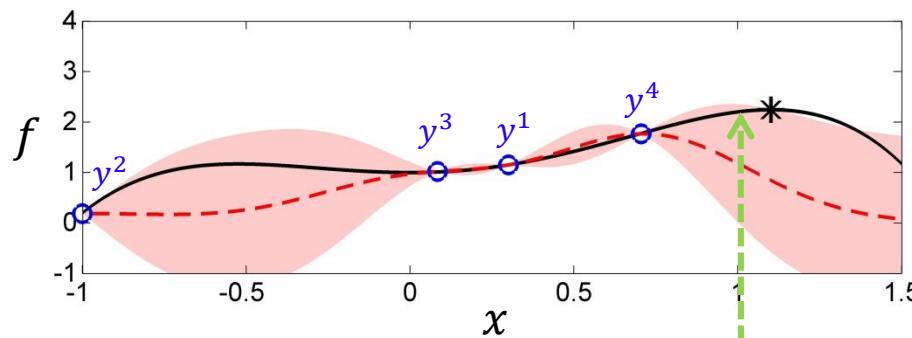
# Bayesian Optimization

## Illustrative example

$$\begin{aligned} & \text{maximize}_x f(x) = -1.3x^4 + 1x^3 + 1.5x^2 + 1 + \epsilon \\ & \text{subject to } -1 \leq x \leq 1.5 \quad \epsilon \sim N(0, 0.01^2) \end{aligned}$$

$$\begin{bmatrix} y^1 \\ y^2 \\ y^3 \\ y^4 \\ f \end{bmatrix} \sim N\left(\mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k} \\ \mathbf{k}^T & k(x, x) \end{bmatrix}\right)$$

$$n = 4$$



**Construct probability distribution on the unknown function value**

$$p(f|\mathcal{D}^{1:n}) = N(\mu(x|\mathcal{D}^{1:n}), \sigma^2(x|\mathcal{D}^{1:n}))$$

$$\mu(x|\mathcal{D}^{1:n}) = \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$$

$$\sigma^2(x|\mathcal{D}^{1:n}) = k(x, x) - \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$$

**Select the next trial action that maximizes**

$$x^5 = \arg \max_x EI(x) \triangleq E[\max\{0, f - f^{max}\} | \mathcal{D}^{1:n}]$$

$$x^5 = 1.01$$

**Query the function value**

$$y^5 = f(x^5) + \epsilon, \epsilon \sim N(0, 0.01^2)$$

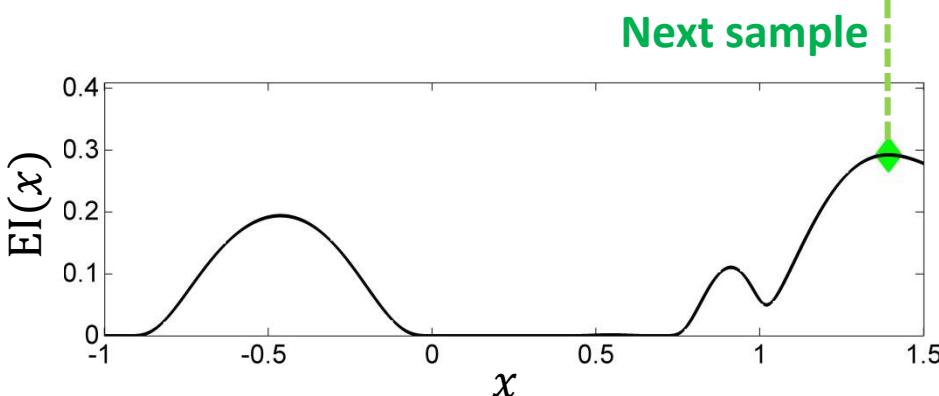
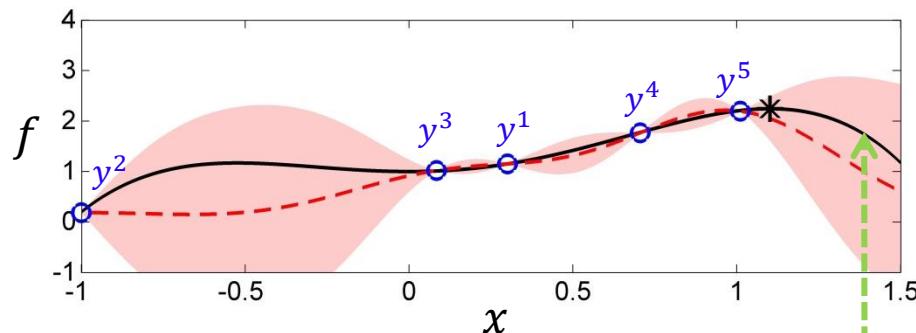
$$y^5 = 2.19$$

# Bayesian Optimization

## Illustrative example

$$\begin{aligned} & \text{maximize}_x f(x) = -1.3x^4 + 1x^3 + 1.5x^2 + 1 + \epsilon \\ & \text{subject to } -1 \leq x \leq 1.5 \quad \epsilon \sim N(0, 0.01^2) \end{aligned}$$

$$\begin{array}{c} \begin{matrix} y^1 \\ y^2 \\ y^3 \\ y^4 \\ y^5 \\ f \end{matrix} \end{array} \begin{bmatrix} 1.15 \\ 0.18 \\ 1.02 \\ 1.76 \\ 2.19 \\ f \end{bmatrix} \sim N\left(\mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k} \\ \mathbf{k}^T & k(x, x) \end{bmatrix}\right)$$



**Construct probability distribution on the unknown function value**

$$p(f|\mathcal{D}^{1:n}) = N(\mu(x|\mathcal{D}^{1:n}), \sigma^2(x|\mathcal{D}^{1:n}))$$

$$\mu(x|\mathcal{D}^{1:n}) = \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$$

$$\sigma^2(x|\mathcal{D}^{1:n}) = k(x, x) - \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$$

**Select the next trial action that maximizes**

$$x^6 = \arg \max_x EI(x) \triangleq E[\max\{0, f - f^{max}\} | \mathcal{D}^{1:n}]$$

$$x^6 = 1.39$$

**Query the function value**

$$y^6 = f(x^6) + \epsilon, \epsilon \sim N(0, 0.01^2)$$

$$y^6 = 1.70$$

# Bayesian Optimization

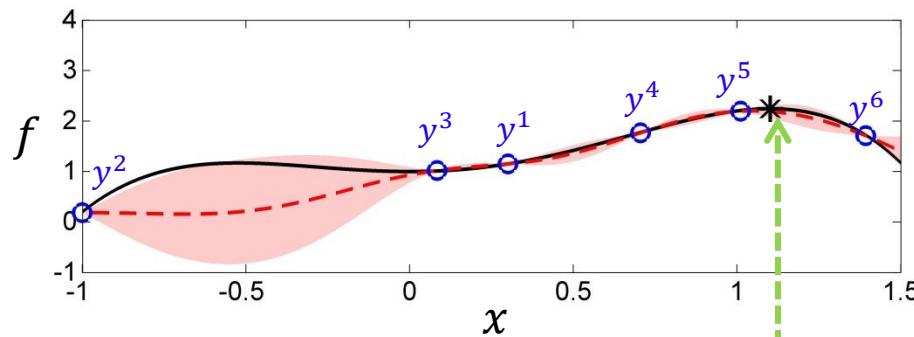
## Illustrative example

$$\underset{x}{\text{maximize}} \ f(x) = -1.3x^4 + 1x^3 + 1.5x^2 + 1 + \epsilon$$

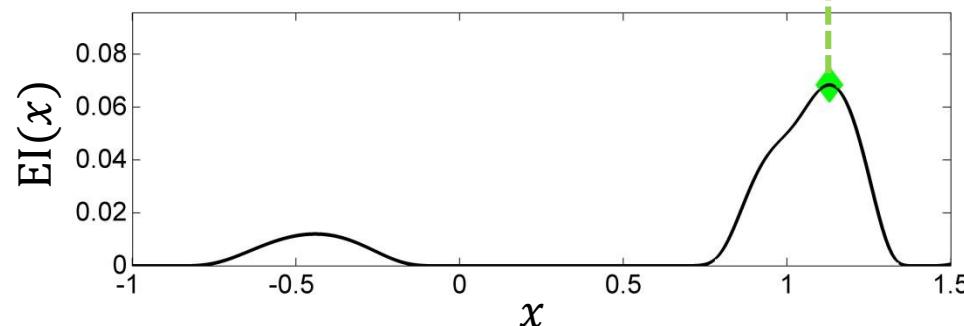
$$\text{subject to} \quad -1 \leq x \leq 1.5$$

$$\epsilon \sim N(0, 0.01^2)$$

$$\begin{bmatrix} y^1 \\ y^2 \\ y^3 \\ y^4 \\ y^5 \\ y^6 \\ f \end{bmatrix} \sim N \left( \mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k} \\ \mathbf{k}^T & k(x, x) \end{bmatrix} \right)$$



Next sample



Construct probability distribution on the unknown function value

$$p(f|\mathcal{D}^{1:n}) = N(\mu(x|\mathcal{D}^{1:n}), \sigma^2(x|\mathcal{D}^{1:n}))$$

$$\mu(x|\mathcal{D}^{1:n}) = \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$$

$$\sigma^2(x|\mathcal{D}^{1:n}) = k(x, x) - \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$$

Select the next trial action that maximizes

$$x^7 = \arg \max_x EI(x) \triangleq E[\max\{0, f - f^{\max}\} | \mathcal{D}^{1:n}]$$

$$x^7 = 1.13$$

Query the function value

$$y^7 = f(x^7) + \epsilon, \epsilon \sim N(0, 0.01^2)$$

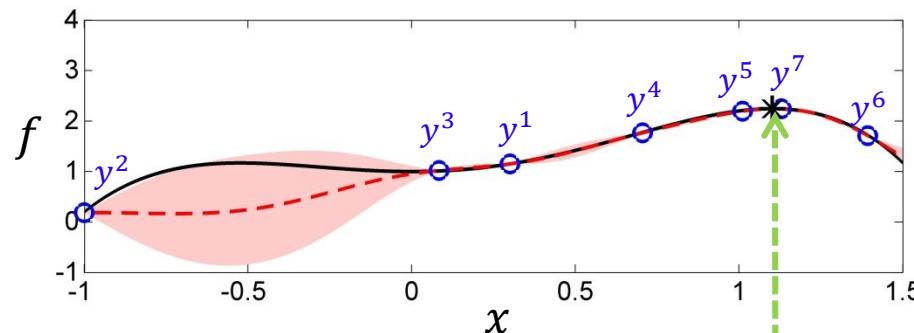
$$y^7 = 2.24$$

# Bayesian Optimization

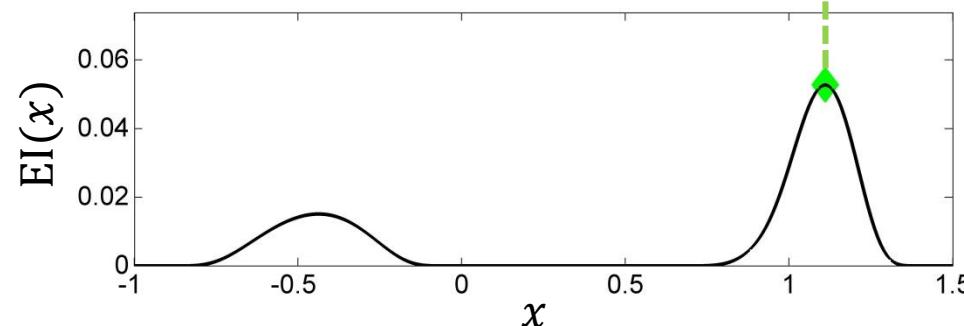
## Illustrative example

$$\begin{aligned} & \text{maximize}_x f(x) = -1.3x^4 + 1x^3 + 1.5x^2 + 1 + \epsilon \\ & \text{subject to } -1 \leq x \leq 1.5 \quad \epsilon \sim N(0, 0.01^2) \end{aligned}$$

$$\begin{bmatrix} y^1 \\ y^2 \\ y^3 \\ y^4 \\ y^5 \\ y^6 \\ y^7 \\ f \end{bmatrix} \sim N \left( \mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k}^T \\ \mathbf{k} & k(x, x) \end{bmatrix} \right)$$



Next sample



Construct probability distribution on the unknown function value

$$p(f|\mathcal{D}^{1:n}) = N(\mu(x|\mathcal{D}^{1:n}), \sigma^2(x|\mathcal{D}^{1:n}))$$

$$\mu(x|\mathcal{D}^{1:n}) = \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$$

$$\sigma^2(x|\mathcal{D}^{1:n}) = k(x, x) - \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$$

Select the next trial action that maximizes

$$x^8 = \arg \max_x EI(x) \triangleq E[\max\{0, f - f^{max}\} | \mathcal{D}^{1:n}]$$

$$x^8 = 1.11$$

Query the function value

$$y^8 = f(x^8) + \epsilon, \epsilon \sim N(0, 0.01^2)$$

$$y^8 = 2.24$$

# Bayesian Optimization

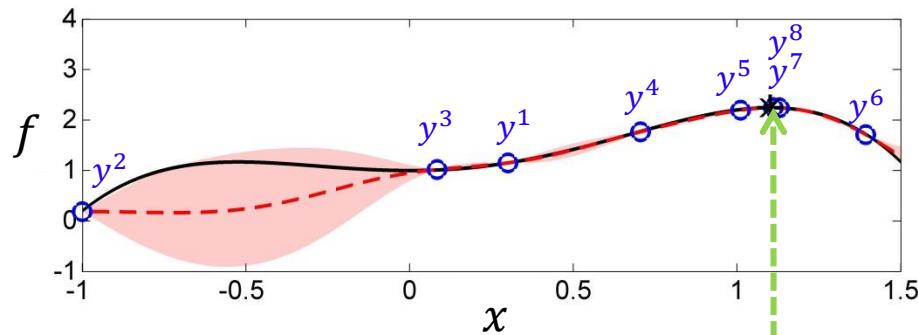
## Illustrative example

$$\underset{x}{\text{maximize}} \ f(x) = -1.3x^4 + 1x^3 + 1.5x^2 + 1 + \epsilon$$

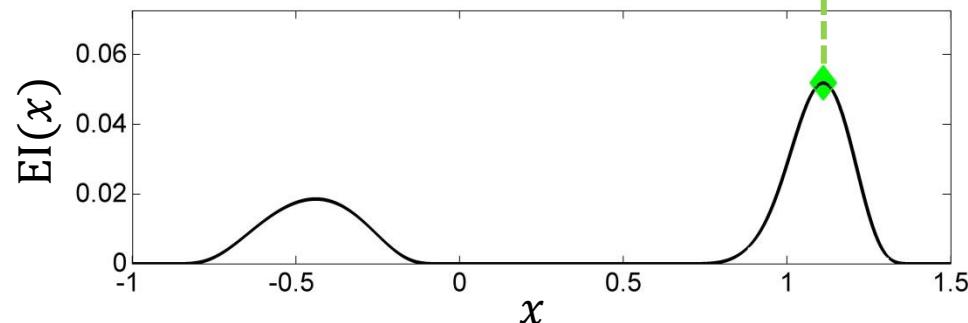
$$\text{subject to} \quad -1 \leq x \leq 1.5$$

$$\epsilon \sim N(0, 0.01^2)$$

$$\begin{bmatrix} y^1 \\ y^2 \\ y^3 \\ y^4 \\ y^5 \\ y^6 \\ y^7 \\ y^8 \\ f \end{bmatrix} \sim N \left( \mathbf{0}, \begin{pmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k}^T \\ \mathbf{k} & k(x, x) \end{pmatrix}^k \right)$$



Next sample



Construct probability distribution on the unknown function value

$$p(f|\mathcal{D}^{1:n}) = N(\mu(x|\mathcal{D}^{1:n}), \sigma^2(x|\mathcal{D}^{1:n}))$$

$$\mu(x|\mathcal{D}^{1:n}) = \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$$

$$\sigma^2(x|\mathcal{D}^{1:n}) = k(x, x) - \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$$

Select the next trial action that maximizes

$$x^9 = \arg \max_x \text{EI}(x) \triangleq E[\max\{0, f - f^{\max}\} | \mathcal{D}^{1:n}]$$

$$x^9 = 1.11$$

Query the function value

$$y^9 = f(x^9) + \epsilon, \epsilon \sim N(0, 0.01^2)$$

$$y^9 = 2.24$$

# Bayesian Optimization

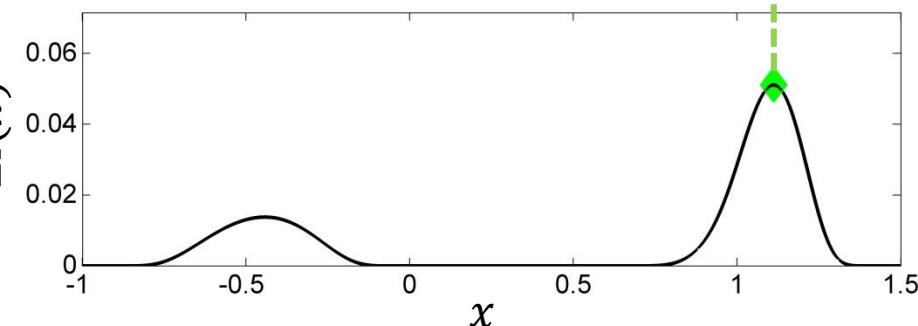
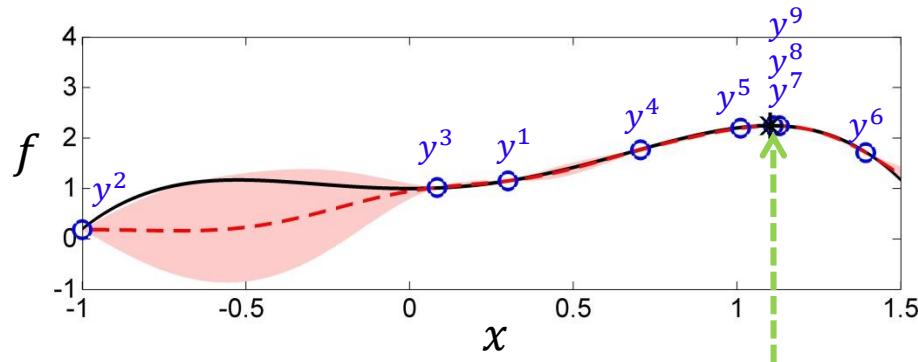
## Illustrative example

$$\underset{x}{\text{maximize}} \ f(x) = -1.3x^4 + 1x^3 + 1.5x^2 + 1 + \epsilon$$

$$\text{subject to} \quad -1 \leq x \leq 1.5$$

$$\epsilon \sim N(0, 0.01^2)$$

$$\begin{bmatrix} y^1 \\ y^2 \\ y^3 \\ y^4 \\ y^5 \\ y^6 \\ y^7 \\ y^8 \\ y^9 \\ f \end{bmatrix} \sim N \left( \mathbf{0}, \begin{pmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k}^T \\ \mathbf{k} & k(\mathbf{x}, \mathbf{x}) \end{pmatrix}^k \right)$$



**Construct probability distribution on the unknown function value**

$$p(f|\mathcal{D}^{1:n}) = N(\mu(x|\mathcal{D}^{1:n}), \sigma^2(x|\mathcal{D}^{1:n}))$$

$$\mu(x|\mathcal{D}^{1:n}) = \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$$

$$\sigma^2(x|\mathcal{D}^{1:n}) = k(x, x) - \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$$

**Select the next trial action that maximizes**

$$x^{10} = \arg \max_z EI(x) \triangleq E[\max\{0, f - f^{max}\} | \mathcal{D}^{1:n}]$$

$$x^{10} = 1.11$$

**Query the function value**

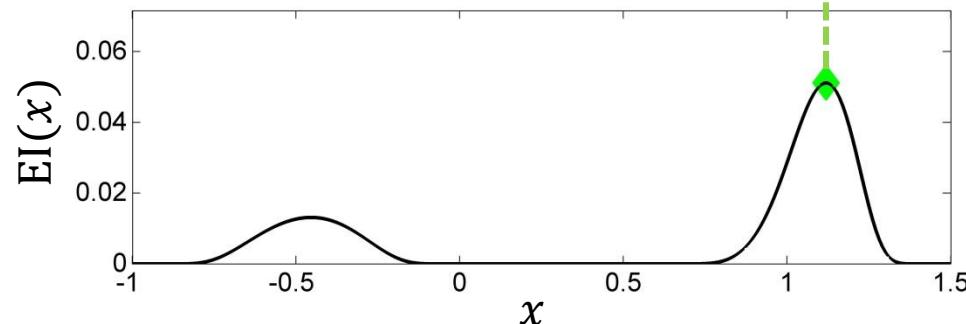
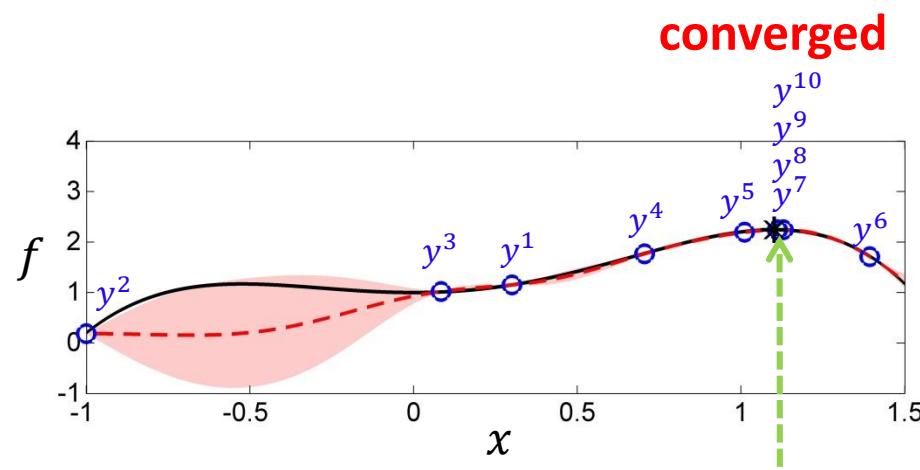
$$y^{10} = f(x^{10}) + \epsilon, \epsilon \sim N(0, 0.01^2)$$

$$y^{10} = 2.24$$

# Bayesian Optimization

## Illustrative example

$$\begin{aligned} & \text{maximize}_x f(x) = -1.3x^4 + 1x^3 + 1.5x^2 + 1 + \epsilon \\ & \text{subject to } -1 \leq x \leq 1.5 \quad \epsilon \sim N(0, 0.01^2) \end{aligned}$$



$$\begin{bmatrix} y^1 \\ y^2 \\ y^3 \\ y^4 \\ y^5 \\ y^6 \\ y^7 \\ y^8 \\ y^9 \\ y^{10} \\ f \end{bmatrix} \sim N \left( \mathbf{0}, \begin{pmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & k^T \\ k & k(x, x) \end{pmatrix} \right)$$

**Construct probability distribution on the unknown function value**

$$p(f|\mathcal{D}^{1:n}) = N(\mu(x|\mathcal{D}^{1:n}), \sigma^2(x|\mathcal{D}^{1:n}))$$

$$\mu(x|\mathcal{D}^{1:n}) = \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$$

$$\sigma^2(x|\mathcal{D}^{1:n}) = k(x, x) - \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$$

**Select the next trial action that maximizes**

$$\begin{aligned} x^{11} &= \arg \max_x \text{EI}(x) \triangleq E[\max\{0, f - f^{\max}\} | \mathcal{D}^{1:n}] \\ x^{11} &= 1.11 \end{aligned}$$

**Query the function value**

$$y^{11} = f(x^{11}) + \epsilon, \epsilon \sim N(0, 0.01^2)$$

$$y^{11} = 2.24$$

# Contextual Bandit Problem



Suggested for you

**Facebook partners with Uber for ride-hailing service via Messenger**  
Reuters - 1 hour ago  
Facebook Inc said on Wednesday it is testing a service that will allow users of its Messenger app, without leaving a conversation or downloading the ride-hailing app.

**Eye on safety, California sets rules for self-driving cars**  
The Seattle Times - 1 hour ago  
FILE - In this May 13, 2015, file photo, Google's new self-driving prototype car is presented at Google campus in Mountain View, Calif.

**The Apple iPhone 5s got a massive price cut, but is it still worth it?**  
Firstpost - 1 hour ago  
Apple launched the iPhone 5s in India back in November 2013. Two years later, its price has dropped significantly. Is it still worth it?

**Philips Hue: Just Kidding About Blocking Third-Party Bulbs**  
PC Magazine - 10 hours ago  
You win, internet. Following user backlash, Philips has decided not to block third-party lighting platform after all.

[More Technology stories](#)

1:12 am 1:12 am 1:12 am 1:12 am 1:12 am

Next Back Done Search

Tell us what you like

Tap once on the genres you like, or twice on the ones you love. Press and hold on the genres you do not like.

Got it.

Tap once on the artists you like, or twice on the ones you love. Press and hold on the artists you do not like.

R&B/Soul Blues Rock Malaysian Music Matthew Shipp Eric Dolphy Alabama Shakes Wilco

K-Pop Electronic J-Pop Dance Jennifer Lopez Calvin Harris David Guetta WALK THE MOON

Classical Jazz Alternative Hiphop/R&B World Hush

Pop Indonesian Music Hip-Hop/R&B Indie Rock Jay Chou

Reset YOU Reset YOU More artists

For You New Radio Connect My Music For You New Radio Connect My Music

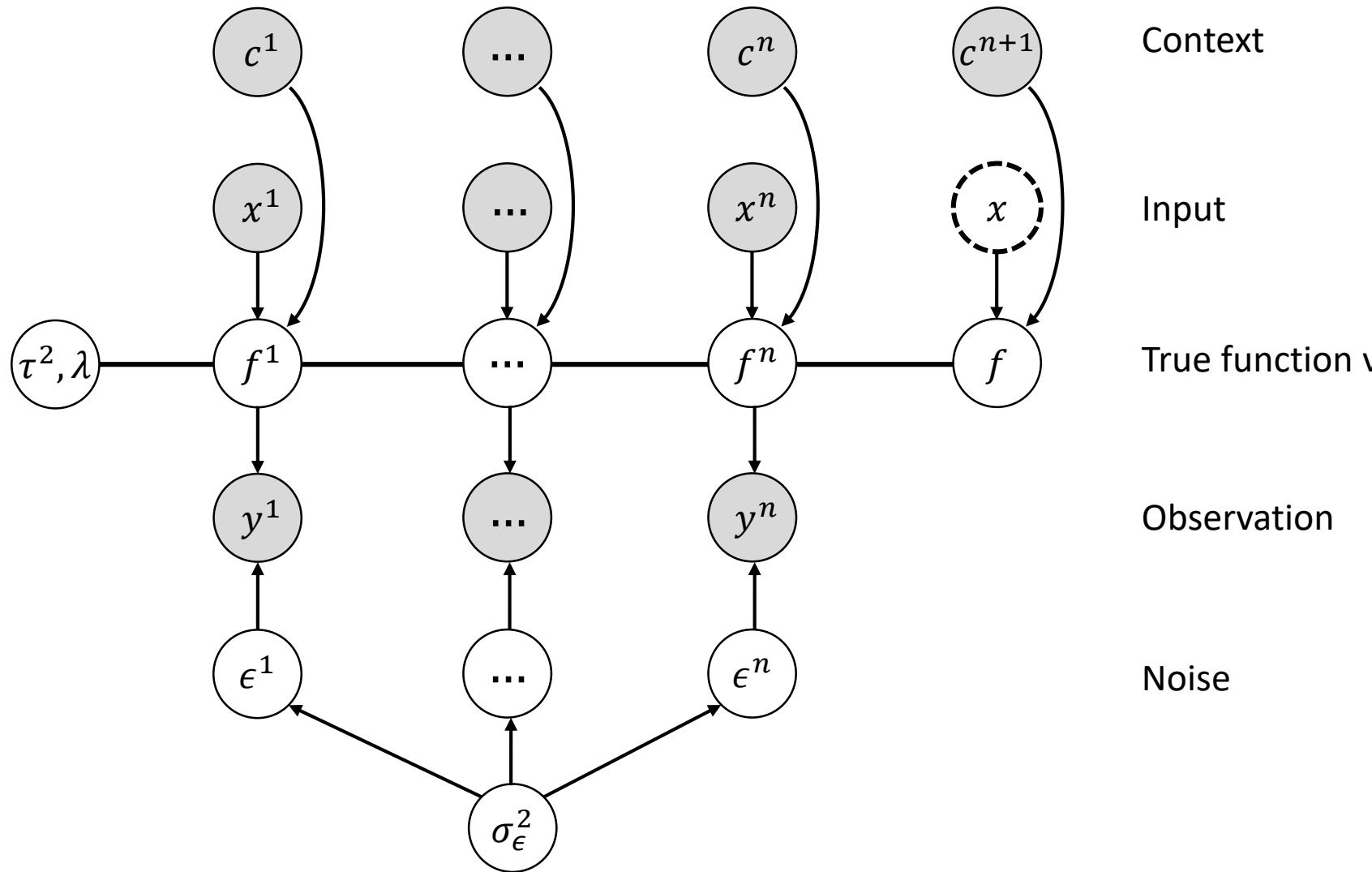


**Finance** : Portfolio optimization under unknown return profiles **given varying economic condition**

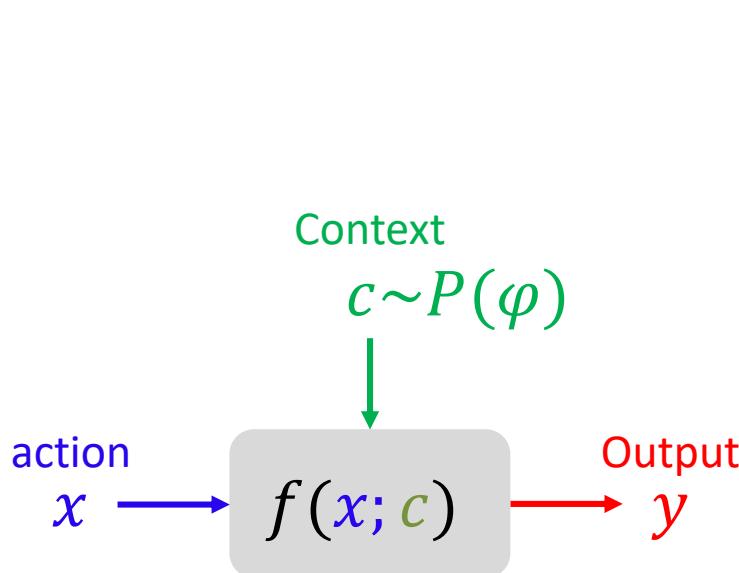
**Health care** : Choosing the best treatment among alternatives **given different patient information**

**Internet shopping** : Choosing the optimum price (sales v.s. profits) **given varying season**

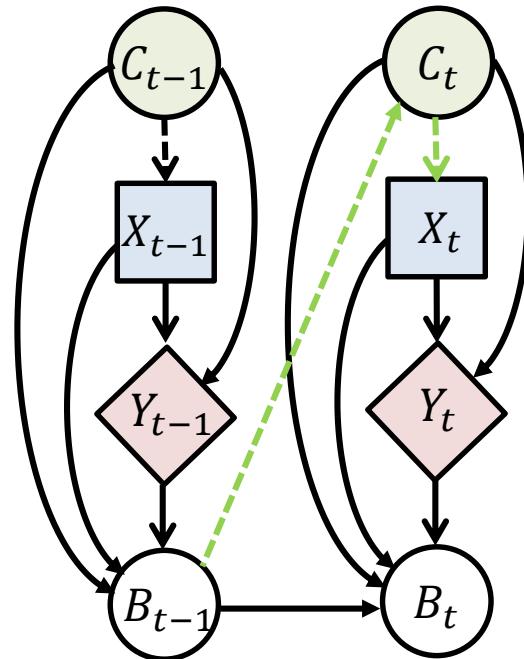
**Experiment design** : Sequential experimental design **given varying environmental condition**



# Contextual Bandit Problem



MDP over belief state



$B_t(f)$  : Belief state about unknown function  $f$  at  $t$

- Policy  $\pi$  maps all the history to new action:

$$\pi: [(c_1, x_1, y_1), (c_2, x_2, y_2), \dots, (c_{t-1}, x_{t-1}, y_{t-1}), c_t] \rightarrow x_t$$

- Find the optimal policy  $\pi^*$  that maximizes  $E[\sum_{t=1}^T y_t]$  or  $E[y_T]$

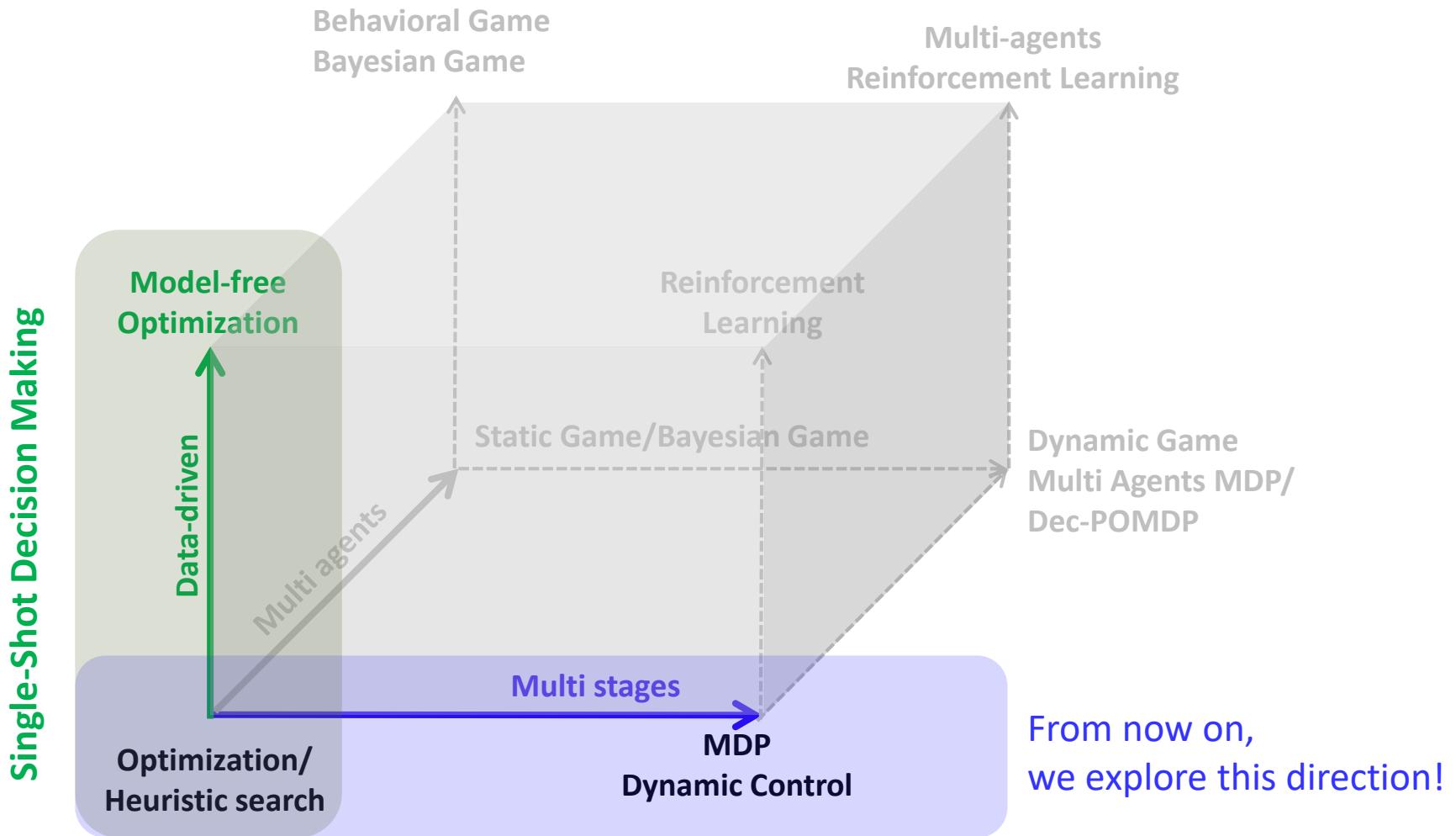
$$x^* = \pi^*(c) = \operatorname{argmax}_x f(x; c)$$

## **L13. Markov Decision Process (Formulation)**

## L10. Markov Decision Process (Formulation)

1. MDP definition
2. Cost (reward)
3. Transition probability

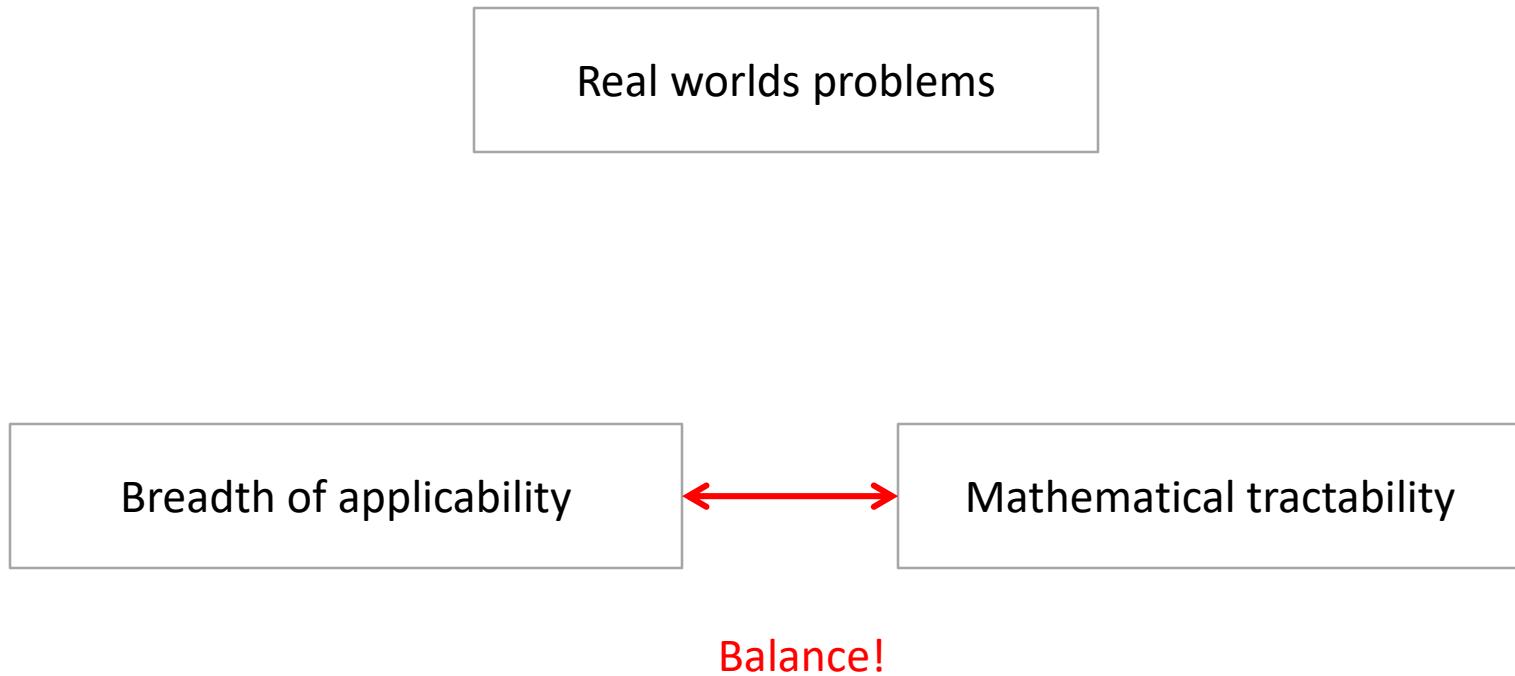
## Introduction



## Sequential Decision Making in Under Uncertainties

- Requires resonating about future *sequences* of actions and observations

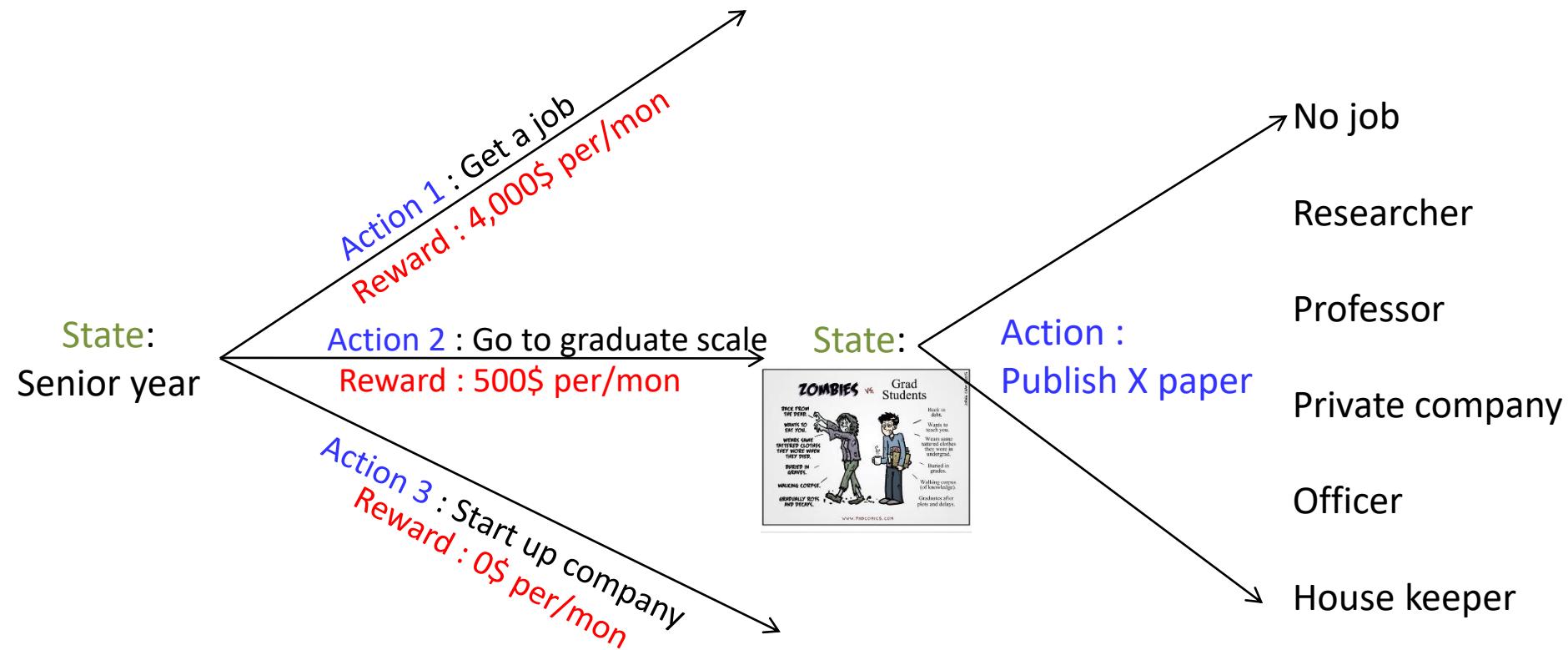
## Tension between breadth of applicability and mathematical tractability



## Introduction

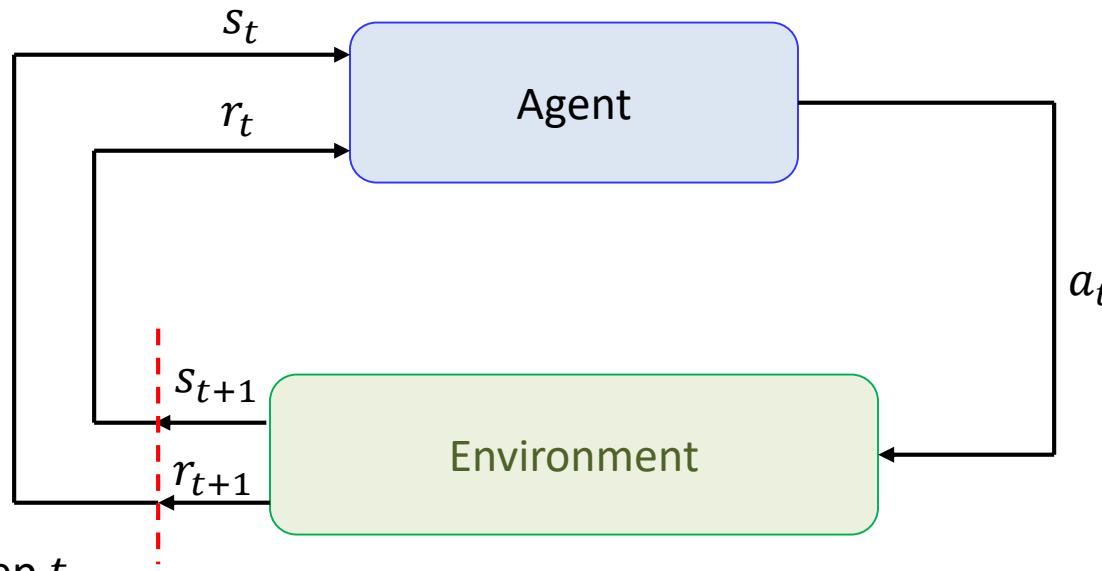
### Sequential Decision Making in Uncertainties

- Many important problems require the decision maker to make a series of decisions.
- It requires resonating about future *sequences* of actions and observations



- taking an action might lead to any one of many possible states
- how we can even hope to act optimally in the face of randomness?

## The Agent-Environment Interface



At each time step  $t$ ,

- The agent receives some representation of the environment's state  $s_t \in \mathcal{S}$  and select an action  $a_t \in \mathcal{A}(s_t)$
- One time step later, in part as a consequence of its action, the agent receives a numerical reward,  $r_{t+1} \in \mathcal{R}$  and finds itself in a new state  $s_{t+1}$

The goal is to find the optimum policy  $a_t = \pi_t(s_t)$  mapping the current state  $s_t$  to the optimum action  $a_t^*$  to maximize the total amount of reward it receives over the long run

- ✓ Find policy  $\pi_t$  using analytical MDP models → **Dynamic programming** (next week)
- ✓ Find policy  $\pi_t$  using data → **Reinforcement learning** (2 weeks later)

## The Agent-Environment Interface

- The time steps  $t = 0, 1, \dots$  need not refer to fixed interval of real time:
  - ✓ they can refer to arbitrary successive stages of decision-making and action
- The **actions** can be
  - ✓ Low-level controls, such as the voltages applied to motors of a robot arm
  - ✓ High-level decisions, such as whether or not to go graduate school
- The **states** can take a wide variety of forms
  - ✓ Can be completely determined by low-level sensations, such as sensor readings
  - ✓ Can be high-level and abstract, such as image or mental status
- The **reward** is a consequence of taking an action given a state
  - ✓ Can be specified according to the target tasks
  - ✓ Maximizing reward should result in achieving the goals of a task

In summary,

Actions can be any decisions we want to learn how to make to affect rewards, and the states can be anything we can know that might be useful in making them

## The Boundary between Agent-Environment



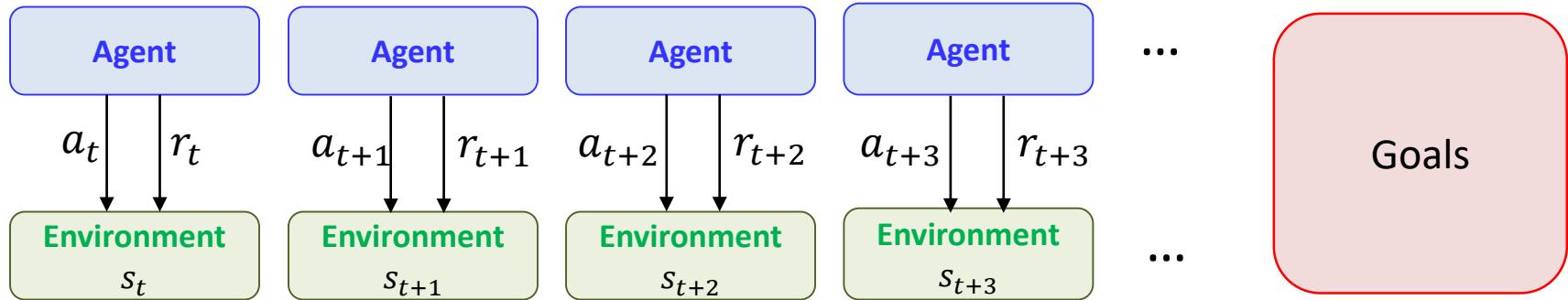
Agent :  
Decision making algorithm

Environment:  
Robot arm, motor, actuator, battery...

- Anything that cannot be changed arbitrarily by the agent is considered to be outside of the agent and thus part of its environment
- Reward computation should be outside of agent so that agent cannot arbitrarily change them

The agent-environment boundary represents the limit of the **agent's absolute control**, not of its knowledge

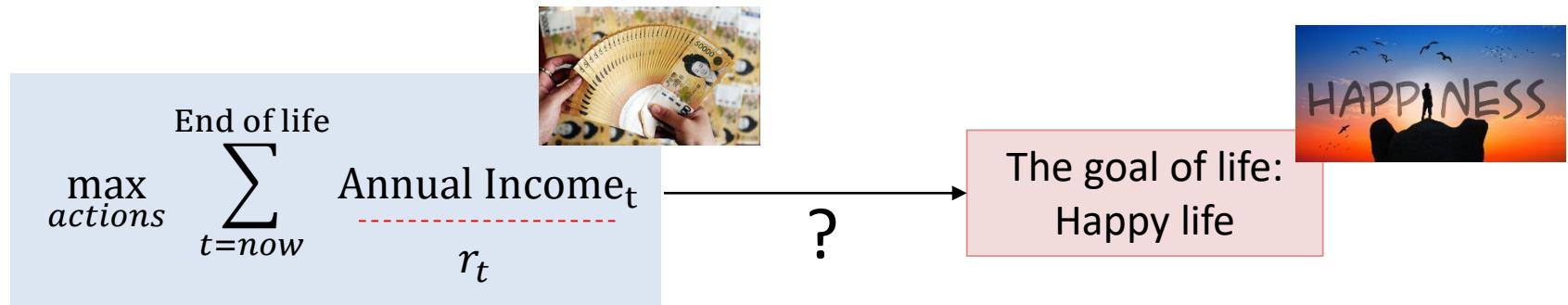
## Goals and Rewards



- The agent's goal is to maximize the total amount of reward (cumulative reward) it receives.

$$\max_{a_t, a_{t+1}, \dots} \sum_k r_{t+k}$$

- Rewards we setup truly indicate what we want accomplished
- The reward signal is your way of communicating to the agent **what you want it to achieve**, not how you want it achieved



## Utility (returns)

### How to formally define the accumulated reward in the long run?

- Denote reward :  $r_t = R(S_t, A_t)$
- In the episodic tasks, the simplest utility is defined as

$$U_t = r_t + r_{t+1} + r_{t+3} + \dots + r_T = \sum_{k=0}^T r_{t+k}$$

- ✓  $T$  is a final time step associated with the terminal time step  $T$
- ✓ Examples include, maze, go, chess, etc.

- In the continuing tasks, the discounted utility is defined as

$$U_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+3} + \dots = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$$

- ✓  $\gamma$  is a parameter,  $0 \leq \gamma \leq 1$ , called discount rate
- ✓ Examples include, maze game, Go, Chess, etc.
- ✓  $\gamma = 0$  (myopic): concern only with maximizing immediate rewards
- ✓  $\gamma = 1$  (farsighted): the objective takes future rewards take into account more strongly

## Utility (returns)

Assumptions:

1. Remove index for episode number

Episode 1  $(s_{1,1}, a_{1,1}, r_{1,1}), (s_{2,1}, a_{2,1}, r_{2,1}), \dots, (s_{t,1}, a_{t,1}, r_{t,1}), \dots, (s_{T,1}, a_{T,1}, r_{T,1})$

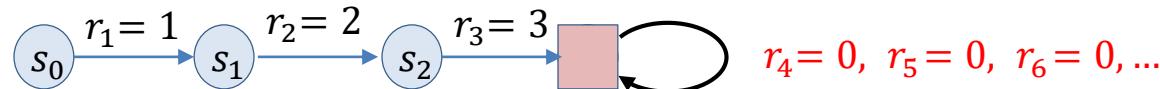
Episode 2  $(s_{1,2}, a_{1,2}, r_{1,2}), (s_{2,2}, a_{2,2}, r_{2,2}), \dots, (s_{t,n}, a_{t,n}, r_{t,2}), \dots, (s_{T,1}, a_{T,1}, r_{T,2})$

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

Episode n  $(s_{1,n}, a_{1,n}, r_{1,n}), (s_{2,n}, a_{2,n}, r_{2,n}), \dots, (s_{t,n}, a_{t,n}, r_{t,n}), \dots, (s_{T,n}, a_{T,n}, r_{T,n})$

$(s_1, a_1, r_1), (s_2, a_2, r_2), \dots, (s_t, a_t, r_t), \dots, (s_T, a_T, r_T)$

2. Introduce the observing state, in which agent transits the same state with no reward



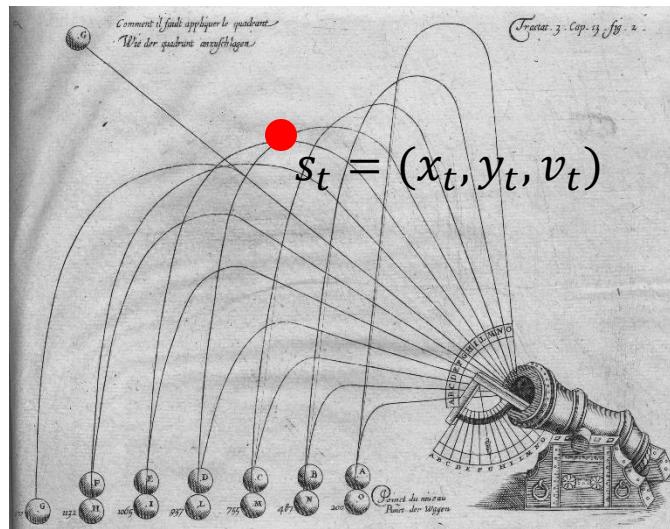
- In the continuing tasks + the episodic tasks , the unified utility is defined as

$$U_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+3} + \dots = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$$

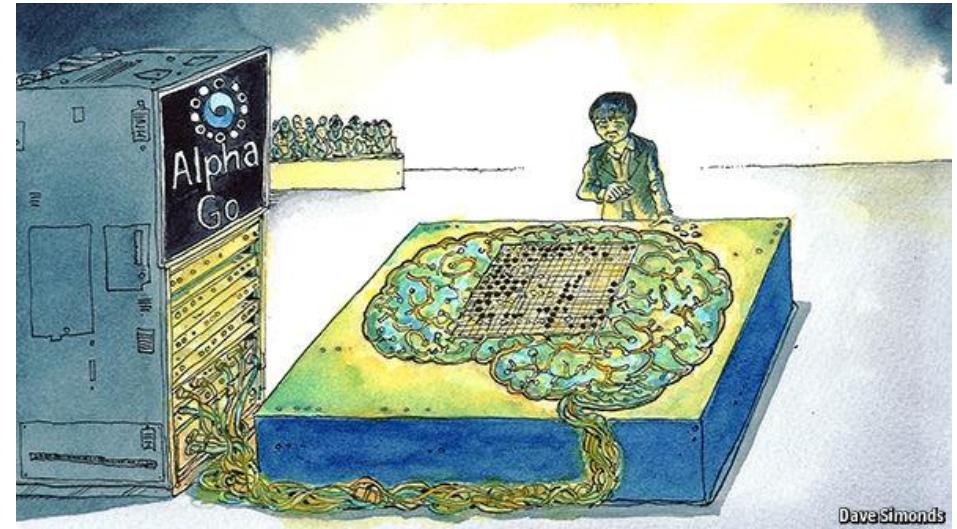
- ✓  $\gamma$  is a parameter,  $0 \leq \gamma \leq 1$ , called discount rate
- ✓ Examples include, maze game, Go, Chess, etc.
- ✓  $\gamma = 0$  (myopic): concern only with maximizing immediate rewards
- ✓  $\gamma = 1$  (farsighted): the objective takes future rewards take into account more strongly

## The Markov Property

- The state is constructed and maintained on the basis of **immediate sensations** together with **the previous state or some other memory of past sensations**
- Ideal state signal summarizes past sensations compactly, yet in such a way that all relevant information is retained
- A state signal that succeeds in retaining all relevant information is said to be **Markov**



$s_t$  = the location and velocity of the flying canon



$s_t$  = the configuration of the black and white stones

## The Markov Property

- “Markov” generally means that given the present state, the future and the past are independent



Andrey Markov (1856-1922)

- For Markov decision processes, “Markov” means action outcomes depend only on the current state

$$\begin{aligned} P(S_{t+1} = s' | S_t = s_t, A_t = a_t, S_{t-1} = s_{t-1}, A_{t-1} = a_{t-1}, \dots, S_0 = s_0, A_0 = a_0) \\ = P(S_{t+1} = s' | S_t = s_t, A_t = a_t, ) \end{aligned}$$

- The best policy for choosing actions as a function of a Markov state is just as good as the best policy for choosing actions as a function of complete history

$$\pi^*(s_t, s_{t-1}, \dots, s_0) = \pi^*(s_t)$$

Note:

- As states become more Markovian, the better performance in MDP solution
- It is useful to think of the state at each time step as an approximation to a Markov value

## Policy

- A policy  $\pi$  gives an action  $a \in \mathcal{A}$  for each state  $s \in \mathcal{S}$ :

$$\pi: \mathcal{S} \rightarrow \mathcal{A}$$

- An optimal policy  $\pi^*$  is one that maximizes expected utility

$$\pi^* = \underset{\pi}{\operatorname{argmax}} \mathbb{E}[U^\pi(s)] \text{ for all } s$$

$$\text{where } \mathbb{E}[U^\pi(s)] = \mathbb{E}[r_t + \gamma r_{t+1} + \gamma^2 r_{t+3} + \dots]$$

- An optimal policy can be either deterministic or stochastic

deterministic

$$a^* = \pi^*(s)$$

Stochastic

$$p(a|s) = \pi^*(s, a)$$

Take action  $a^*$  with a probability one

Take action  $a$  with a probability  $p(a|s)$

We will exclusively consider deterministic policy

Policy



Reward



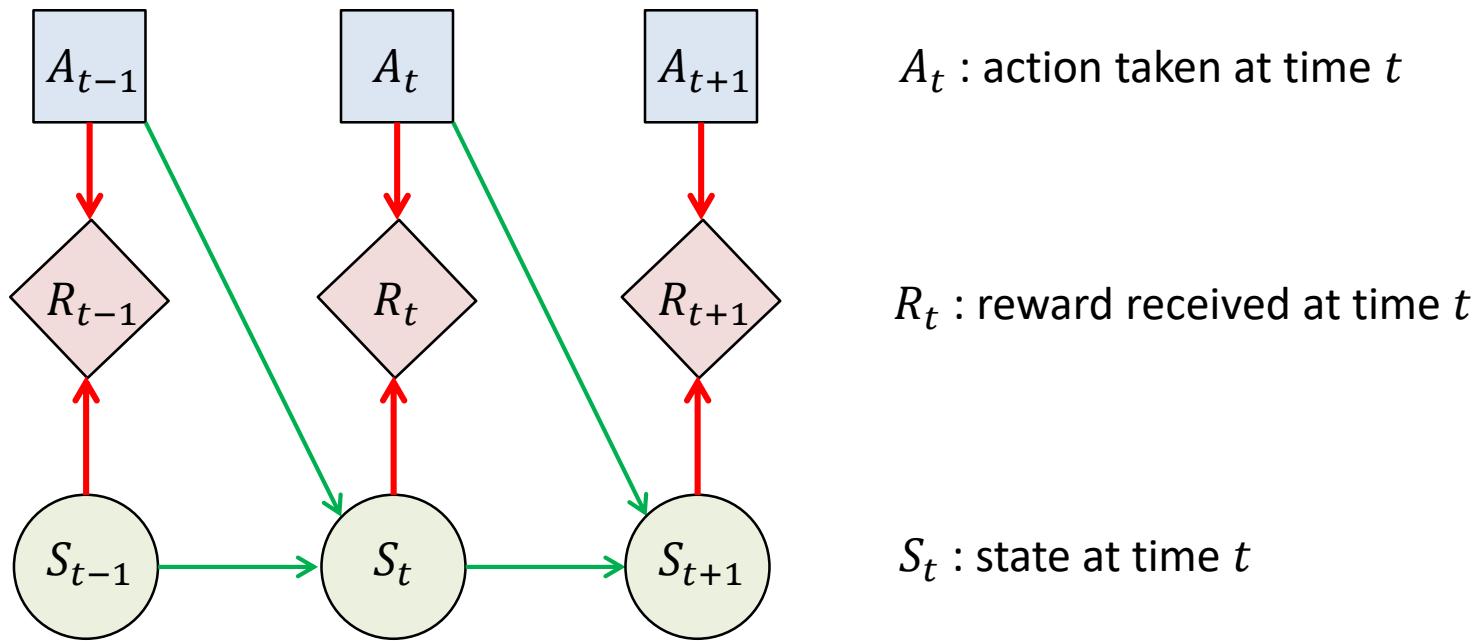
## Markov Decision Processes

Finite Markov Decision Process (MDP) :The state and action space are finite

An MDP is defined by:

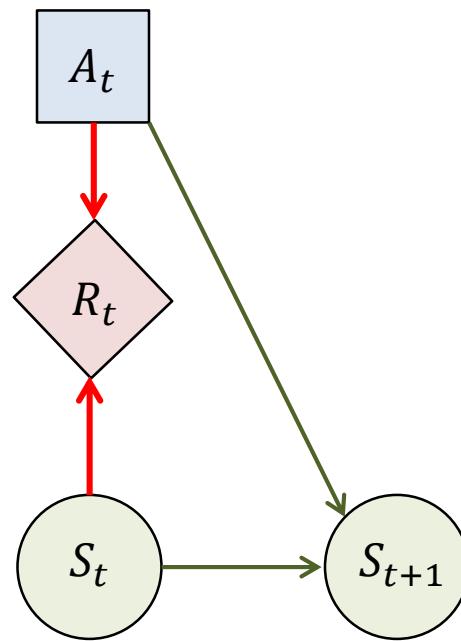
- A set of states  $s \in \mathcal{S}$
- A set of actions  $a \in \mathcal{A}$
- A transition function  $T(s, a, s') = P(S_{t+1} = s' | S_t = s, A_t = a) = P(s' | s, a)$ 
  - ✓ Probability that a from  $s$  leads to  $s'$  when taking action  $a$
  - ✓ Also called the model or the dynamics
- A reward function  $R(s, a, s')$ 
  - ✓  $r_t = R(s_t, a_t, s_{t+1})$  or  $r_{t+1} = R(s_t, a_t)$
  - ✓ If stochastic,  $R(s, a, s') = \mathbb{E}[r_t + r_{t+1}, \dots | S_t = s, A_t = a, S_{t+1} = s']$
- A start state  $s_0 \in \mathcal{S}$
- A terminal state  $s_T \in \mathcal{S}^+$  (for episodic tasks)

## Markov Decision Processes



- Transition probability  $T_t(s_t, a_t, s_{t+1}) = P(S_{t+1} = s_{t+1} | S_t = s_t, A_t = a_t) = P(s_{t+1} | s_t, a_t)$ 
  - ✓ represents the probability of transitioning from state  $s_t$  to  $s_{t+1}$  after executing action  $a_t$  at time  $t$  (**Markov assumption**)
- Reward function:  $r_t = R_t(s_t, a_t)$  : represents the probability of receiving reward  $r_t$  when executing action  $a_t$  from state  $s_t$  at time  $t$

## (Stationary) Markov Decision Processes



### Stationary Markov Decision Process (MDP)

→ transition and reward models are **stationary**

- Transition probability  $T(s_t, a_t, s_{t+1})$  are the same for all time step  $t$
- Reward function:  $r_t = R(s_t, a_t)$  are the same for all time step  $t$

## Value Function & Q function

### Value function (**state value function for $\pi$** )

“How good it is for the agent to be in a given state”

$V^\pi(s)$  : The expected utility received by following policy  $\pi$  from state  $s$

$$V^\pi(s) = \mathbb{E}_\pi(U_t | S_t = s) = \mathbb{E}_\pi\left(\sum_{k=0}^{\infty} \gamma^k r_{t+k} \mid S_t = s\right)$$

$\mathbb{E}_\pi$  : not expectation over policy  $\pi$  but all stochastic state transitions associated with  $\pi$

### Q-function (**action-value function for $\pi$** )

“How good it is for the agent to perform a given action in a given state”

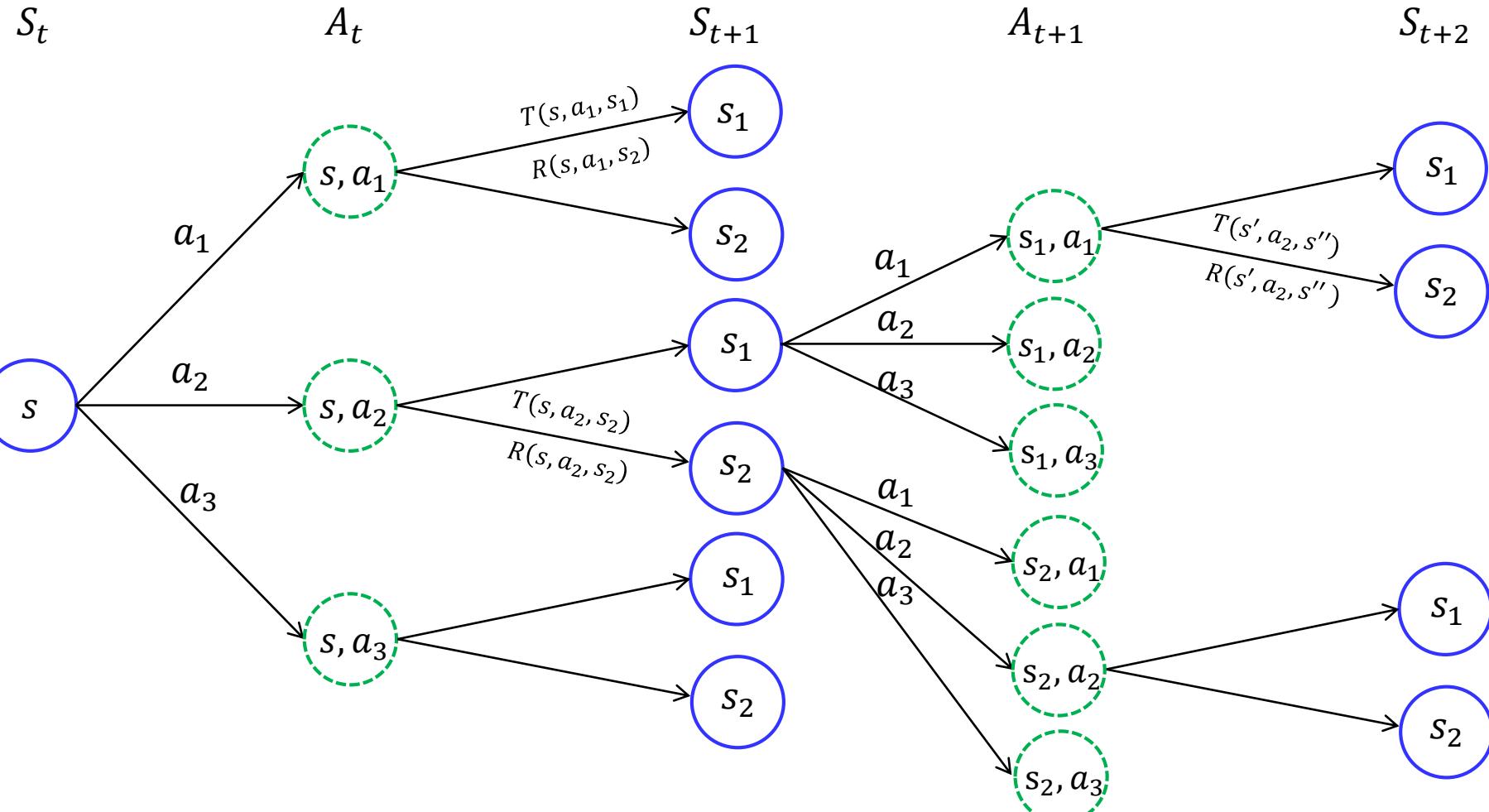
$Q^\pi(s, a)$  : The expected utility of taking action  $a$  from state  $s$ , and then following policy  $\pi$

$$Q^\pi(s, a) = \mathbb{E}_\pi(U_t | S_t = s, A_t = a) = \mathbb{E}_\pi\left(\sum_{k=0}^{\infty} \gamma^k r_{t+k} \mid S_t = s, A_t = a\right)$$

Because the agent can expect to receive in the future depend on what actions it will take  
→ Value and Q functions are defined with respect to a particular policy mapping state  $s \in \mathcal{S}$  to an action  $a \in \mathcal{A}$

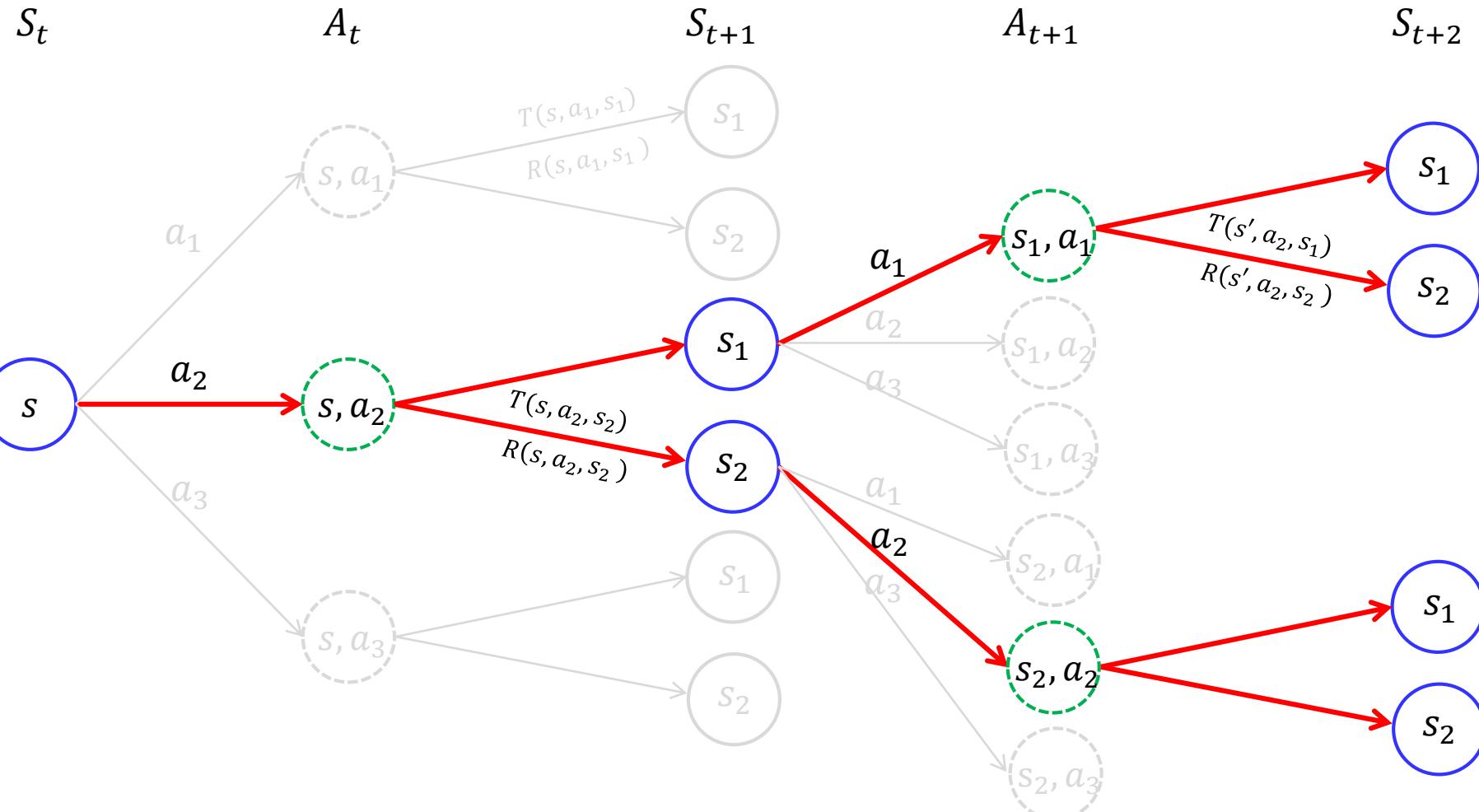
## Value Function

All possible trajectories



## Value Function

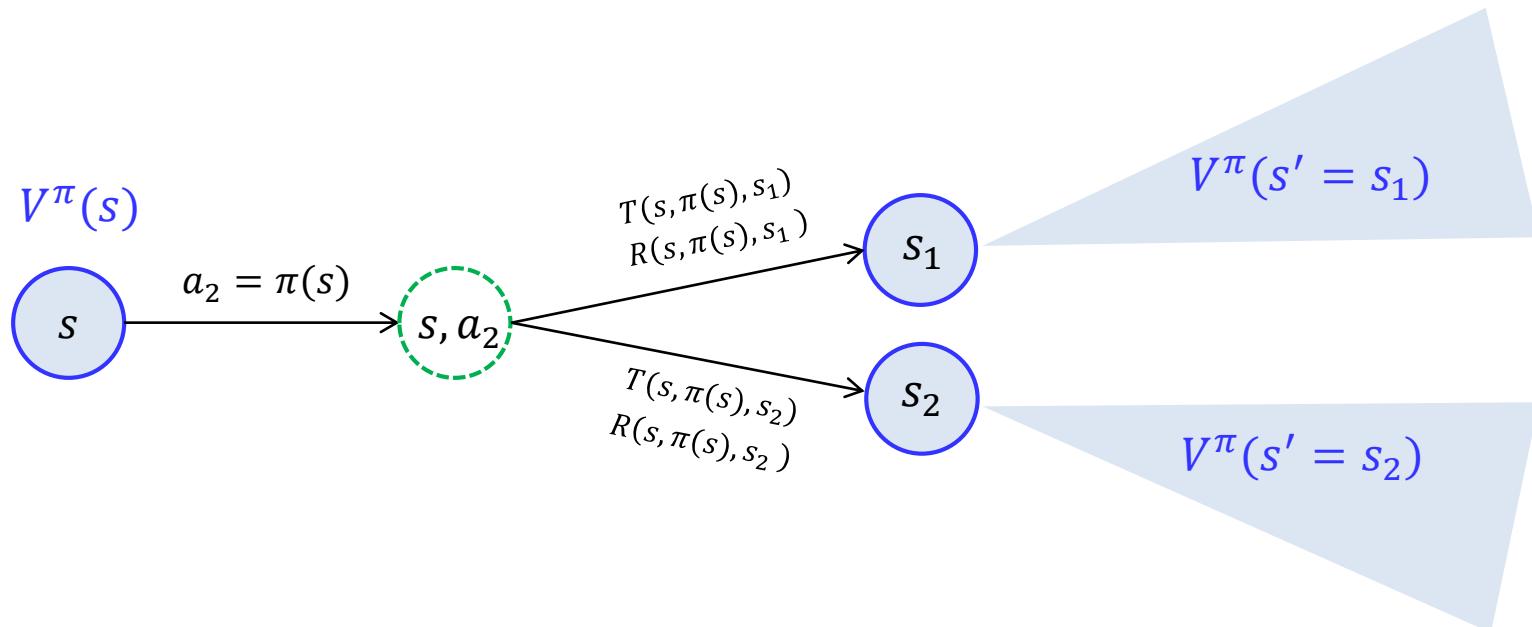
A policy  $\pi$  is given as:  $\pi(s) = a_2$ ;  $\pi(s_1) = a_1$ ;  $\pi(s_2) = a_2$



## Value Function

A policy  $\pi$  is given as:  $\pi(s) = a_2$ ;  $\pi(s_1) = a_1$ ;  $\pi(s_2) = a_2$

$S_t$                      $A_t$                      $S_{t+1}$                      $A_{t+1}$                      $S_{t+2}$



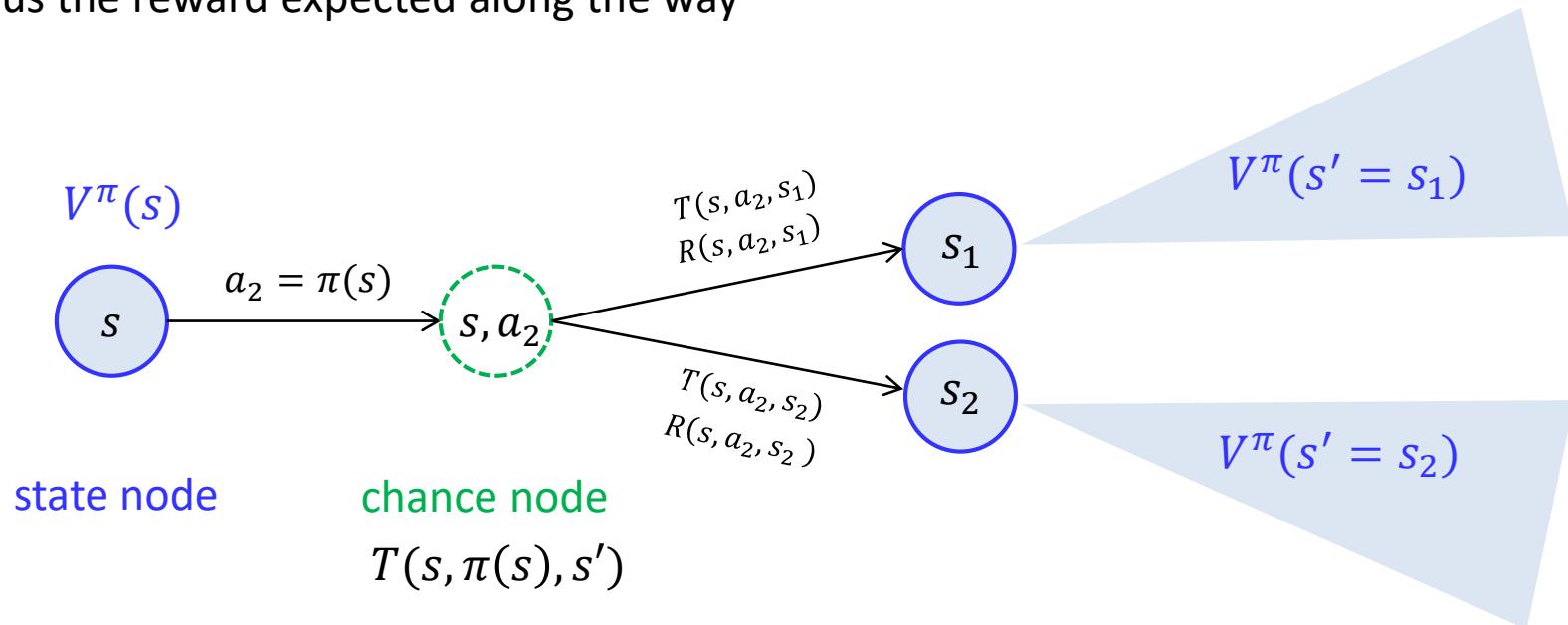
$$V^\pi(s) = T(s, \pi(s), s_1)\{R(s, \pi(s), s_1) + \gamma V^\pi(s_1)\} + T(s, \pi(s), s_2)\{R(s, \pi(s), s_2) + \gamma V^\pi(s_2)\}$$

## The Bellman Equation for Value Function

### Recursive Formulation (The Bellman equation)

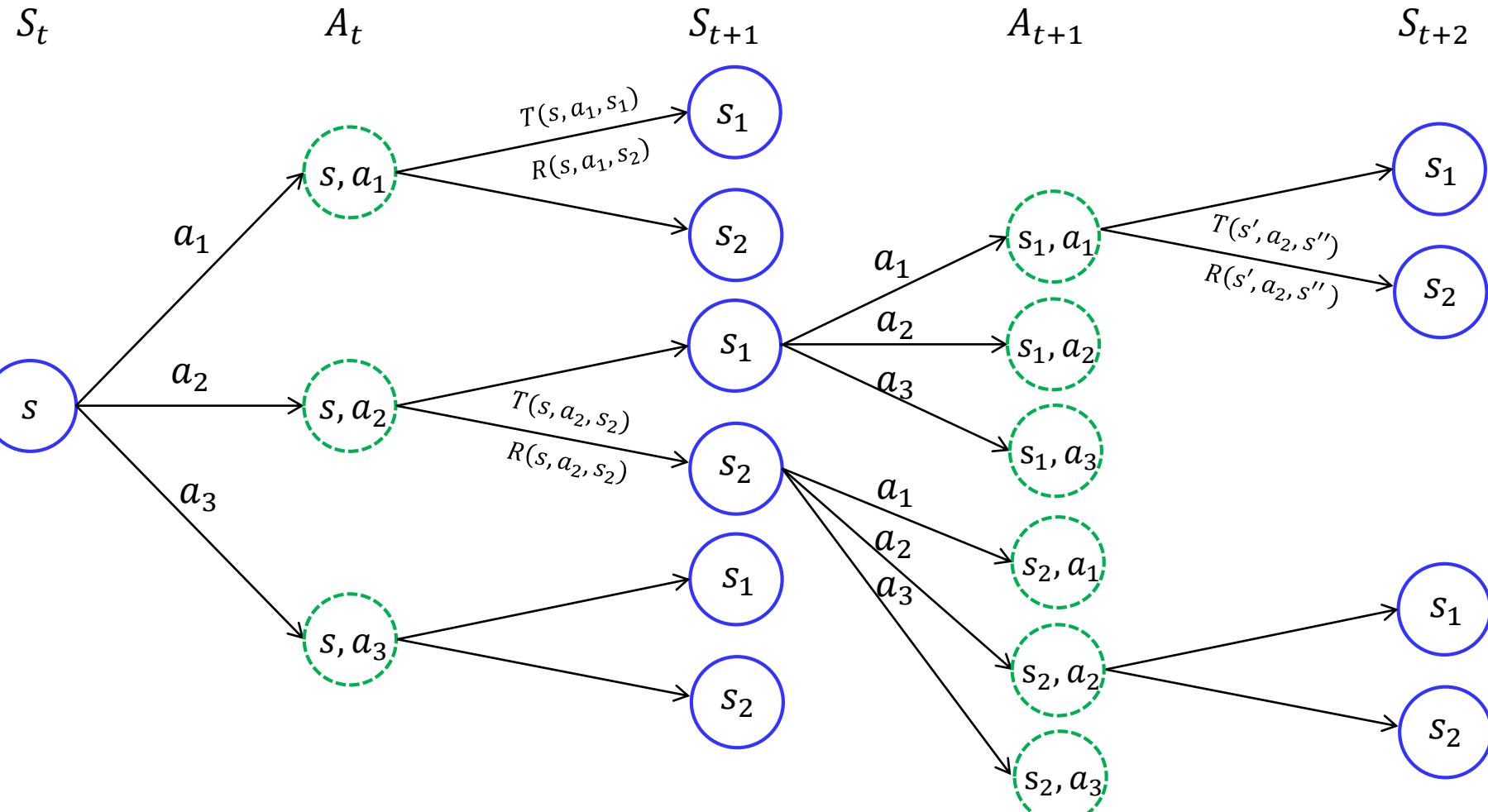
$$\begin{aligned} V^\pi(s) &= \mathbb{E}_\pi \left( \sum_{k=0}^{\infty} \gamma^k r_{t+k} \mid S_t = s \right) \\ &= \mathbb{E}_\pi \left( r_t + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+1+k} \mid S_t = s \right) \\ &= \sum_{s'} T(s, \pi(s), s') \{ R(s, \pi(s), s') + \gamma \mathbb{E}_\pi \left( \sum_{k=0}^{\infty} \gamma^k r_{t+1+k} \mid S_{t+1} = s' \right) \} \\ &= \sum_{s'} T(s, \pi(s), s') \{ R(s, \pi(s), s') + \gamma V^\pi(s') \} \end{aligned}$$

The value of the start state must equal the (discounted) value of the expected next state, plus the reward expected along the way



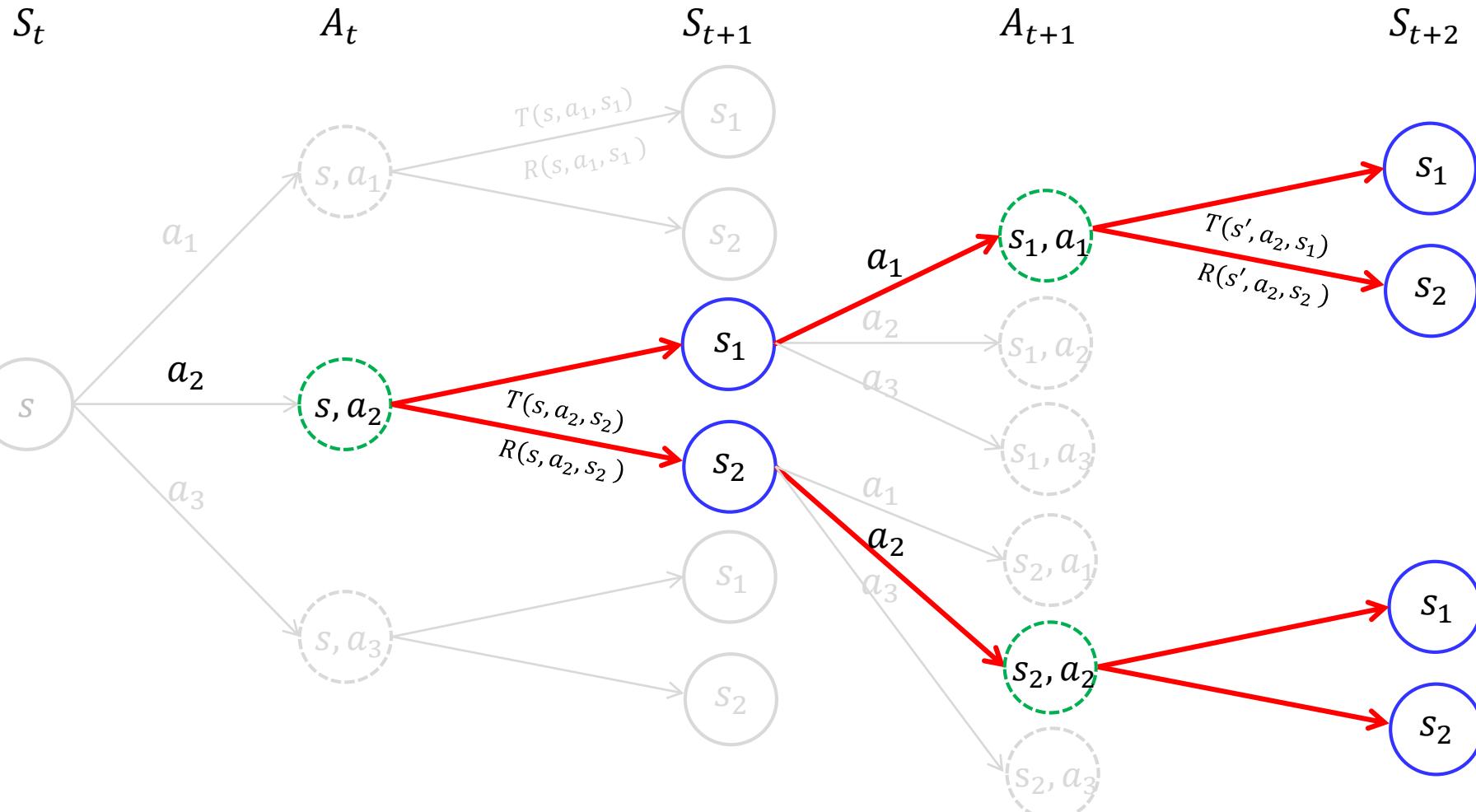
## Q function

All possible trajectories



## Q function

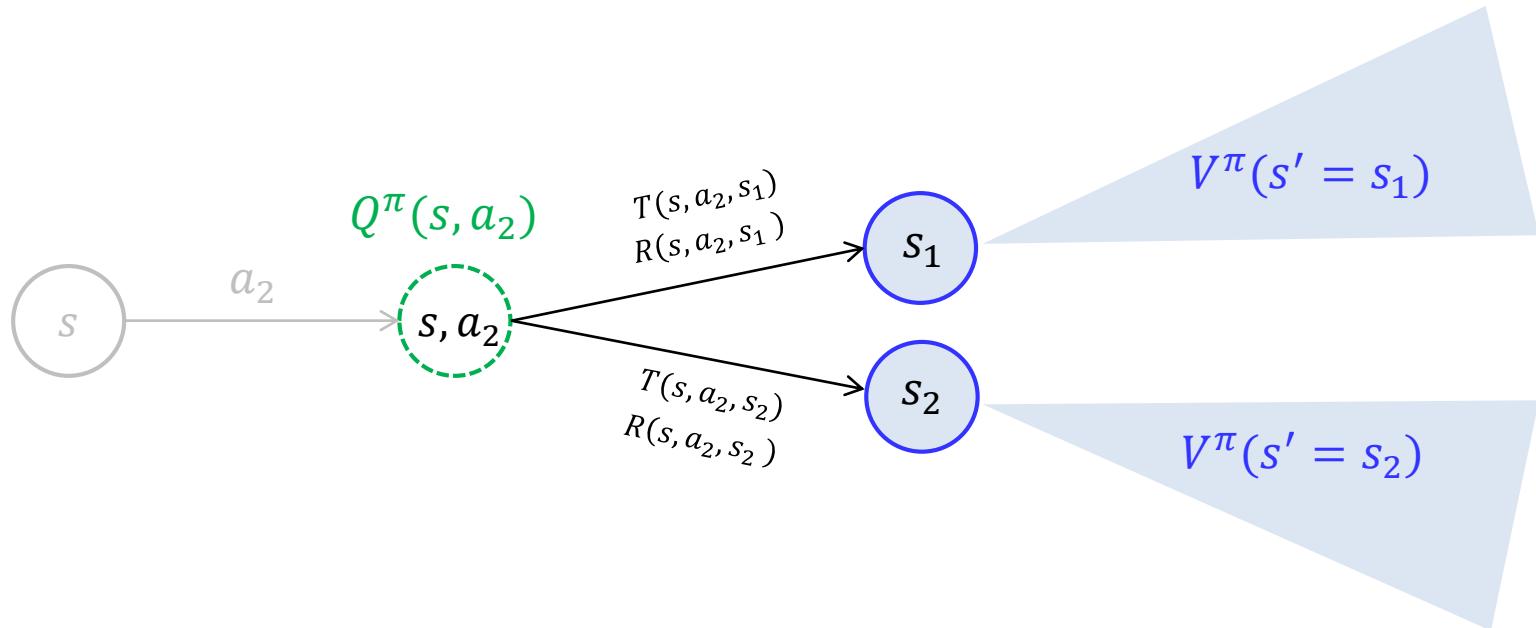
A policy  $\pi$  is given as:  $\pi(s) = a_2$ ;  $\pi(s_1) = a_1$ ;  $\pi(s_2) = a_2$



## Q Function

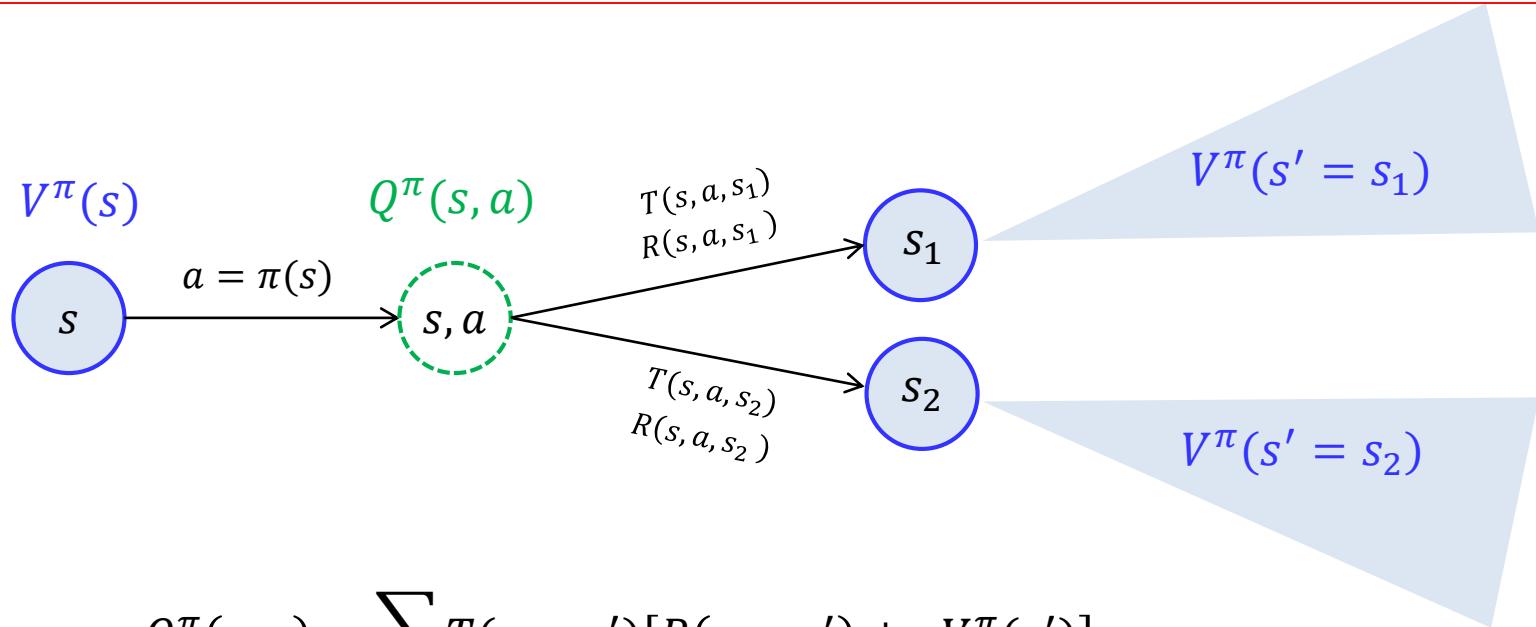
A policy  $\pi$  is given as:  $\pi(s) = a_2$ ;  $\pi(s') = a_1$ ;  $\pi(s'') = a_2$

$s_t$                      $A_t$                      $s_{t+1}$                      $A_{t+1}$                      $s_{t+2}$



$$Q^\pi(s, a_2) = T(s, a_2, s_1)\{R(s, a_2, s_1) + \gamma V^\pi(s_1)\} + T(s, a_2, s_2)\{R(s, a_2, s_2) + \gamma V^\pi(s_2)\}$$

## Q Function

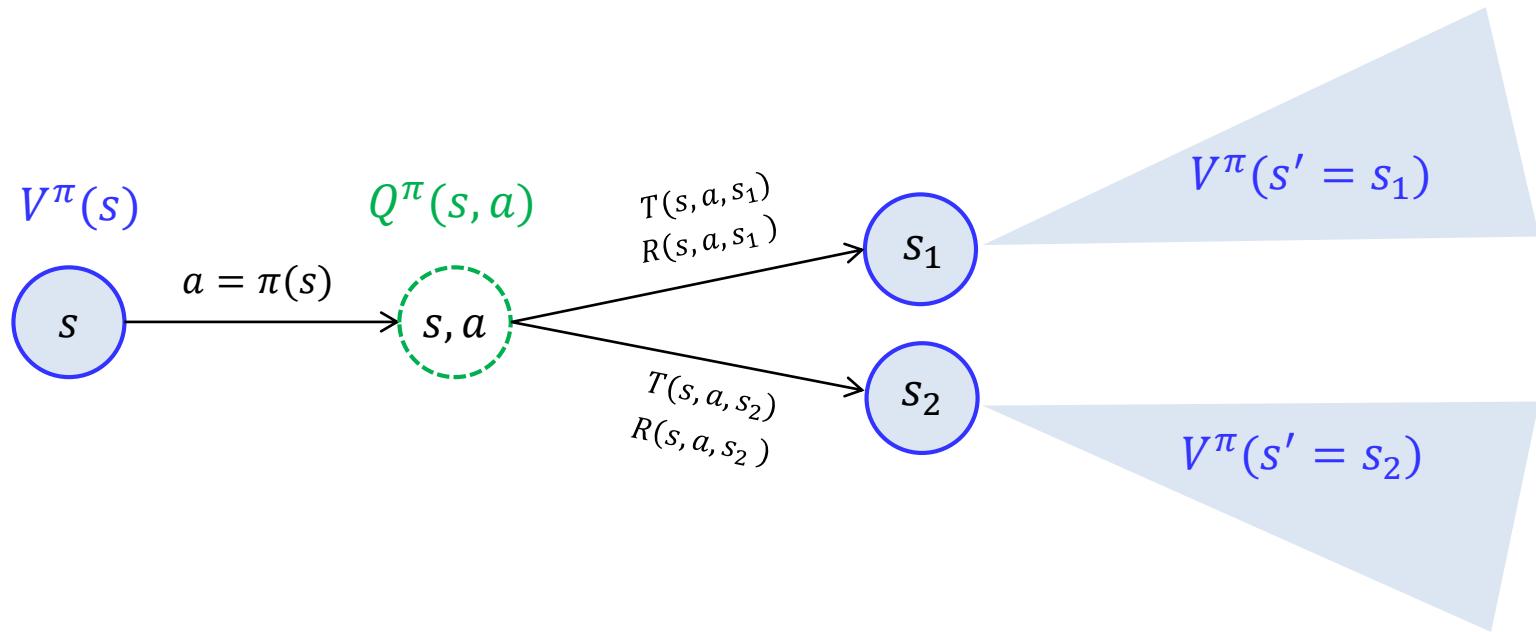


Note that:

$$\begin{aligned} V^\pi(s) &= \sum_{s'} T(s, \pi(s), s') \{R(s, \pi(s), s') + \gamma V^\pi(s')\} \\ &= Q^\pi(s, \pi(s)) \end{aligned}$$

- $Q^\pi(s, a)$  is more general since it has the option to select an action  $a$  given state  $s$
- If the action is enforced to select  $a = \pi(s)$  according to the policy  $\pi$ ,  $V^\pi(s) = Q^\pi(s, \pi(s))$

## Summary for Value function and Q function



$$V^\pi(s) = \sum_{s'} T(s, \pi(s), s') \{ R(s, \pi(s), s') + \gamma V^\pi(s') \}$$

$$Q^\pi(s, a) = \sum_{s'} T(s, a, s') [R(s, a, s') + \gamma V^\pi(s')]$$

$$V^\pi(s) = Q^\pi(s, \pi(s))$$

## Optimal Value Function & Q function

$\pi' \geq \pi$  if and only if  $V^{\pi'}(s) \geq V^\pi(s)$  for all  $s$

### Optimal state-value function

$$V^*(s) = \max_{\pi} V^\pi(s) \text{ for all } s$$

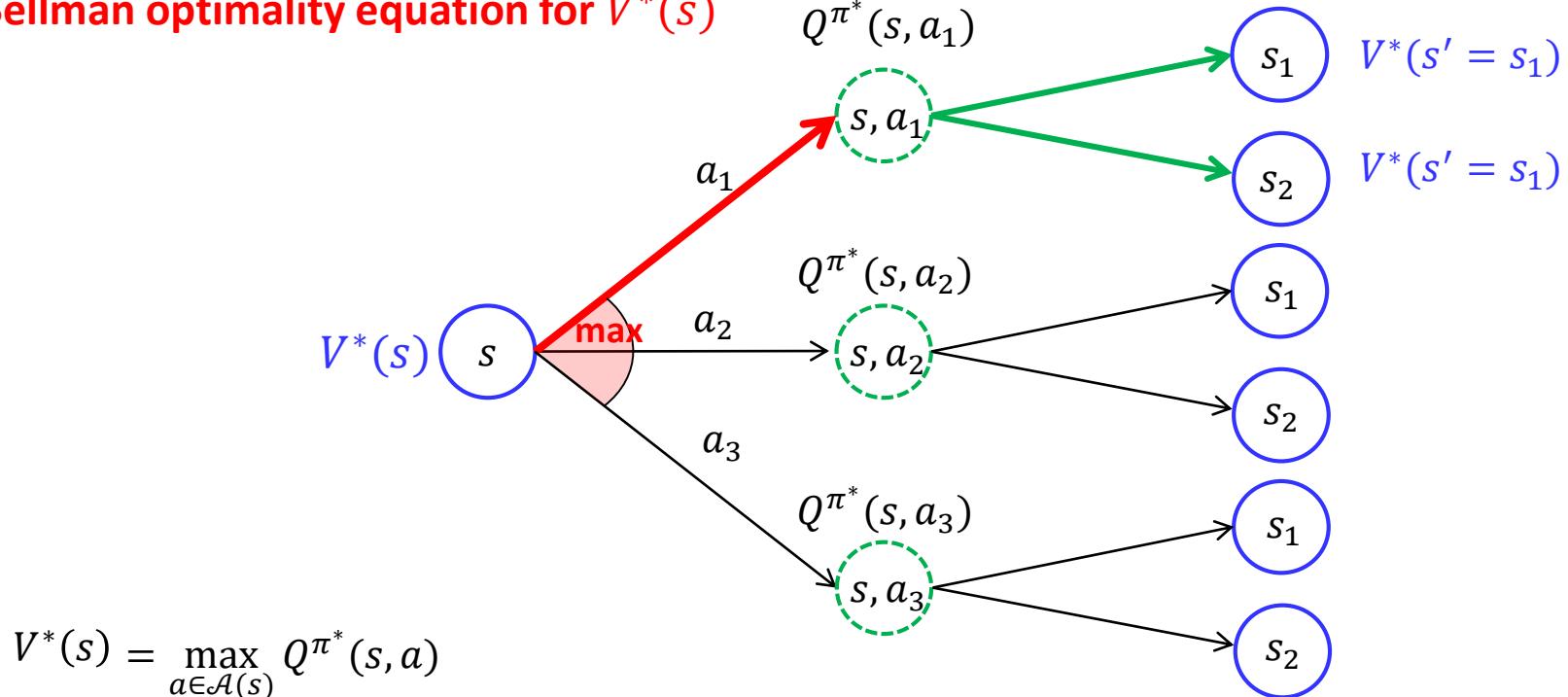
### Optimal action-value function

$$Q^*(s, a) = \max_{\pi} Q^\pi(s, a) \text{ for all } s \in \mathcal{S} \text{ and } a \in \mathcal{A}(s)$$

$$\begin{aligned} Q^*(s, a) &= \max_{\pi} Q^\pi(s, a) && \because Q^\pi(s, a) = \mathbb{E}[R(s, a, s') + \gamma V^\pi(s') | s_t = s, a_t = a] \\ &= \max_{\pi} \mathbb{E}[R(s, a, s') + \gamma V^\pi(s') | s_t = s, a_t = a] && \mathbb{E} \text{ is over the next state } s' \\ &= \mathbb{E} \left[ R(s, a, s') + \gamma \max_{\pi} V^\pi(s') | s_t = s, a_t = a \right] \\ &= \mathbb{E}[R(s, a, s') + \gamma V^*(s') | s_t = s, a_t = a] \end{aligned}$$

## Bellman Optimality Equation for State-Value Function

Bellman optimality equation for  $V^*(s)$



$$= \max_{a \in \mathcal{A}(s)} \mathbb{E}_{\pi^*} \left( \sum_{k=0}^{\infty} \gamma^k r_{t+k} \mid S_t = s, A_t = a \right)$$

$$= \max_{a \in \mathcal{A}(s)} \mathbb{E}_{\pi^*} \left( r_t + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid S_t = s, A_t = a \right)$$

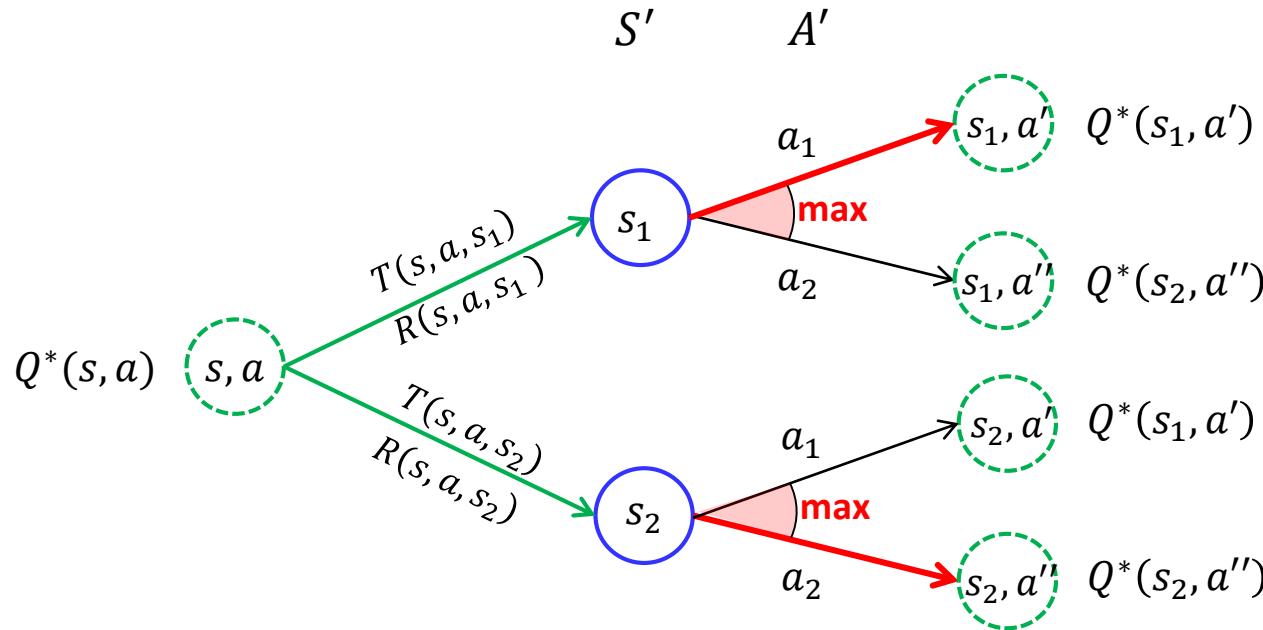
$$= \max_{a \in \mathcal{A}(s)} \mathbb{E} (r_t + \gamma V^*(s_{t+1}) \mid S_t = s, A_t = a)$$

$$= \max_{a \in \mathcal{A}(s)} \sum_{s'} T(s, a, s') \{R(s, a, s') + \gamma V^*(s')\}$$

First take optimum action and follow the optimum policy

## Bellman Optimality Equation for State-Action Value Function

Bellman optimality equation for  $Q^*(s, a)$

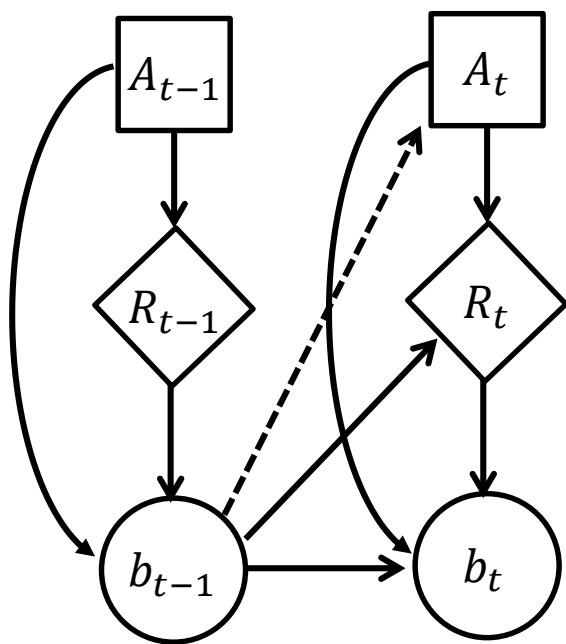


$$\begin{aligned} Q^*(s, a) &= \mathbb{E} \left\{ r_t + \gamma \max_{a'} Q^*(s', a') | s_t = s, a_t = a \right\} && \text{E is over transitions } s' \rightarrow s' \\ &= \sum_{s'} T(s, a, s') \left\{ R(s, a, s') + \gamma \max_{a'} Q^*(s', a') \right\} \end{aligned}$$

First transits by transition probability and take the optimum action for each consequent states

## Reinforcement Learning Approach

### MDP over belief state and finding optimal policy using Dynamic Programming



- $h_t = [(a_1, r_1), (a_1, r_1), \dots, (a_t, r_t)]$
- $\theta = (\theta_i, \dots, \theta_n)$  : Unknown machine parameters
- Belief state  $b_t(\theta) = P(\theta|h_t)$  : probability dist. on para.
- Updating **belief state**  $b_t(\theta)$  for Binary bandit with prior Beta( $\theta_i|\alpha, \beta$ ) : **Deterministic**

$$b_t(\theta_i) = b_{t-1}[a_t, r_t] = \text{Beta}(\theta_i|\alpha + w_{t,i}, \beta + l_{t,i})$$

$w_{t,i}$ : Accumulated wins with arm  $i$  up to time  $t$

$l_{t,i}$ : Accumulated loses with arm  $i$  up to time  $t$

### Dynamic programming on the value function

$$V_{t-1}(b_{t-1}) = \max_{\pi} E \left[ \sum_{t=t}^T r_t \right]$$

$$= \max_{a_t} \sum_{r_t} P(r_t|a_t, b_{t-1}) [r_t + V_t(b_{t-1}[a_t, r_t])]$$

$$P(r|a, b) = \int_{\theta_a} b(\theta_a) P(r|\theta_a) d\theta_a$$

## Optimum Planning as a Greedy Search

$$V^*(s) = \max_{a \in \mathcal{A}(s)} \sum_{s'} T(s, a, s') \{R(s, a, s') + \gamma V^*(s')\}$$

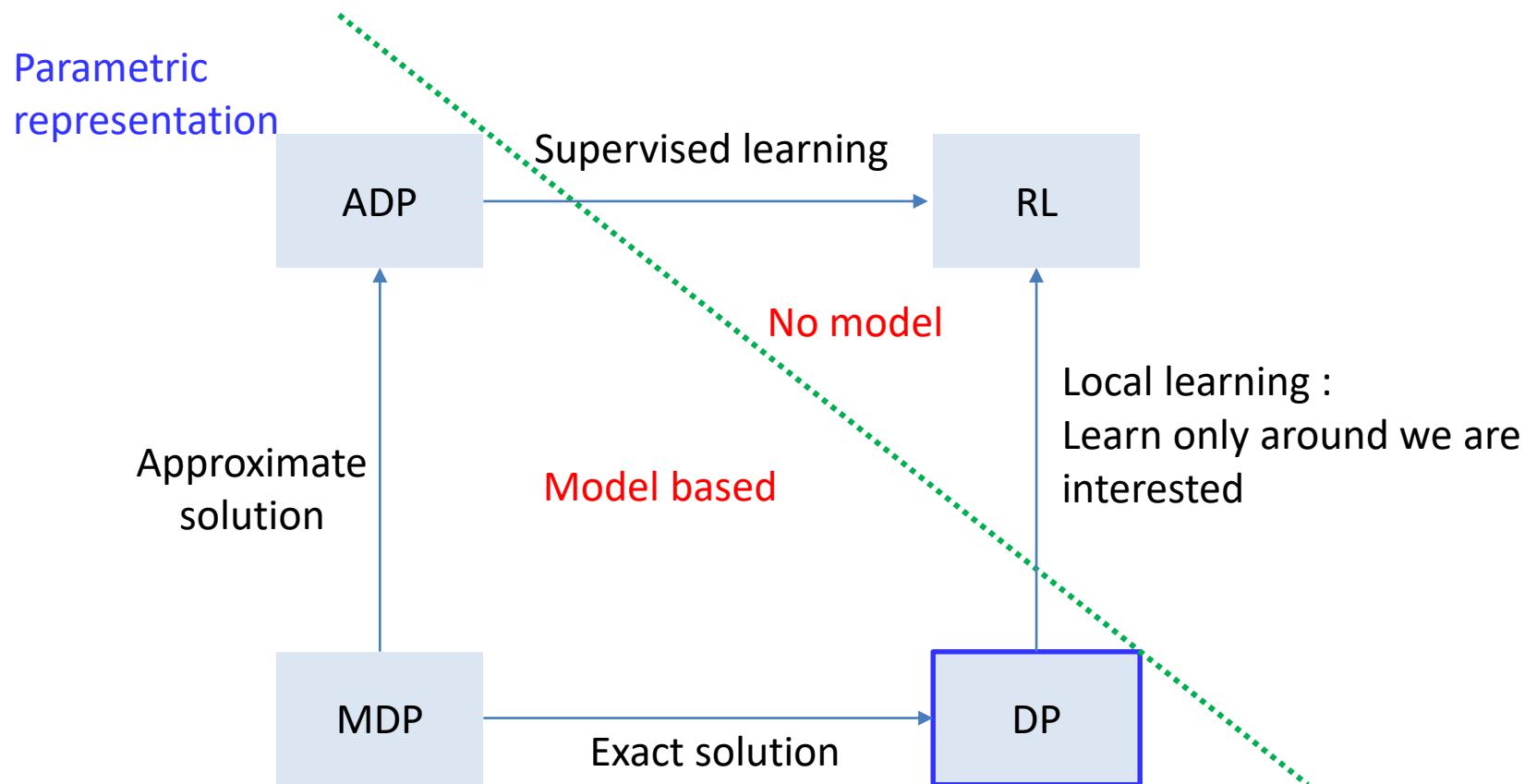
- The Bellman optimality equation is a system of equations, one for each state
  - ✓ For  $N$  states,  $N$  unknown and  $N$  equations
  - ✓ With known  $T(s, a, s')$  and  $R(s, a, s')$ , the equations can be solved using various mathematical programming (i.e., Linear programming, Quadratic programming)

### Reconstructing optimal policy with $Q^*(s, a)$ and $V^*(s)$

$$\begin{aligned} a^* &= \operatorname{argmax}_a Q^*(s, a) \\ &= \operatorname{argmax}_a \mathbb{E}[R(s, a, s') + \gamma V^*(s') | s_t = s, a_t = a] \end{aligned}$$

- Any greedy policy with respect to the optimal value function  $V^*(s)$  is an optimal policy  
→ because  $V^*(s)$  already takes into account the reward consequences of all possible future behavior
- The Q function effectively cashes the results of all one-step-ahead search

## Road Map for Next Lectures



- Usually Tabular representation
- For special case, DP can be used for solving a MDP with continuous space and action

- All other methods can be viewed as attempts to achieve much the same effect as DP, only with less computation and without assuming a perfect model of the environment

## Midterm statistics

### Under

Mean: 68.81  
std: 13.94  
Median: 71

### Grad

Mean: 79.14  
std: 6.99  
Median: 80

### Total:

Mean: 72.48  
std: 12.87  
Median: 74.75  
**Max: 88**

## Claims :

Problem 1:5 : 정요한 : becre1776@kaist.ac.kr

Problem 6:7: 박준영 : joon0105@kaist.ac.kr

Problem 8:10: 이한선: Joanna@kaist.ac.kr

# **L14. Markov Decision Process (Dynamic Programming Approach)**

## **L14. Markov Decision Process (Dynamic Programming Approach)**

1. Policy Evaluation
2. Policy Improvement
3. Policy iteration
4. Value Iteration

## Summary

$$\begin{aligned}
 V^\pi(s) &= \mathbb{E}\{r_{t+1} + \gamma V^\pi(s_{t+1}) | s_t = s\} \\
 &= \sum_{s'} T(s, \pi(s), s') \{R(s, \pi(s), s') + \gamma V^\pi(s')\}
 \end{aligned}$$

$$\begin{aligned}
 Q^\pi(s, a) &= \mathbb{E}\{r_{t+1} + \gamma Q^\pi(s, \pi(s_{t+1})) | s_t = s, a_t = a\} \\
 &= \sum_{s'} T(s, a, s') [R(s, a, s') + \gamma Q^\pi(s, \pi(s'))] \\
 &= \sum_{s'} T(s, a, s') [R(s, a, s') + \gamma V^\pi(s')] \quad V^\pi(s) = Q^\pi(s, \pi(s))
 \end{aligned}$$

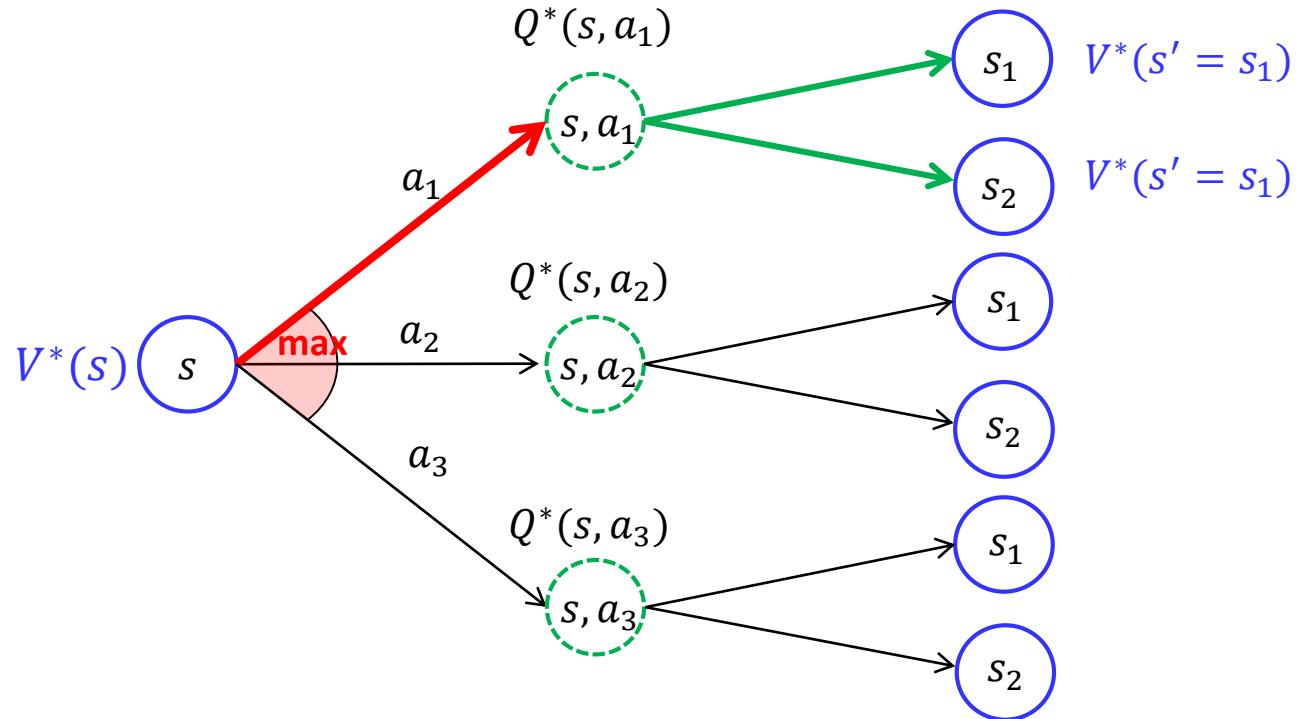

---

$$\begin{aligned}
 V^*(s) &= \max_{a \in \mathcal{A}(s)} \mathbb{E}\{r_{t+1} + \gamma V^*(s_{t+1}) | s_t = s\} \\
 &= \max_{a \in \mathcal{A}(s)} \sum_{s'} T(s, a, s') \{R(s, a, s') + \gamma V^*(s')\}
 \end{aligned}$$

$$\begin{aligned}
 Q^*(s, a) &= \mathbb{E}\left\{r_{t+1} + \gamma \max_{a'} Q^*(s_{t+1}, a') | s_t = s, a_t = a\right\} \\
 &= \sum_{s'} T(s, a, s') \left\{R(s, a, s') + \gamma \max_{a'} Q^*(s', a')\right\} \\
 &= \sum_{s'} T(s, a, s') \{R(s, a, s') + \gamma V^*(s')\} \quad \because V^*(s') = \max_{a'} Q^*(s', a')
 \end{aligned}$$

## Summary

Bellman optimality equation for  $V^*(s)$

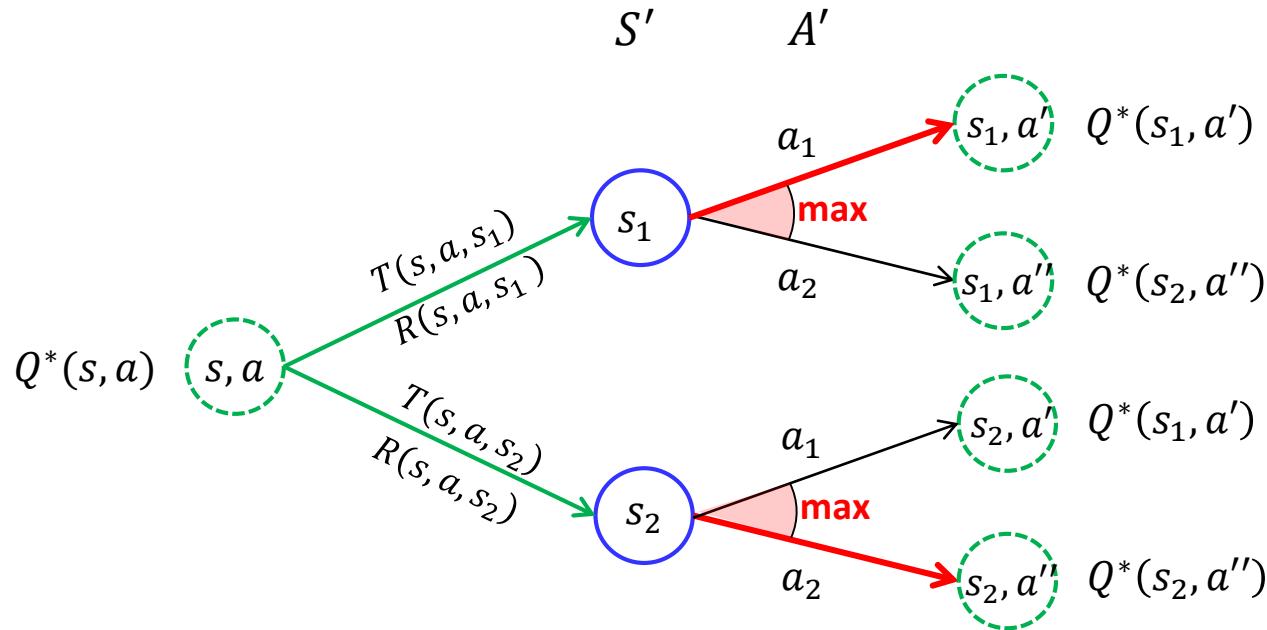


$$V^*(s) = \max_{a \in \mathcal{A}(s)} \sum_{s'} T(s, a, s') \{R(s, a, s') + \gamma V^*(s')\}$$

$$\therefore V^*(s) = \max_{a'} Q^*(s, a')$$

## Summary

### Bellman optimality equation for $Q^*(s, a)$



$$Q^*(s, a) = \sum_{s'} T(s, a, s') \{ R(s, a, s') + \gamma \max_{a'} Q^*(s', a') \}$$

$$= \sum_{s'} T(s, a, s') \{ R(s, a, s') + \gamma V^*(s') \} \quad \because V^*(s') = \max_{a'} Q^*(s', a')$$

## Dynamic Programming

- The term **dynamic programming (DP)** refers to a collection of algorithms that can be used to compute optimal policies given **a perfect model of the environment** as a Markov decision process (MDP)
- The key idea of DP (and reinforcement learning) is the use of value functions to organize and structure the search for good policies
- Optimal policies can be derived from the optimal value functions that satisfy the Bellman optimality equations

$$V^*(s) = \max_{a \in \mathcal{A}(s)} \sum_{s'} T(s, a, s') \{R(s, a, s') + \gamma V^*(s')\}$$

$$\begin{aligned} Q^*(s, a) &= \sum_{s'} T(s, a, s') \left\{ R(s, a, s') + \gamma \max_{a'} Q^*(s', a') \right\} \\ &= \sum_{s'} T(s, a, s') \{R(s, a, s') + \gamma V^*(s')\} \quad \because V^*(s') = \max_{a'} Q^*(s', a') \end{aligned}$$



Optimal policy

$$\begin{aligned} \pi^*(s) &= \operatorname{argmax}_a Q^*(s, a) \\ &= \operatorname{argmax}_a \sum_{s'} T(s, a, s') \{R(s, a, s') + \gamma V^*(s')\} \end{aligned}$$

### **Recycling Robot Example Systems of equations for the optimum Bellman function**

## Policy Evaluation

### Policy evaluation :

A method to compute the state-value function  $V^\pi(s)$  for an arbitrary policy  $\pi: \mathcal{S} \rightarrow \mathcal{A}$

$$V^\pi(s) = \sum_{s'} T(s, \pi(s), s') \{R(s, \pi(s), s') + \gamma V^\pi(s')\}$$

- A system of  $|\mathcal{S}|$  simultaneous linear equations in  $|\mathcal{S}|$  unknown

### Algorithm

**Initialize**  $V_{t=0}^\pi(s) \leftarrow 0$  for all states  $s \in S$

**Repeat** (iteration  $t = 0, \dots$ ):

**For each state**  $s$ :

$$V_{t+1}^\pi(s) \leftarrow \sum_{s'} T(s, \pi(s), s') \{R(s, \pi(s), s') + \gamma V_t^\pi(s')\}$$

**Until**  $\max_{s \in \mathcal{S}} |V_{t+1}^\pi - V_t^\pi(s)| \leq \epsilon$

### Full backup:

Each iteration of iterative policy evaluation backs up the value of every state once to produce the new approximate value function  $V_{t+1}^\pi$

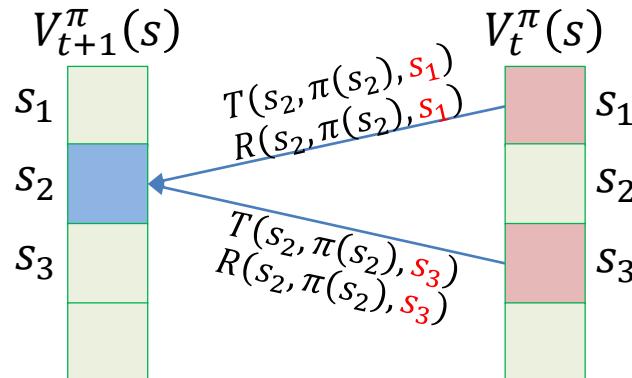
## Policy Evaluation

$$V_{t+1}^{\pi}(s) \leftarrow \sum_{s'} T(s, \pi(s), s') \{R(s, \pi(s), s') + \gamma V_t^{\pi}(s')\}$$

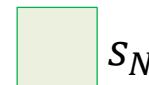
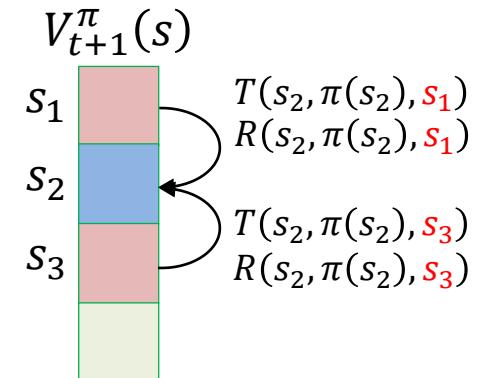
Example:

$$\begin{aligned} V_{t+1}^{\pi}(s_2) &= \sum_{s'} T(s_2, \pi(s_2), s') \{R(s_2, \pi(s_2), s') + \gamma V_t^{\pi}(s')\} \\ &= T(s_2, \pi(s_2), s_1) \{R(s_2, \pi(s_2), s_1) + \gamma V_t^{\pi}(s_1)\} + T(s_2, \pi(s_2), s_3) \{R(s_2, \pi(s_2), s_3) + \gamma V_t^{\pi}(s_3)\} \end{aligned}$$

“Two-arrays” update



“In place” update



Usually faster!  
Less memory

## Policy Evaluation

Example : Grid world

$$MDP = \{\mathcal{S}, \mathcal{A}, T, R, \gamma\}$$

- $\mathcal{S} = \{1, 2, \dots, 14\}$
- $\mathcal{A} = \{\uparrow, \downarrow, \rightarrow, \leftarrow\}$
- $T(s, s', a) = \begin{cases} 1, & \text{if move is allowed} \\ 0, & \text{if move is not allowed} \end{cases}$

$$T(5, 6, \rightarrow) = 1 \quad T(5, 10, \rightarrow) = 0 \quad T(7, 7, \rightarrow) = 1$$

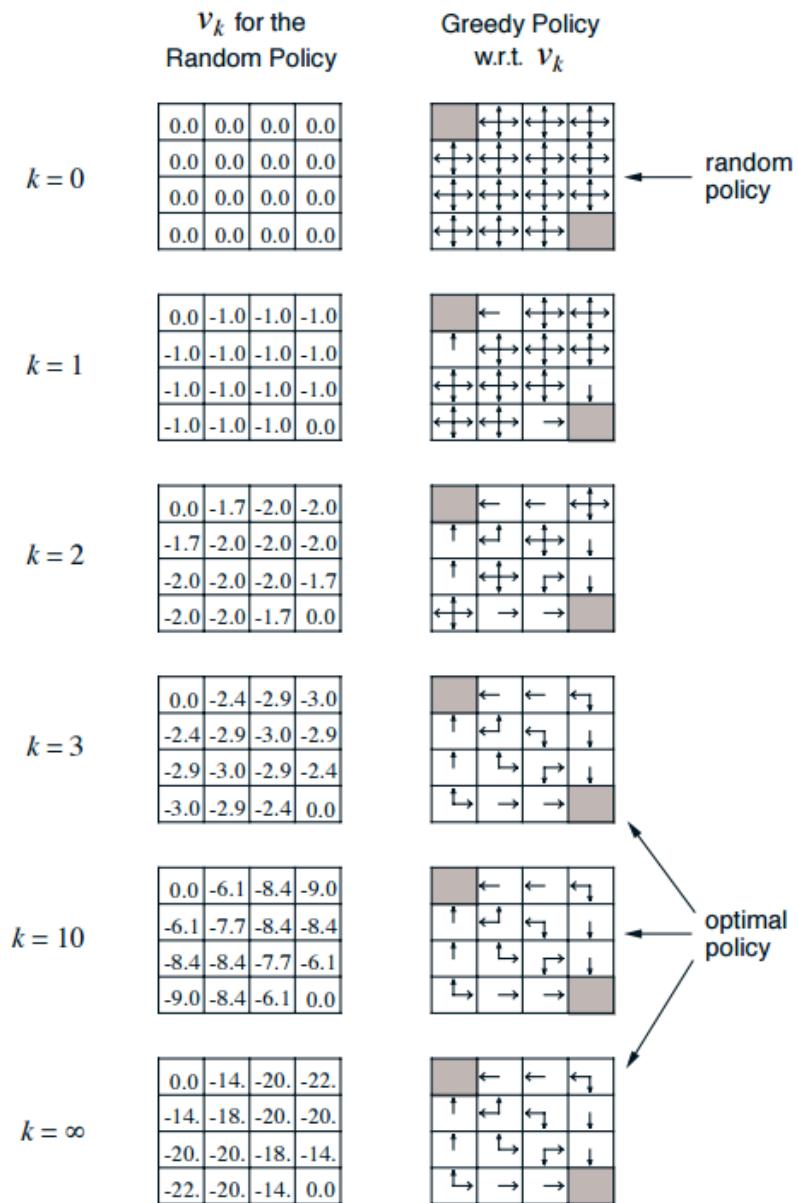
The actions that would take the agent off the grid leave the state unchanged

- $R(s, s', a) = -1$  for all  $s, s', a$
- $\gamma = 1$

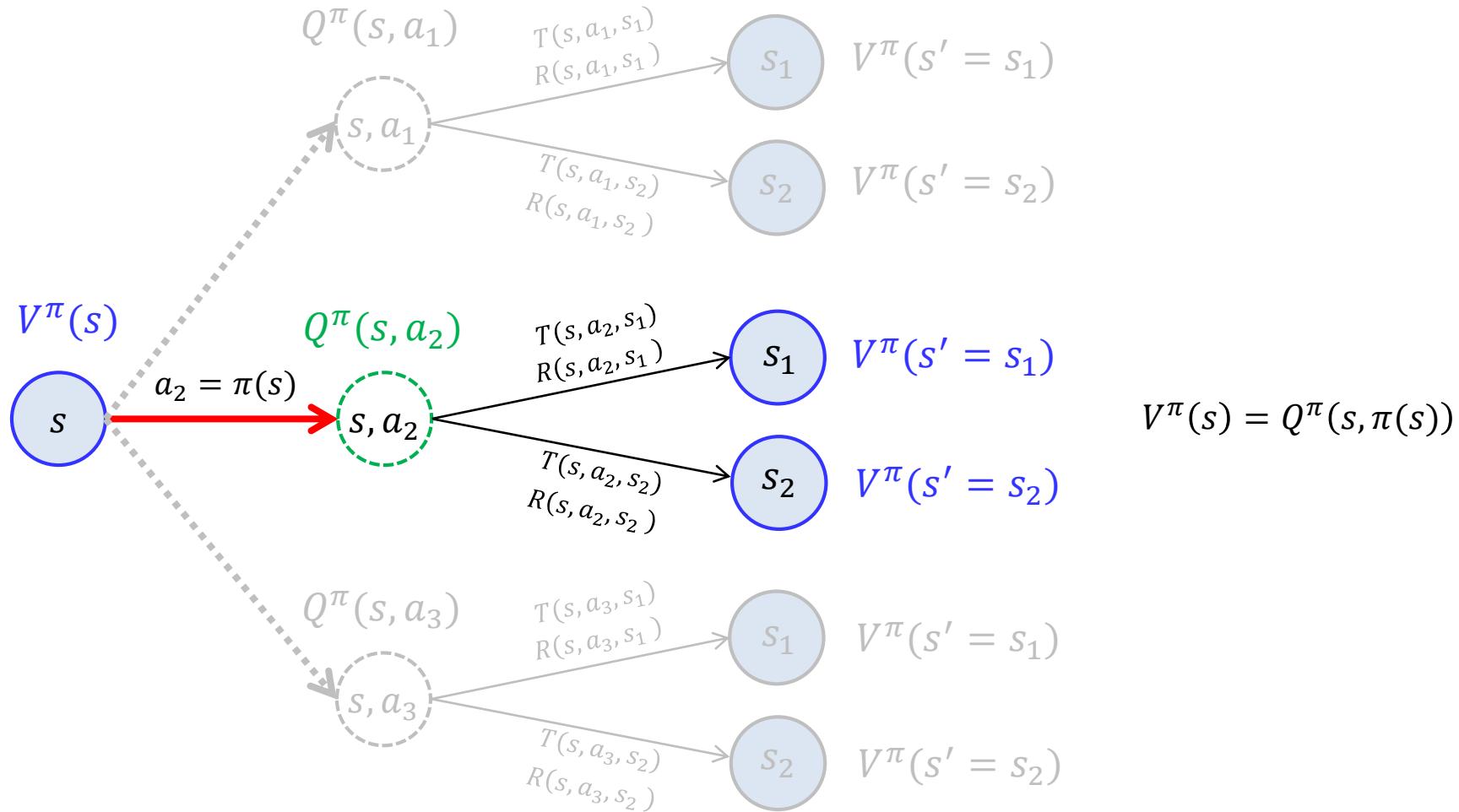
★	1	2	3
4	5	6	7
8	9	10	11
12	13	14	★

Suppose the agent follows the equiprobable random policy (all actions equally likely), what is the value function?

# Policy Evaluation



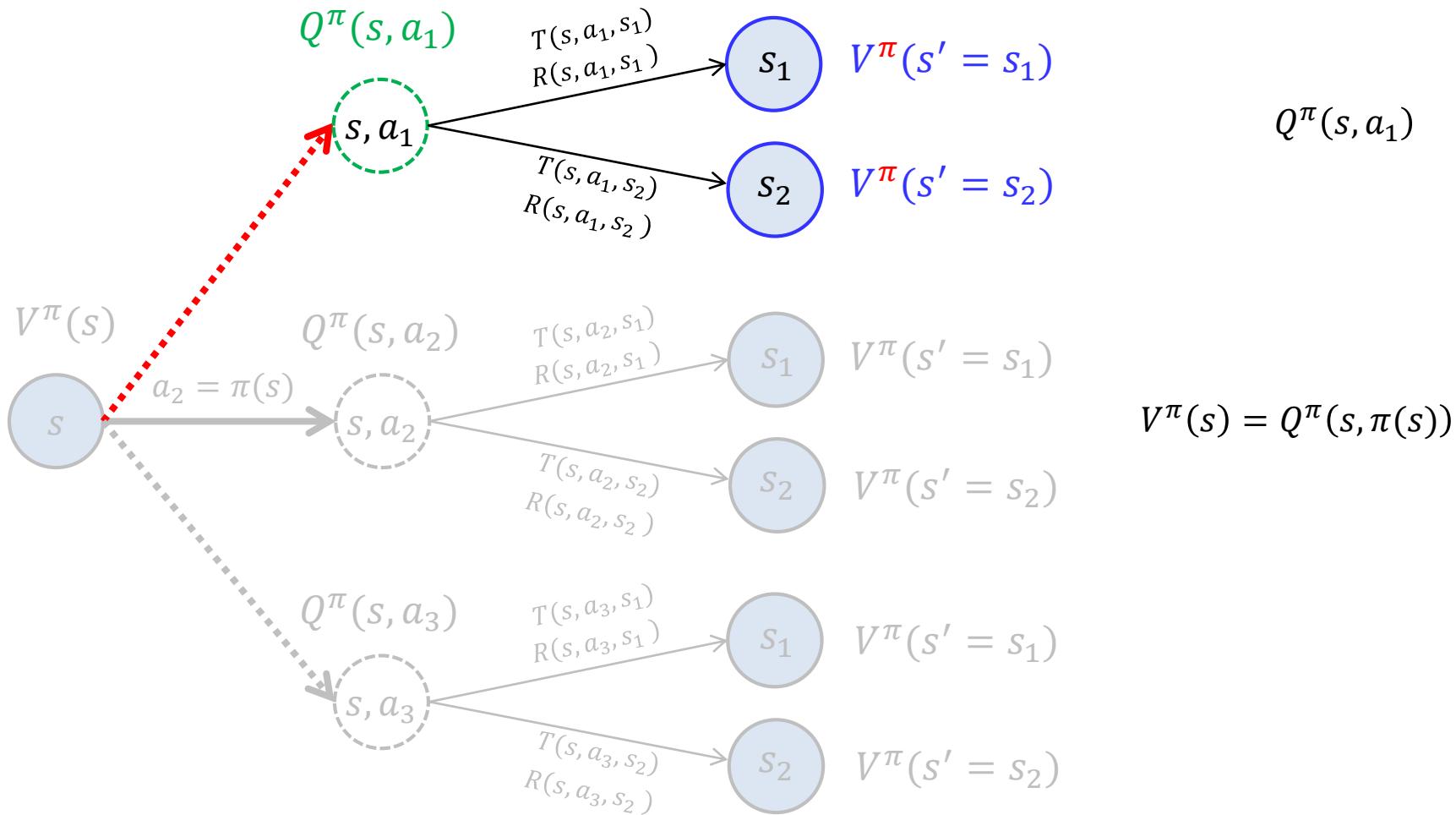
## Policy Improvement



We know how good it is to follow the current policy from  $s$  based on  $V^\pi(s)$   
 Would it be better or worse to change to the new policy?

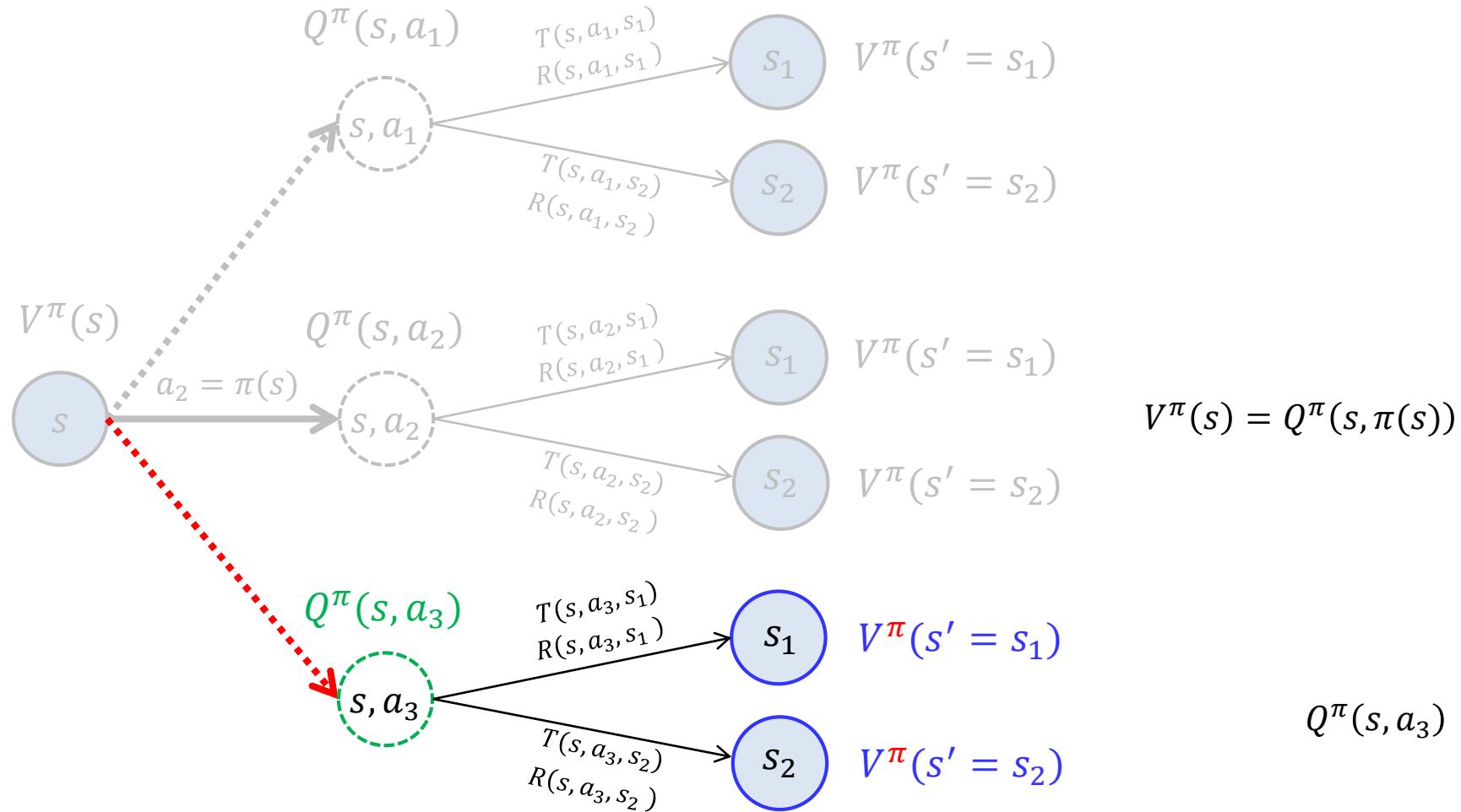
→ Select  $a$  given  $s$  and thereafter following the existing policy  $\pi$  (a single step change)

## Policy Improvement



$$Q^\pi(s, a_1) \geq V^\pi(s) = Q^\pi(s, \pi(s))?$$

## Policy Improvement



$$Q^\pi(s, a_3) \geq V^\pi(s) = Q^\pi(s, \pi(s))?$$

## Policy Improvement

$$\pi'(s) = \operatorname{argmax}_{a \in \mathcal{A}(s)} Q^\pi(s, a)$$

$$\rightarrow Q^{\pi'}(s, \pi'(s)) \geq Q^{\pi}(s, \pi(s)) = V^{\pi}(s)$$

$$x^* = \operatorname{argmax}_x f(x)$$

$\rightarrow f(x^*) \geq f(x)$  for all  $x$

**Improvement criterion** =

Expected reward provided by **changing one step action** and **following the original policy**

If it is better to select  $a = \pi'(s)$  once in  $s$  and thereafter follow  $\pi$  than it would be to follow  $\pi$  all the time,



It is better still to select  $a = \pi'(s)$  whenever  $s$  is encountered  
**(The new policy  $\pi'(s)$  is a better policy overall)**

## Policy Improvement

$$\pi'(s) = \operatorname{argmax}_{a \in \mathcal{A}(s)} Q^\pi(s, a)$$

$$\rightarrow Q^{\pi'}(s, \pi'(s)) \geq Q^\pi(s, \pi(s)) = V^\pi(s)$$

$$x^* = \operatorname{argmax}_x f(x)$$

$\rightarrow f(x^*) \geq f(x)$  for all  $x$

**Improvement criterion** =

Expected reward provided by **changing one step action** and **following the original policy**

## Proof (Policy improvement Theorem)

Policy improvement must give us a strictly better policy  $\pi'(s)$  than the older policy  $\pi(s)$  except when the original policy is already optimal  $\pi(s) = \pi^*(s)$

$$Q^\pi(s, \pi'(s)) \geq V^\pi(s) \rightarrow$$

---

$$V^{\pi'}(s) \geq V^\pi(s) \text{ for all states } s \in \mathcal{S}$$

$$\pi' \geq \pi$$

## Policy Improvement

### Proof (Policy improvement Theorem)

Policy improvement must give us a strictly better policy  $\pi'(s)$  than the older policy  $\pi(s)$  except when the original policy is already optimal  $\pi(s) = \pi^*(s)$

$$Q^\pi(s, \pi'(s)) \geq V^\pi(s) \rightarrow V^{\pi'}(s) \geq V^\pi(s) \text{ for all states } s \in \mathcal{S}$$

$$V^\pi(s) \leq Q^\pi(s, \pi'(s))$$

Given

$$= \mathbb{E}_{\pi'}[r_{t+1} + \gamma V^\pi(s_{t+1}) | s_t = s]$$

$\mathbb{E}_{\pi'}$  is expectation over  $s_{t+1}$  induced by  $\pi'$

$$\leq \mathbb{E}_{\pi'}[r_{t+1} + \gamma Q^\pi(s_{t+1}, \pi'(s_{t+1})) | s_t = s]$$

$$\because V^\pi(s_{t+1}) \leq Q^\pi(s_{t+1}, \pi'(s_{t+1}))$$

$$= \mathbb{E}_{\pi'}[r_{t+1} + \gamma \mathbb{E}_{\pi'}[r_{t+2} + \gamma V^\pi(s_{t+2})] | s_t = s]$$

$$= \mathbb{E}_{\pi'}[r_{t+1} + \gamma r_{t+2} + \gamma^2 V^\pi(s_{t+2}) | s_t = s]$$

$$\leq \mathbb{E}_{\pi'}[r_{t+1} + \gamma r_{t+2} + \gamma^2 Q^\pi(s_{t+2}, \pi'(s_{t+2})) | s_t = s]$$

$$= \mathbb{E}_{\pi'}[r_{t+1} + \gamma r_{t+2} + \gamma^2 \mathbb{E}_{\pi'}[r_{t+3} + \gamma V^\pi(s_{t+3})] | s_t = s]$$

$$= \mathbb{E}_{\pi'}[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 V^\pi(s_{t+3}) | s_t = s]$$

:

$$= \mathbb{E}_{\pi'}[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 r_{t+4} + \dots | s_t = s]$$

$$= V^{\pi'}(s)$$

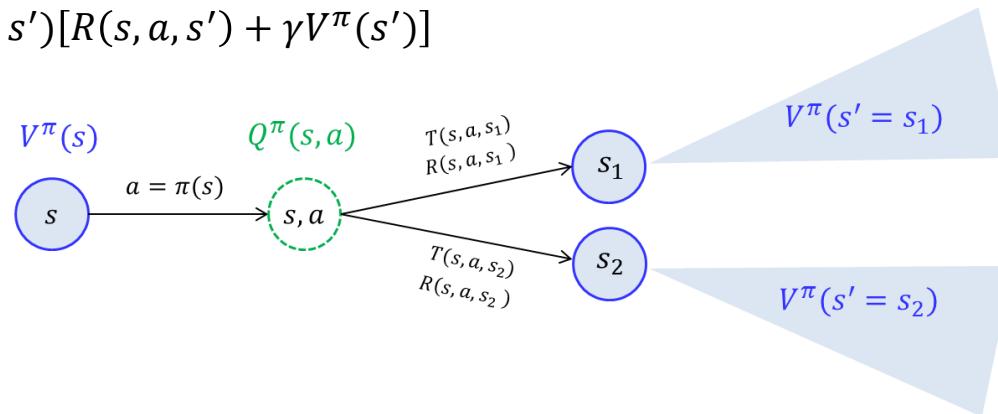
Thus,  $\pi \leq \pi'$

## Policy Improvement

### Policy improvement :

The process of making a new policy  $\pi^{new}$  that improves the original policy  $\pi$ , by making it greedy or nearly greedy with respect of the value function of the original policy

Recall:  $Q^\pi(s, a) = \sum_{s'} T(s, a, s')[R(s, a, s') + \gamma V^\pi(s')]$



### Algorithm

Input : value of policy  $V^\pi(s)$

Output: new policy  $\pi'$

For each state  $s \in \mathcal{S}$

1. Compute  $Q^\pi(s, a) = \sum_{s'} T(s, a, s')[R(s, a, s') + \gamma V^\pi(s')]$  for each  $a$

2. Compute  $\pi'(s) = \operatorname{argmax}_{a \in \mathcal{A}(s)} Q^\pi(s, a)$

$$= \operatorname{argmax}_{a \in \mathcal{A}(s)} \sum_{s'} T(s, a, s')[R(s, a, s') + \gamma V^\pi(s')]$$

## Policy Iteration

### Policy iteration :

Iterative way of finding the optimum policy through sequence of policy evaluation and policy improvement



### Algorithm

```
\pi \leftarrow \text{arbitrary}
For t = 1, ..., t_{PI} (or until \pi stops changing)
    Run policy evaluation to compute V^\pi
    Run policy improvement to get new improved policy \pi'
    \pi \leftarrow \pi'
```

- Policy evaluation require iterative computation, requiring multiple sweeps through the state set
- policy evaluation starts with the value function for the *previous policy*

→ Fast convergence

## Policy Iteration

### Policy iteration :

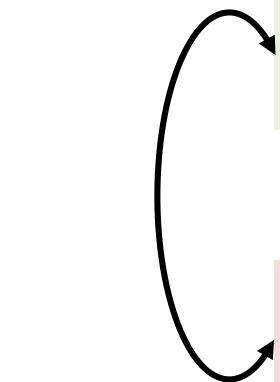
Iterative way of finding the optimum policy through sequence of policy evaluation and policy improvement

Iteration

For  $t = 0, \dots$  until convergence

For each state  $s$ :

$$V_{t+1}^{\pi}(s) \leftarrow \sum_{s'} T(s, \pi(s), s') \{R(s, \pi(s), s') + \gamma V_t^{\pi}(s')\}$$



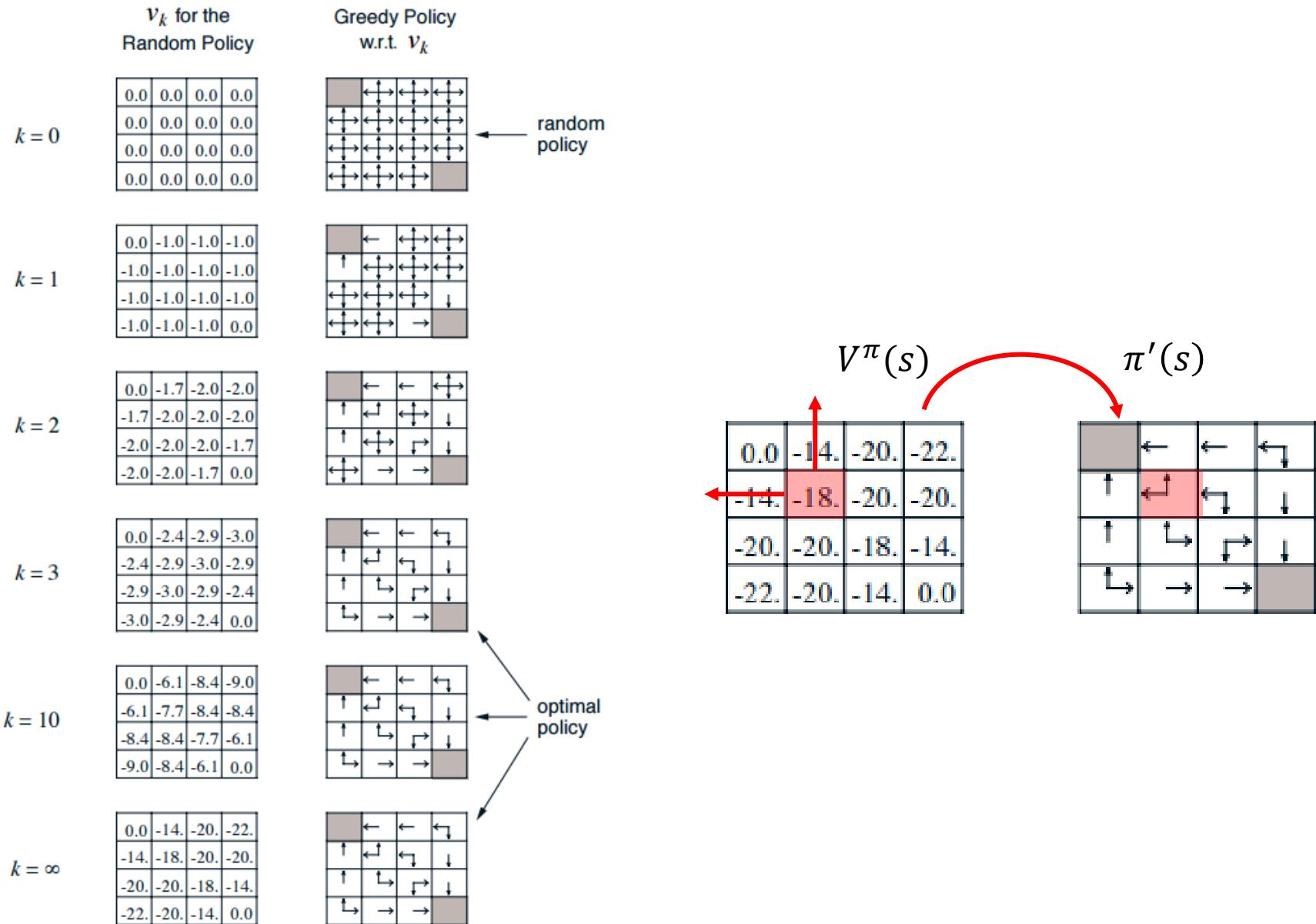
Converged state value function  $V^{\pi}(s)$

For each state  $s$ :

$$\pi'(s) = \operatorname{argmax}_{a \in \mathcal{A}(s)} Q^{\pi}(s, a)$$

$$= \operatorname{argmax}_{a \in \mathcal{A}(s)} \sum_{s'} T(s, a, s') [R(s, a, s') + \gamma V^{\pi}(s')]$$

# Policy Iteration

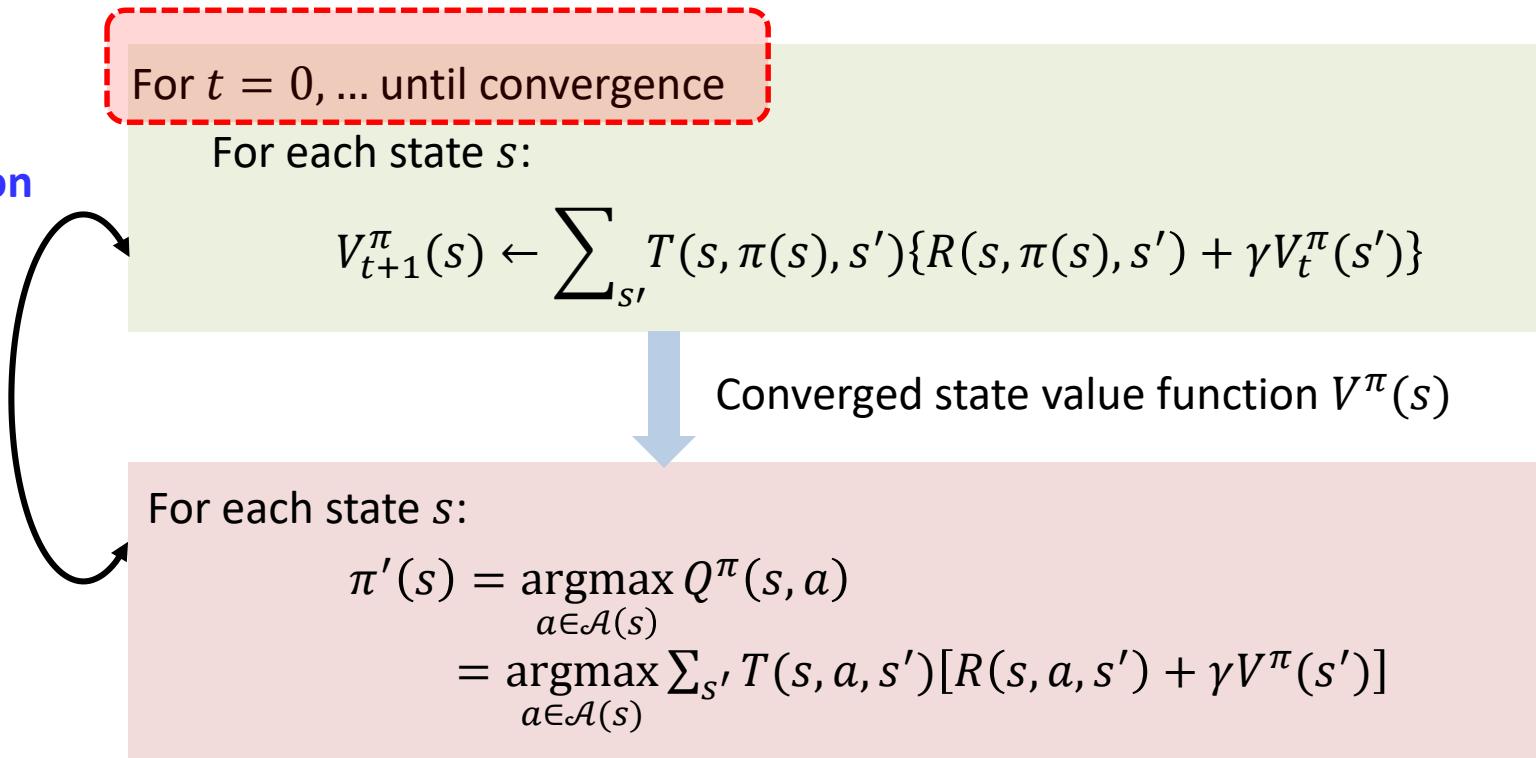


## Policy Iteration

### Policy iteration :

Iterative way of finding the optimum policy through sequence of policy evaluation and policy improvement

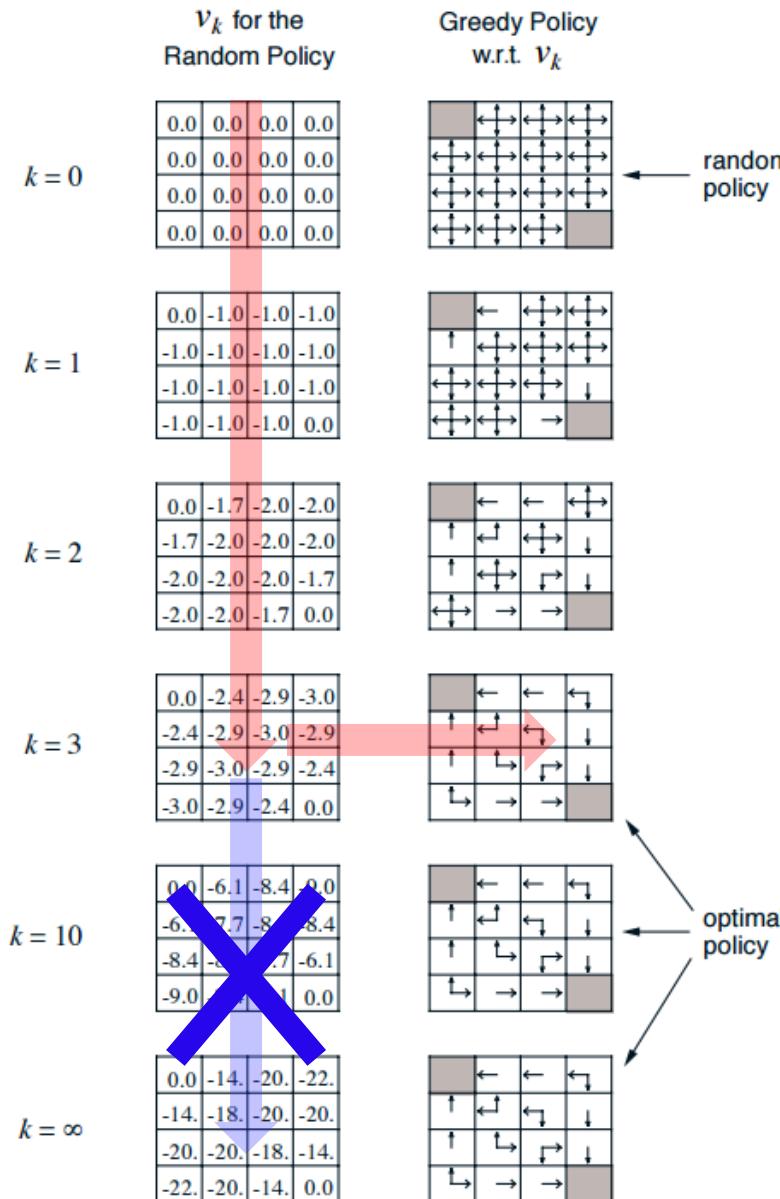
Iteration



### Issues :

Policy evaluation requires iterative computation, requiring multiple sweeps through the state set → slow to converge

# Policy Iteration



## Value Iteration

**Solution:**

- Stop policy evaluation after just one sweep (one backup of each state)
- Combine one sweep of policy evaluation and one sweep of policy improvement

For each state  $s$ :

$$V_{t+1}^{\pi}(s) \leftarrow \sum_{s'} T(s, \pi(s), s') \{R(s, \pi(s), s') + \gamma V_t^{\pi}(s')\}$$



For each state  $s$ :

$$\pi'(s) = \operatorname{argmax}_{a \in \mathcal{A}(s)} \sum_{s'} T(s, a, s') [R(s, a, s') + \gamma V_{t+1}^{\pi}(s')]$$



For each state  $s$ :

**Value Iteration**

$$V_{t+1}(s) \leftarrow \max_{a \in \mathcal{A}(s)} \sum_{s'} T(s, a, s') \{R(s, a, s') + \gamma V_t(s)\}$$

or, can be obtained simply by turning the Bellman optimality equation into an update rule :

$$V^*(s) = \max_{a \in \mathcal{A}(s)} \sum_{s'} T(s, a, s') \{R(s, a, s') + \gamma V^*(s')\}$$

## Value Iteration

### Value Iteration:

A method to compute the optimum state-value function  $V^*(s)$  by combining one sweep of policy evaluation and one sweep of policy improvement

### Algorithm

Initialize  $V(s) \leftarrow 0$  for all states  $s \in S$

Repeat

    For each state  $s$ :

$$V(s) \leftarrow \max_a \sum_{s'} T(s, a, s') \{R(s, a, s') + \gamma V(s)\}$$

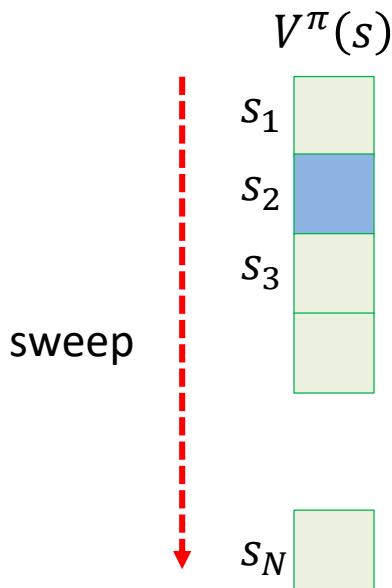
Until  $\max_{s \in S} |V_t(s) - V_{t-1}(s)| \leq \epsilon$

**Optimum policy** can be obtained from the converged  $V^*(s)$ :

$$\pi^*(s) = \operatorname{argmax}_{a \in \mathcal{A}(s)} \sum_{s'} T(s, a, s') \{R(s, a, s') + \gamma V^*(s)\}$$

## Asynchronous DP Algorithms

A major drawback to the DP methods is that they involve operations over the entire state set of the MDP



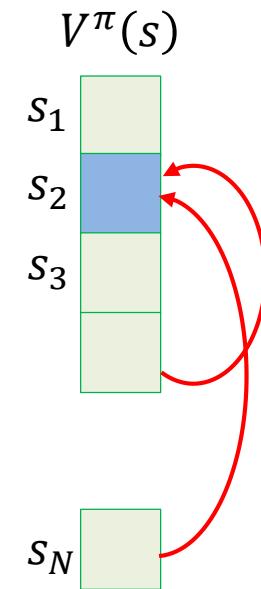
Conventional DP Algorithms

- Black mon game has  $10^{20}$  states
- Go game has  $3^{(19 \times 19)}$  states

...



- Take forever to sweep all states
- Does not improve policy until value functions are full backed up



Asynchronous DP Algorithms

- Back up the values of states in any order whatsoever, using whatever values of other states happen to be available
- Allow great flexibility in selecting states to which backup operations are applied
- Make it easier to intermix computation with real-time interaction: To solve a given MDP, we can run iterative DP algorithm at the same time that an agent is actually experiencing the MDP (Reinforcement Learning !!!!)

# Value Iteration

## Policy Evaluation

For  $t = 1, \dots$

For each state  $s$ :

$$V_{t+1}^{\pi}(s) \leftarrow \sum_{s'} T(s, \pi(s), s') \{R(s, \pi(s), s') + \gamma V_t^{\pi}(s')\}$$

Policy Iteration

## Policy Improvement

For each state  $s$ :

$$\pi'(s) = \operatorname{argmax}_{a \in \mathcal{A}(s)} \sum_{s'} T(s, a, s') [R(s, a, s') + \gamma V^{\pi}(s')]$$

## Value Iteration

For each state  $s$ :

$$V_{t+1}(s) \leftarrow \max_{a \in \mathcal{A}(s)} \sum_{s'} T(s, a, s') \{R(s, a, s') + \gamma V_t(s)\}$$

## Asynchronous Value iteration

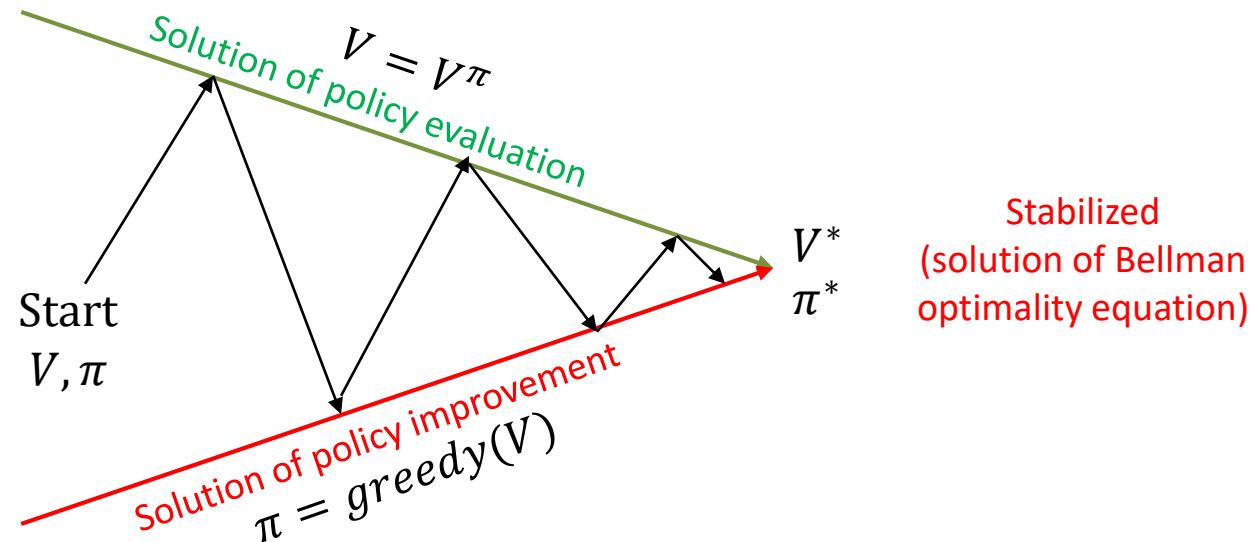
For any single state  $s$ :

$$V(s) \leftarrow \max_{a \in \mathcal{A}(s)} \sum_{s'} T(s, a, s') \{R(s, a, s') + \gamma V(s)\}$$

As long as both processes continue to update all states, the ultimate result is typically the same-convergence to the optimal value function and an optimal policy

## Generalized Policy Iteration

Generalized Policy Iteration :  
an general idea of interaction between policy evaluation and policy improvement processes



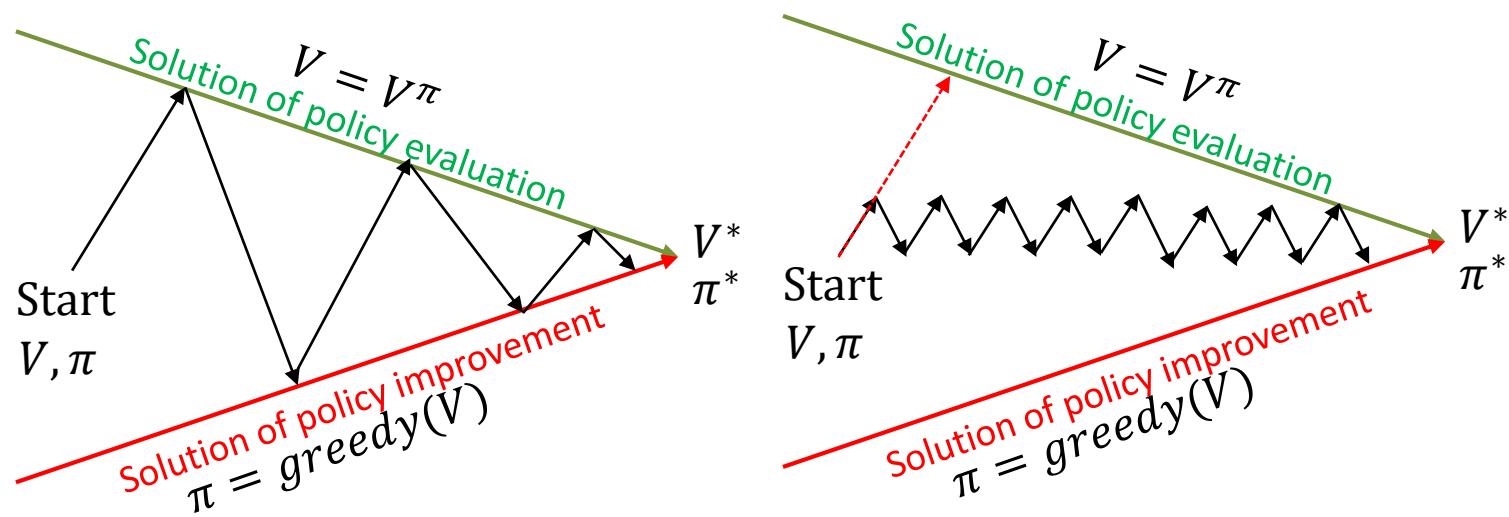
- Making policy greedy with respect to the value function typically makes the value function incorrect for the changed policy
- Making the value function consistent with the policy typically causes that policy no longer to be greedy
- In the long run, however, these two processes interact to find a single joint solution: the optimal value function and optimal policy

## Generalized Policy Iteration

### Generalized Policy Improvement (GPI)

Dynamic Programming  
“full backup”

Asynchronous  
Dynamic Programming



Asynchronous Dynamic Programming is a core concept in Reinforcement learning

## Efficiency of Dynamic Programming

$n$  : Number of states

$m$  : Number of actions

- A DP method is guaranteed to find an optimal policy in polynomial time even though the total number of deterministic policies is  $m^n$
- Linear programming methods can also be used to solve MDPs, and in some case their worst-case convergence are better than those of DP methods. But linear programming methods become impractical at a much smaller number of states than do DP methods
- A DP method is guaranteed to find an optimal
- For a large problem, asynchronous DP is more efficient
- DP methods update estimates of the values of states based on estimates of the values of successor states → **update is based on other update (bootstrapping)**

# L15. Reinforcement Learning (Monte Carlo Methods)

## Project

1. Define your own problem. It can be anything
2. Formulate the problem using mathematical expressions
3. Describe what type of data is required for the formulation
  - describe how to obtain data
  - describe how to process data
  - describe how to build models that are required for your formulation
4. Discuss what types of decision making methods can be used to solve the formulated problem
  - list more than two methods and discuss their pros and cons
  - discuss limitations in each method
5. Future Plan

Midterm 25%, Final 25%, Project 25%, HW 25% (5% each)

Format: Report (more than 2pages but less than 5 pages)

Evaluations: 25, 24, 23, 22, 21, 20 (Full score is 25)

Bonus points: up to 5 points

# From MDP to Reinforcement Learning



## Markov Decision Process (Offline)

- Have mental model of how the world works
- Find policy to collect the maximum rewards

$$\text{Solve } Q^*(s, a) = \sum_{s'} T(s, a, s')[R(s, a, s') + \gamma V^*(s')] \\ \text{Find } \pi^*(s) = \max_a Q^*(s, a)$$



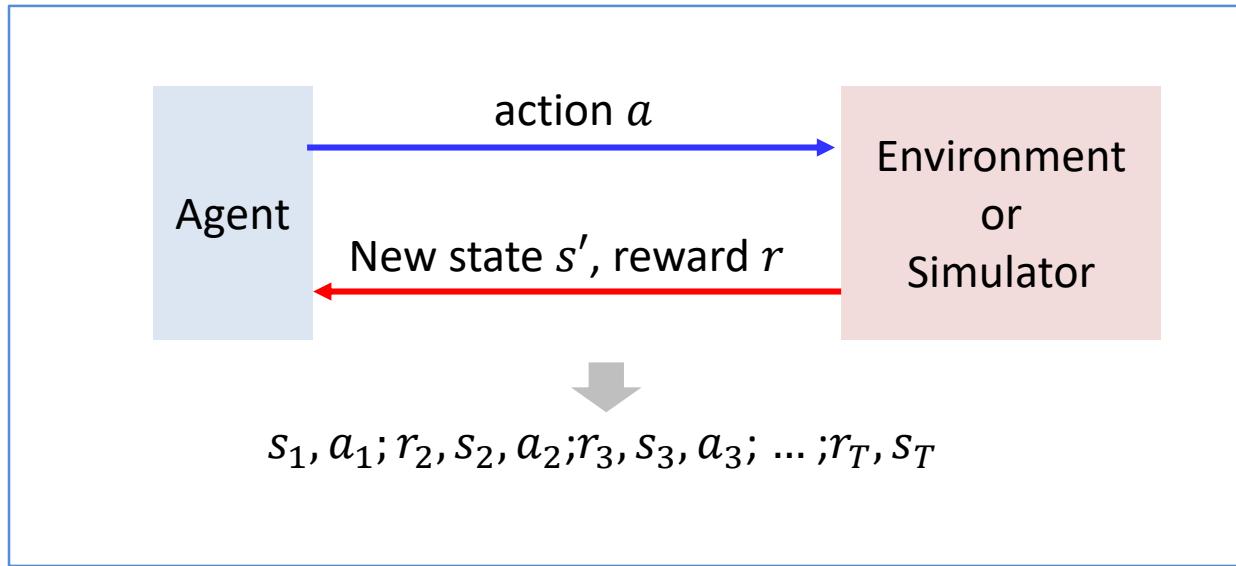
## Reinforcement Learning (Offline & Online)

- Don't know how the world works
- Perform a sequence of actions in the world to maximize the rewards

$$s_1, a_1, r_1; s_2, a_2, r_2; s_3, a_3, r_3; \dots; s_T, a_T, r_T \rightarrow Q^*(s, a) \rightarrow \pi^*(s)$$

- Reinforcement learning is really the way humans work:  
→ we go through life, taking various actions, getting feedback.  
→ We get rewarded for doing well and learn along the way.

## Reinforcement Learning Template



### Template for Reinforcement Learning

For  $t = 1, 2, 3, \dots$

Choose action  $a_t = \pi(s_t)$  (how?) : Decision making

Receive reward  $r_{t+1}$  and observe new state  $s_{t+1}$  (Environment)

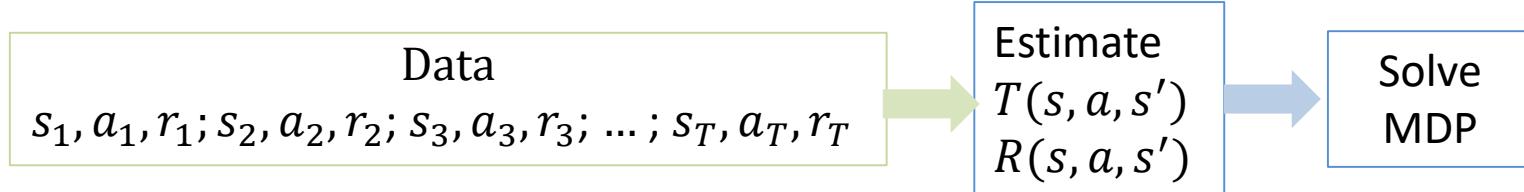
Update parameters associated with  $V(t), Q(s, a)$  (how?) : Learning

## Road Map

- **Monte Carlo Method (Sutton & Barto Ch.5)**
  - Model-Based Monte Carlo method
  - Model-free Monte Carlo method
    - Policy Evaluation
    - Policy Improvement
    - Policy Iteration (Monte Carlo control)
      - ✓ On-policy
      - ✓ Off-policy
- **Temporal Difference Learning (Sutton & Barto Ch.6)**
  - SARSA
  - Q-Learning

## Road Map

- Model-Based Reinforcement learning



- Model-FREE Reinforcement learning

		How to estimate $V^*(s)$ and $Q^*(s, a)$	
		Monte Carlo method	Temporal Difference methods
How to explore ?	Non-Bootstrap	Bootstrap	
	On-policy	On-policy Monte Carlo Control	SARSA
	Off-policy	Off-policy Monte Carlo Control	Q-Learning

• Episodic based      • Single-data-point based

## Model-Based Reinforcement Learning (Monte Carlo Methods)

### Key Idea : model-based learning

- Formulate the MDP using the estimated  $T(s, a, s')$  and  $R(s, a, s')$
- Solve the MDP using various solution methods, i.e., DP approach

$$\text{Solve } Q^*(s, a) = \sum_{s'} T(s, a, s')[R(s, a, s') + \gamma V^*(s')]$$

$$\text{Find } \pi^*(s) = \max_a Q^*(s, a)$$

- Data following policy  $\pi$

$$s_1, a_1, r_1; s_2, a_2, r_2; s_3, a_3, r_3; \dots; s_T, a_T, r_T$$

- Transition model

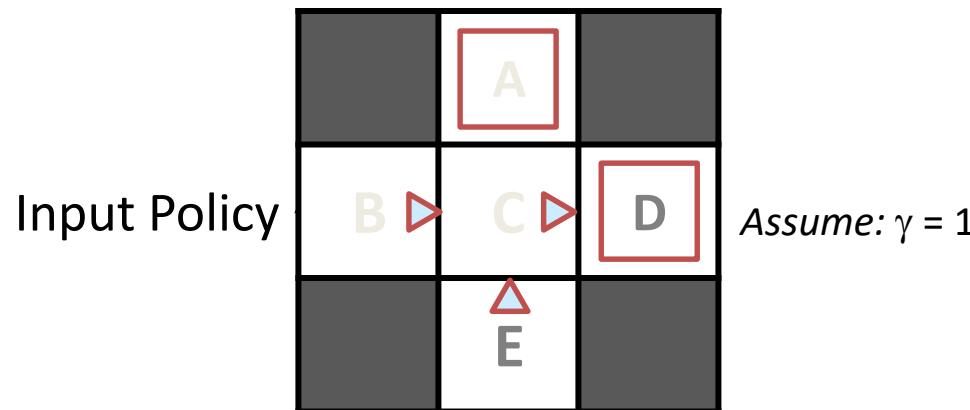
$$\hat{T}(s, a, s') = \frac{\#\text{times } (s, a, s') \text{ occurs}}{\#\text{times } (s, a) \text{ occurs}}$$

- Reward model

$$\hat{R}(s, a, s') = \text{average of } r \text{ in } (s, a, r, s')$$

## Model-Based Reinforcement Learning (Monte Carlo Methods)

### Example



- Observed Episodes (Training)

Episode 1

B, east, C, -1  
C, east, D, -1  
D, exit, x, +10

Episode 2

B, east, C, -1  
C, east, D, -1  
D, exit, x, +10

Episode 3

E, north, C, -1  
C, east, D, -1  
D, exit, x, +10

Episode 4

E, north, C, -1  
C, east, A, -1  
A, exit, x, -10

- Learned Model

$$T(s, a, s')$$

$T(B, \text{east}, C) = 1.00$   
 $T(C, \text{east}, D) = 0.75$   
 $T(C, \text{east}, A) = 0.25$

$$R(s, a, s')$$

$R(B, \text{east}, C) = -1$   
 $R(C, \text{east}, D) = -1$   
 $R(D, \text{exit}, x) = +10$

## From Model-Based to Model Free

$$Q^*(s, a) = \sum_{s'} \hat{T}(s, a, s') [\hat{R}(s, a, s') + \gamma V^*(s')]$$

↓  
Estimation

Why not estimating  $Q^*(s, a)$  directly  
instead of separately estimating  $\hat{T}(s, a, s')$  and  $\hat{R}(s, a, s')$  ?

$$\pi^*(s) = \operatorname{argmax}_a Q^*(s, a)$$

# Model-Free Monte Carlo Based Methods

How to estimate  $V^*(s)$  and  $Q^*(s, a)$

		Monte Carlo method	Temporal Difference methods
		Non-Bootstrap	Bootstrap
How to explore ?	On-policy	On-policy Monte Carlo Control	SARSA
	Off-policy	Off-policy Monte Carlo Control	Q-Learning

- Episodic based
- Single-data-point based

## Model-Free Monte Carlo Based Methods

- Monte Carlo methods require only experience-sample sequences of states, actions, and rewards from on-line or simulated interaction with an environment
- Monte Carlo methods are ways of solving the reinforcement learning problem based on **averaging sample returns (Monte Carlo)**

$$R(s, a) = \text{average of } r \text{ in } (s, a, r)$$

- Monte Carlo methods are incremental in an **episode-by-episode sense**, but not in a step-by-step sense  
→ after a final state has reached, reward appears, e.g., Go

### Dynamic Programming

- Policy Evaluation
- Policy Iteration
- Policy Improvement

Key ideas

### Monte Carlo Methods

- Policy Evaluation
- Policy Iteration
- Policy Improvement

## Monte Carlo Policy Evaluation

### Key Idea : Monte Carlo Policy Evaluation

- Learn the state-value function  $V^\pi(s)$  for a given policy  $\pi$
- The value of state is the expected utility - **expected accumulative future reward** starting from  $s$  and following the policy  $\pi$

Data (following policy  $\pi$ ):

$$s_t \in \mathcal{S} = \{s^1, s^2, s^3\}, a_t \in \mathcal{A} = \{a^1, a^2, a^3\}$$

Episode 1:  $s_1, a_1, r_1; s_2, a_2, r_2; s_3, a_3, r_3; \dots; s_T, a_T, r_T$

Episode 2:  $s_1, a_1, r_1; s_2, a_2, r_2; s_3, a_3, r_3; \dots; s_T, a_T, r_T$

Episode 3:  $s_1, a_1, r_1; s_2, a_2, r_2; s_3, a_3, r_3; \dots; s_T, a_T, r_T$

⋮

Utility  $u_t$ :

$$u_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots$$

$$u_t = r_t + r_{t+2} + \dots + r_T$$

Estimate  $V^\pi(s)$  or  $Q^\pi(s, a)$  :

$$V^\pi(s) = \text{average of } u_t \text{ where } s_t = s \text{ and following } \pi$$

Because the value being computed is dependent on the policy used to generate the data, we call this an **on-policy** algorithm.

## Monte Carlo Policy Evaluation

### Key Idea : Monte Carlo Policy Evaluation

- Learn the state-value function  $V^\pi(s)$  for a given policy  $\pi$
- The value of state is the expected utility - **expected accumulative future reward** starting from  $s$  and following the policy  $\pi$

$$s_t \in \mathcal{S} = \{s^1, s^2, s^3\}, a_t \in \mathcal{A} = \{a^1, a^2, a^3\}$$

(State, Action, Reward ) pairs generated by policy  $\pi$

Episode 1:  $(s^1, a^2, 1); (s^3, a^1, 5); (s^2, a^3, 3), (s^1, a^3, 10), (s^2, a^2, 2)$

Episode 2:  $(s^3, a^1, 5); (s^2, a^2, 2); (s^1, a^2, 1); (s^2, a^3, 3), (s^1, a^3, 10)$

Episode 3:  $(s^2, a^3, 3); (s^1, a^2, 1); (s^3, a^1, 5); (s^1, a^3, 10), (s^2, a^2, 2)$

$(\gamma = 1)$

Episode 1:  $V^\pi(s^1) = 1 + 5 + 3 + 10 + 2 = 21$

Episode 2:  $V^\pi(s^1) = 1 + 3 + 10 = 14$

First visit to  $s$

Episode 3:  $V^\pi(s^1) = 1 + 5 + 10 + 2 = 19$

$V^\pi(s^1)$  = average of accumulated reward over all episodes

$$= \frac{21+14+19}{3} = 14.6$$

## Monte Carlo Policy Evaluation

### Key Idea : Monte Carlo Policy Evaluation

- Learn the state-value function  $V^\pi(s)$  for a given policy  $\pi$
- The value of state is the expected utility - **expected accumulative future reward** starting from  $s$  and following the policy  $\pi$

$$s_t \in \mathcal{S} = \{s^1, s^2, s^3\}, a_t \in \mathcal{A} = \{a^1, a^2, a^3\}$$

(State, Action, Reward ) pairs generated by policy  $\pi$

Episode 1:  $(s^1, a^2, 1); (s^3, a^1, 5); (s^2, a^3, 3), (s^1, a^3, 10), (s^2, a^2, 2)$

Episode 2:  $(s^3, a^1, 5); (s^2, a^2, 2); (s^1, a^2, 1); (s^2, a^3, 3), (s^1, a^3, 10)$

Episode 3:  $(s^2, a^3, 3); (s^1, a^2, 1); (s^3, a^1, 5); (s^1, a^3, 10), (s^2, a^2, 2)$

$(\gamma = 1)$

Episode 1:  $V^\pi(s^1) = 1 + 5 + 3 + 10 + 2 = 21; V^\pi(s^3) = 10 + 2 = 12$

Episode 2:  $V^\pi(s^3) = 1 + 3 + 10 = 14; V^\pi(s^1) = 10 = 10$

Every visit to  $s$

Episode 3:  $V^\pi(s^1) = 1 + 5 + 10 + 2 = 19; V^\pi(s^3) = 10 + 2 = 12$

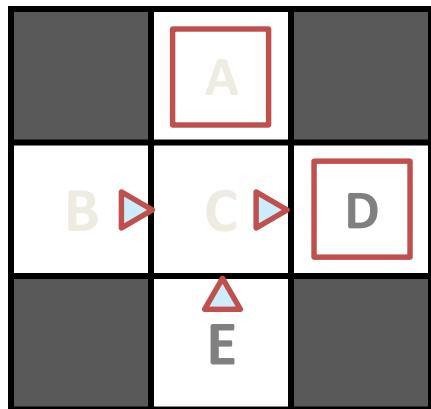
$V^\pi(s^1)$  = average of accumulated reward over all episodes

$$= \frac{21+12+14+10+19+12}{6} = 15$$

## Monte Carlo Policy Evaluation

### Example

Input Policy  $\pi$



Assume:  $\gamma = 1$

Observed Episodes (Training)

Episode 1

B, east, C, -1  
C, east, D, -1  
D, exit, x, +10

Episode 2

B, east, C, -1  
C, east, D, -1  
D, exit, x, +10

Episode 3

E, north, C, -1  
C, east, D, -1  
D, exit, x, +10

Episode 4

E, north, C, -1  
C, east, A, -1  
A, exit, x, -10

Output Values

	-10	
A	+4	+10
B	+8	D
C		
E	-2	

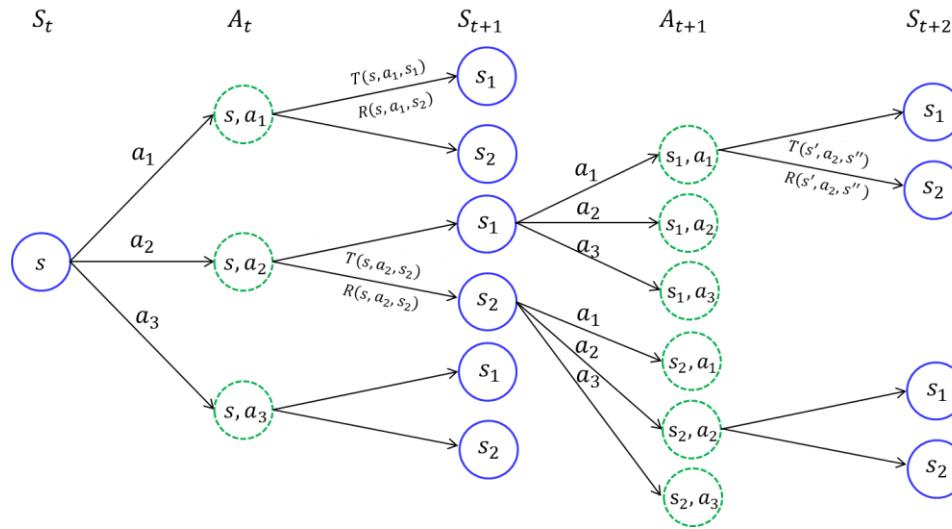
### Example

- What's good about direct evaluation?
  - It's easy to understand
  - It doesn't require any knowledge of  $T$ ,  $R$
  - It eventually computes the correct average values, using just sample transitions
- What bad about it?
  - It wastes information about state connections
  - Each state must be learned separately
  - So, it takes a long time to learn

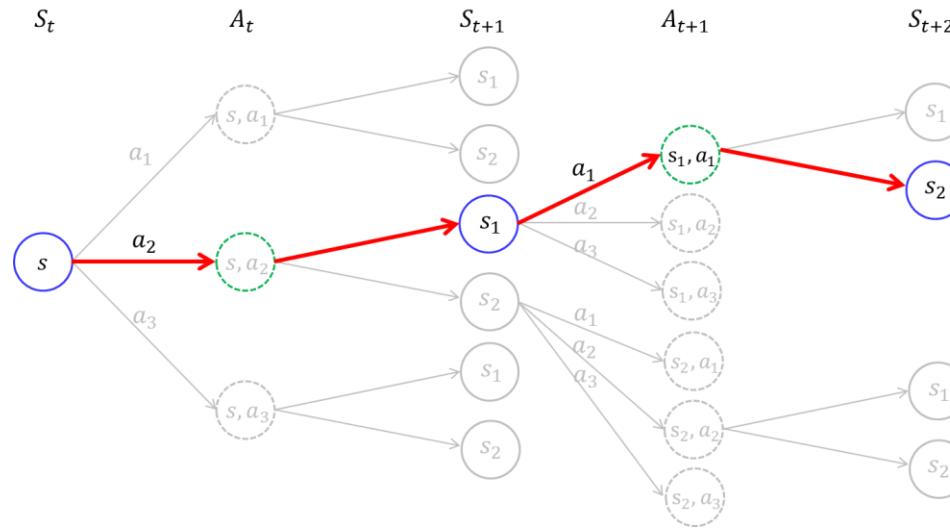
Output Values

		-10 A	
+8 B		+4 C	+10 D
		-2 E	

## Monte Carlo Policy Evaluation



## Monte Carlo Policy Evaluation



- Estimates for each state are independent
  - ✓ The estimate for one state does not build upon the estimate of any other state  
(No bootstrap)
- The computational expense of estimating the value of a single state is independent of the number of states
  - ✓ Attractive when one requires the value of only a subset of the states

## Monte Carlo Estimation of Action Values

- Without a model, state value function  $V^\pi(s)$  is not enough
  - ✓ We need  $T(s, a, s')$  and  $R(s, a, s')$  to compute the optimum policy:

$$Q^*(s, a) = \sum_{s'} T(s, a, s') \{R(s, a, s') + \gamma V^*(s')\}$$

$$a^* = \pi^*(s) = \operatorname{argmax}_{a \in \mathcal{A}(s)} Q^*(s, a)$$

- One of primary goals for Monte Carlo method is to estimate  $Q^*(s, a)$

## Monte Carlo Estimation of Action Values

### Key Idea : Monte Carlo Policy Evaluation

- Learn the action-value function  $Q^\pi(s, a)$  for a given policy  $\pi$
- The value of action-state is the expected utility - **expected accumulative future reward** starting from  $s$  and following the policy  $\pi$

Data (following policy  $\pi$ ):

$$s_t \in \mathcal{S} = \{s^1, s^2, s^3\}, a_t \in \mathcal{A} = \{a^1, a^2, a^3\}$$

Episode 1:  $s_1, a_1, r_1; s_2, a_2, r_2; s_3, a_3, r_3; \dots; s_T, a_T, r_T$

Episode 2:  $s_1, a_1, r_1; s_2, a_2, r_2; s_3, a_3, r_3; \dots; s_T, a_T, r_T$

Episode 3:  $s_1, a_1, r_1; s_2, a_2, r_2; s_3, a_3, r_3; \dots; s_T, a_T, r_T$

⋮

Utility  $u_t$ :

$$u_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots$$

$$u_t = r_t + r_{t+2} + \dots + r_T$$

Estimate  $Q^\pi(s, a)$  :

$Q^\pi(s, a) = \text{average of } u_t \text{ where } s_t = s, a_t = a \text{ and following } \pi$

## Monte Carlo Policy Evaluation

### Key Idea : Monte Carlo Policy Evaluation

- Learn the state-value function  $Q^\pi(s, a)$  for a given policy  $\pi$
- The value of action-state is the expected utility - **expected accumulative future reward** starting from  $s$  and following the policy  $\pi$

$$s_t \in \mathcal{S} = \{s^1, s^2, s^3\}, a_t \in \mathcal{A} = \{a^1, a^2, a^3\}$$

(State, Action, Reward ) pairs generated by policy  $\pi$

Episode 1:  $(s^1, a^2, 1); (s^3, a^1, 5); (s^2, a^3, 3), (s^1, a^3, 10), (s^2, a^2, 2)$

Episode 2:  $(s^1, a^1, 5); (s^2, a^2, 2); (s^1, a^2, 1); (s^2, a^3, 3), (s^1, a^3, 10)$

Episode 3:  $(s^2, a^3, 3); (s^1, a^2, 1); (s^3, a^1, 5); (s^1, a^3, 10), (s^2, a^2, 2)$

$(\gamma = 1)$

Episode 1:  $Q^\pi(s^1, a^2) = 1 + 5 + 3 + 10 + 2 = 21$

Episode 2:  $Q^\pi(s^1, a^2) = 1 + 3 + 10 = 14$

Episode 3:  $Q^\pi(s^1, a^2) = 1 + 5 + 10 + 2 = 19$

First visit to  $s$

$Q^\pi(s^1, a^2) = \text{average of accumulated reward over all episodes}$

$$= \frac{21+14+19}{3} = 14.6$$

## Incremental Formulation

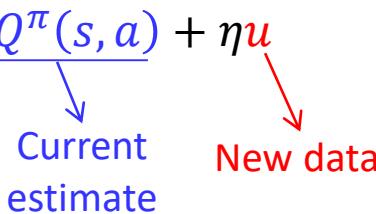
$s_1, a_1, r_1; s_2 = s, a_2 = a, r_2; s_3, a_3, r_3; \dots; s_T, a_T, r_T$   
 $s_1 = s, a_1 = a, r_1; s_2, a_2, r_2; s_3, a_3, r_3; \dots; s_T, a_T, r_T$   
 $s_1, a_1, r_1; s_2, a_2, r_2; s_3 = s, a_3 = a, r_3; \dots; s_T, a_T, r_T$   
 $\vdots$   
 $s_1, a_1, r_1; s_2, a_2 = a, r_2; s_3, a_3, r_3; \dots; s_T, a_T, r_T$

Episode 1  
 Episode 2  
 Episode 3  
 Episode  $\infty$

For each episode,  
 We can get state and action pair  
 $(s, a)$  and the following accumulated  
 reward (utility)  $u$ :  $(s, a) \rightarrow u$

### Incremental formulation (convex combination):

On each  $(s, a, u)$  for a single episode:

$$Q^\pi(s, a) \leftarrow (1 - \eta) \underbrace{Q^\pi(s, a)}_{\text{Current estimate}} + \eta u$$


where  $\eta = \frac{1}{1 + (\# \text{updates to } (s, a))}$

### Stochastic gradient form :

On each  $(s, a, u)$  for a single episode:

$$Q^\pi(s, a) \leftarrow Q^\pi(s, a) - \eta(Q^\pi(s, a) - u)$$

The incremental formulations can be thought of as a way to solve the following least square estimation in an online manner

$$\min_{Q^\pi} \sum_{(s, a, u)} (Q^\pi(s, a) - u)^2$$

## Issue of computing $Q^\pi(s, a)$

- If  $\pi$  is a deterministic policy, many relevant state –action pairs may never be visited
  - ✓ then in following  $\pi$ , one will observe returns only for one of the actions from each state, i.e., we won't even see  $(s, a)$  if  $a \neq \pi(s)$
  - ✓ To improve the policy  $\pi$ , we need to **compare** the state-action values of all the possible actions,  $Q^\pi(s, a_1), Q^\pi(s, a_2), \dots$
- To estimate  $Q^\pi(s, a)$  reliably, we need a large number of episodes
  - ✓ *Infinite* number of episodes are required for accurate estimations

## How to resolve?

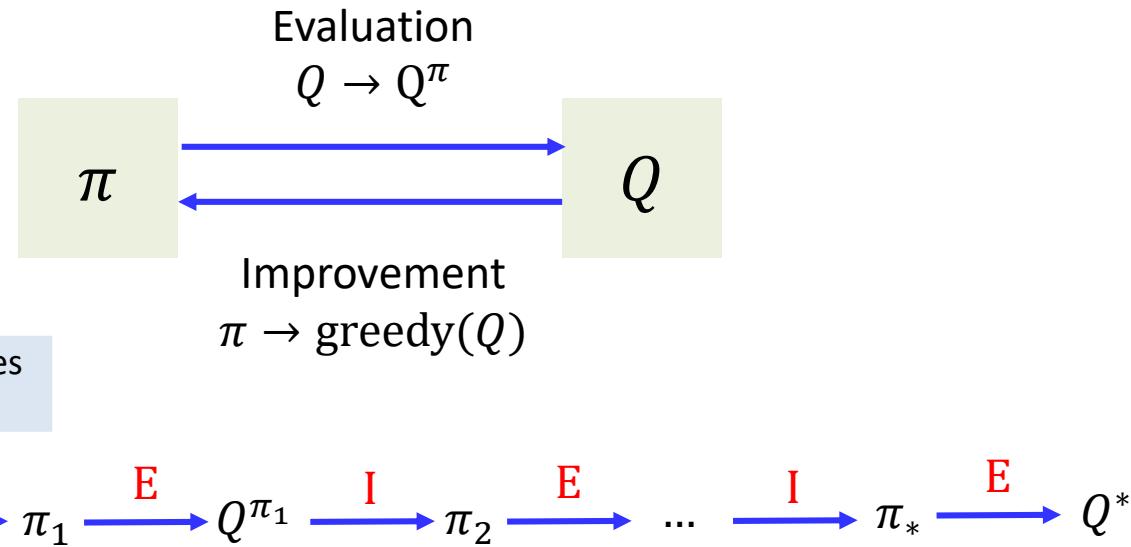
- Exploring starts + infinite number of episodes
  - ✓ Start from an arbitrary state-action pair  $(s, a)$  with a non-zero probability
  - ✓ Guarantees **that all state-action pair will be visited an infinite number of times in the limit of an infinite number of episodes**
  - ✓ Cannot be used when learning directly from experience
- Stochastic Policy + infinite number of episodes
  - ✓ **Policy with a nonzero probability of selecting all actions**
  - ✓ Guarantees that all state-action pair will be visited an infinite number of times in the limit of an infinite number of episodes

We need an efficient exploration strategy!

## Monte Carlo Control

### Key Idea : Monte Carlo Control

- Idea of generalized policy iteration (GPI)
- Monte Carlo Policy Evaluation + Policy improvement



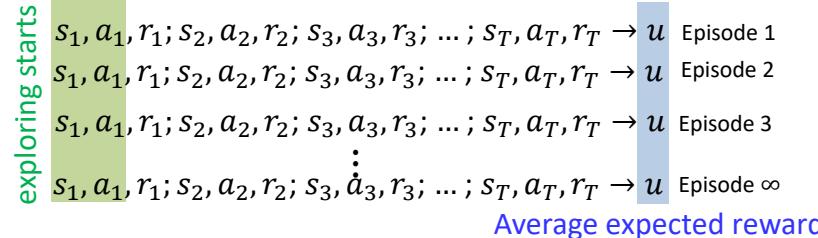
- **Policy Evaluation**
  - ✓ The value function is repeatedly altered to more closely approximate the value function for the current policy  $\pi$
- **Policy Improvement**
  - ✓ The policy is repeatedly improved with respect to the current action value function  $Q$

## Monte Carlo Control



### E Policy evaluation:

- The value function is repeatedly altered to more closely approximate the value function for the current policy  $\pi$
  - Infinite number of episodes under policy  $\pi$
  - The episodes are generated with exploring starts
- Converge to exact  $Q^\pi(s, a)$



### I Policy Improvement:

- Policy improvement can be done by constructing each  $\pi_{k+1}$  as the greedy policy with respect to  $Q^{\pi_k}$

$$\pi_{k+1}(s) = \operatorname{argmax}_a Q^{\pi_k}(s, a)$$

$$\begin{aligned} Q^{\pi_k}(s, \pi_{k+1}(s)) &= Q^{\pi_k}\left(s, \operatorname{argmax}_a Q^{\pi_k}(s, a)\right) \\ &= \max_a Q^{\pi_k}(s, a) \\ &\geq Q^{\pi_k}(s, \pi_k(s)) \\ &= V^{\pi_k}(s) \end{aligned}$$

$V^{\pi_{k+1}}(s) \geq V^{\pi_k}(s)$  for all  $s$   
 (Policy improvement theorem)

Overall process converges to an optimal policy and the optimal value function

## Make Monte Carlo Control Practical

### Assumptions:

- *Infinite* number of episodes
- Exploring starts

→ **Relax** these two assumptions



- **Relaxing “Infinite number of episodes”**
  - ✓ Policy improvement with an early stop (Generalized policy improvement concept)
  - ✓ The “in place” version of value iteration
- Relaxing “exploring starts”
  - ✓ Discussed later

## Monte Carlo control algorithm assuming exploring starts

### Algorithm : Monte Carlo ES Control

Initialize, for all  $s \in S, a \in A(s)$

$$Q(s, a) \leftarrow \text{arbitrary}$$

$$\pi(s) \leftarrow \text{arbitrary}$$

$$U(s, a) \leftarrow \text{empty list}$$

Repeat forever:

(a) Generate *an episode* using *exploring starts* and  $\pi$

(b) For each pair  $(s, a)$  appearing in the episode:

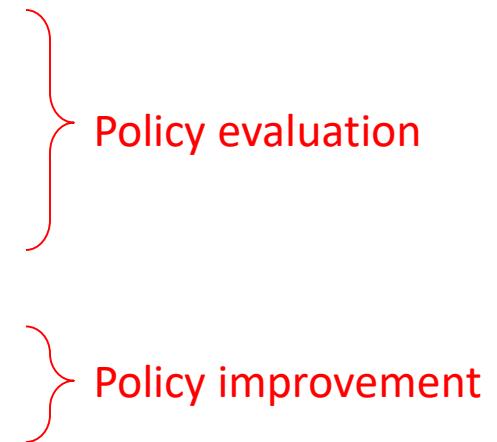
$U \leftarrow \text{utility following the first occurrence of } s, a$

Append  $U$  to  $U(s, a)$

$Q(s, a) \leftarrow \text{average}(U(s, a))$

(c) For each  $s$  in the episode:

$\pi(s) \leftarrow \underset{a}{\operatorname{argmax}} Q(s, a)$



In a single episode, both Policy evaluation and policy improvement proceed together

## Make Monte Carlo Control Practical

### Assumptions:

- *Infinite* number of episodes
- Exploring starts

→ **Relax** these two assumptions



- Relaxing “Infinite number of episodes”
  - ✓ Policy improvement with an early stop (Generalized policy improvement concept)
  - ✓ The “in place” version of value iteration
- **Relaxing “exploring starts”**
  - ✓ The only general way to ensure that all actions are selected infinitely often is for the agent to continue to select them
    - ❖ On-policy methods
    - ❖ Off-policy methods

## Monte Carlo Control

### On policy algorithm



- On-policy methods attempt to evaluate or improve the policy that is used to make decisions (generate data)
  - ✓ Policy is soft:  $\pi(s, a) > 0$  for all  $s \in S$  and  $a \in A(s)$

### Off policy algorithm



- Behavioral policy is used to generate behavior
- Estimation policy is evaluated and improved
  - ✓ Estimation policy may be deterministic, while the behavior policy can continue to sample all possible actions

## On-policy Monte Carlo Control

### Key Idea : On-policy methods: Soft policies

- attempt to evaluate or improve the policy that is used to make decisions
- $\pi(s, a) > 0$  for all  $s \in S$  and  $a \in A(s)$

$\epsilon$  – soft policy

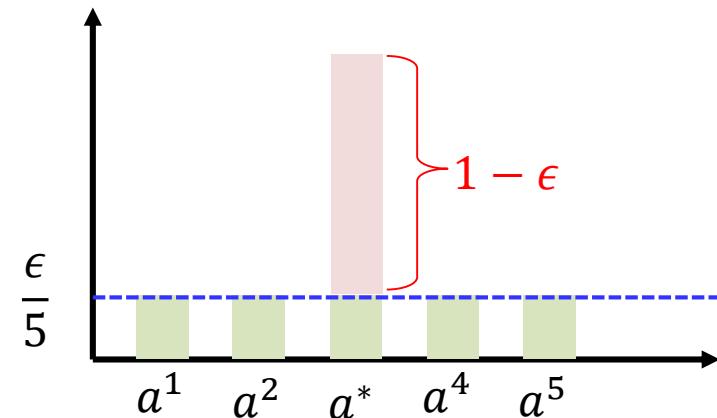
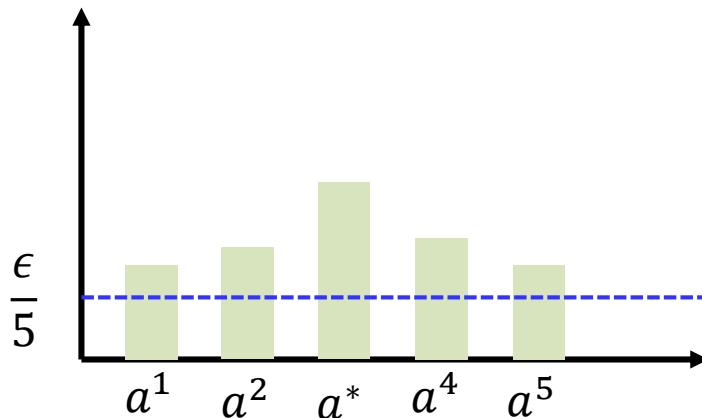
$$\pi(s, a) \geq \frac{\epsilon}{|A(s)|} \text{ for all } a$$

$$\sum_a \pi(s, a) = 1$$

$\epsilon$  – greedy policy

$$\pi(s, a) = \begin{cases} 1 - \epsilon + \epsilon/|A(s)| & \text{If } a = a^* \\ \epsilon/|A(s)| & \text{If } a \neq a^* \end{cases}$$

$$\text{Total probability} = \left(1 - \epsilon + \frac{\epsilon}{|A(s)|}\right) 1 + \frac{\epsilon}{|A(s)|} (|A(s)| - 1) = 1$$



## On-policy Monte Carlo Control

Any  $\epsilon - \text{greedy}$  policy  $\pi'$  respect to  $Q^\pi$  is an improvement over any  $\epsilon - \text{soft}$  policy  $\pi$

Let  $\pi'$  be the  $\epsilon - \text{greedy}$  policy,

$$\begin{aligned} Q^\pi(s, \pi'(s, a)) &= \sum_a \pi'(s, a) Q^\pi(s, a) \\ &= \frac{\epsilon}{|A(s)|} \sum_a Q^\pi(s, a) + (1 - \epsilon) \max_a Q^\pi(s, a) \\ &\geq \frac{\epsilon}{|A(s)|} \sum_a Q^\pi(s, a) + (1 - \epsilon) \sum_a \frac{\pi(s, a) - \frac{\epsilon}{|A(s)|}}{1 - \epsilon} Q^\pi(s, a) \\ &= \frac{\epsilon}{|A(s)|} \sum_a Q^\pi(s, a) - \frac{\epsilon}{|A(s)|} \sum_a Q^\pi(s, a) + \sum_a \pi(s, a) Q^\pi(s, a) \\ &= V^\pi(s) \end{aligned}$$

### Recall Policy improvement Theorem

Policy improvement must give us a strictly better policy  $\pi'(s)$  than the older policy  $\pi(s)$  except when the original policy is already optimal  $\pi(s, a) = \pi^*(s, a)$

$$Q^\pi(s, \pi'(s, a)) \geq V^\pi(s) \rightarrow V^{\pi'}(s) \geq V^\pi(s)$$

Thus, by the policy improvement theorem,  $\pi' \geq \pi$  (i.e.,  $V^{\pi'}(s) \geq V^\pi(s)$  for all  $s \in S$ )

## On-policy Monte Carlo Control

### Algorithm : $\epsilon$ – soft On-Policy Monte Carlo Control

Initialize, for all  $s \in \mathcal{S}, a \in \mathcal{A}(s)$

$$Q(s, a) \leftarrow \text{arbitrary}$$

$\pi \leftarrow \text{an arbitrary } \epsilon\text{-soft policy}$

$$U(s, a) \leftarrow \text{empty list}$$

Repeat forever:

(a) Generate *an episode* using  $\pi$

(b) For each pair  $(s, a)$  appearing in the episode:

$U \leftarrow \text{utility following the first occurrence of } s, a$

Append  $U$  to  $U(s, a)$

$Q(s, a) \leftarrow \text{average}(U(s, a))$

$\} \quad \text{Policy evaluation}$

(c) For each  $s$  in the episode:

$a^* \leftarrow \underset{a \in \mathcal{A}(s)}{\operatorname{argmax}} Q(s, a)$

For all  $a \in \mathcal{A}(s)$

$$\pi(s, a) = \begin{cases} 1 - \epsilon + \epsilon / |\mathcal{A}(s)| & \text{If } a = a^* \\ \epsilon / |\mathcal{A}(s)| & \text{If } a \neq a^* \end{cases}$$

$\} \quad \text{Policy improvement}$

## Monte Carlo control algorithm

Black Jack Example

# Model-Free Monte Carlo Based Methods

How to estimate  $V^*(s)$  and  $Q^*(s, a)$

		Monte Carlo method	Temporal Difference methods
		Non-Bootstrap	Bootstrap
How to explore ?	On-policy	On-policy Monte Carlo Control	SARSA
	Off-policy	Off-policy Monte Carlo Control	Q-Learning

- Episodic based
- Single-data-point based

## Off-policy Monte Carlo Control

### Evaluate One Policy While Following Another

Episodes generated by  $\pi'$ :

$$s_t \in \mathcal{S} = \{s^1, s^2, s^3\}, a_t \in \mathcal{A} = \{a^1, a^2, a^3\}$$

Episode 1:  $s_1, a_1, r_1; s_2, a_2, r_2; s_3, a_3, r_3; \dots; s_T, a_T, r_T$

Episode 2:  $s_1, a_1, r_1; s_2, a_2, r_2; s_3, a_3, r_3; \dots; s_T, a_T, r_T$

Episode 3:  $s_1, a_1, r_1; s_2, a_2, r_2; s_3, a_3, r_3; \dots; s_T, a_T, r_T$

$\vdots$

Estimate  $Q^\pi(s, a)$ :

$$Q^\pi(s, a) = \text{average of } u_t \text{ where } s_t = s, a_t = a \text{ and following } \pi$$

Is it possible?

Yes, if  $\pi'(s, a) > 0 \rightarrow \pi(s, a) > 0$ :

In order to use episodes from  $\pi'$  to estimate values for  $\pi$ , we require that every action taken under  $\pi$  is also taken, at least occasionally, under  $\pi'$

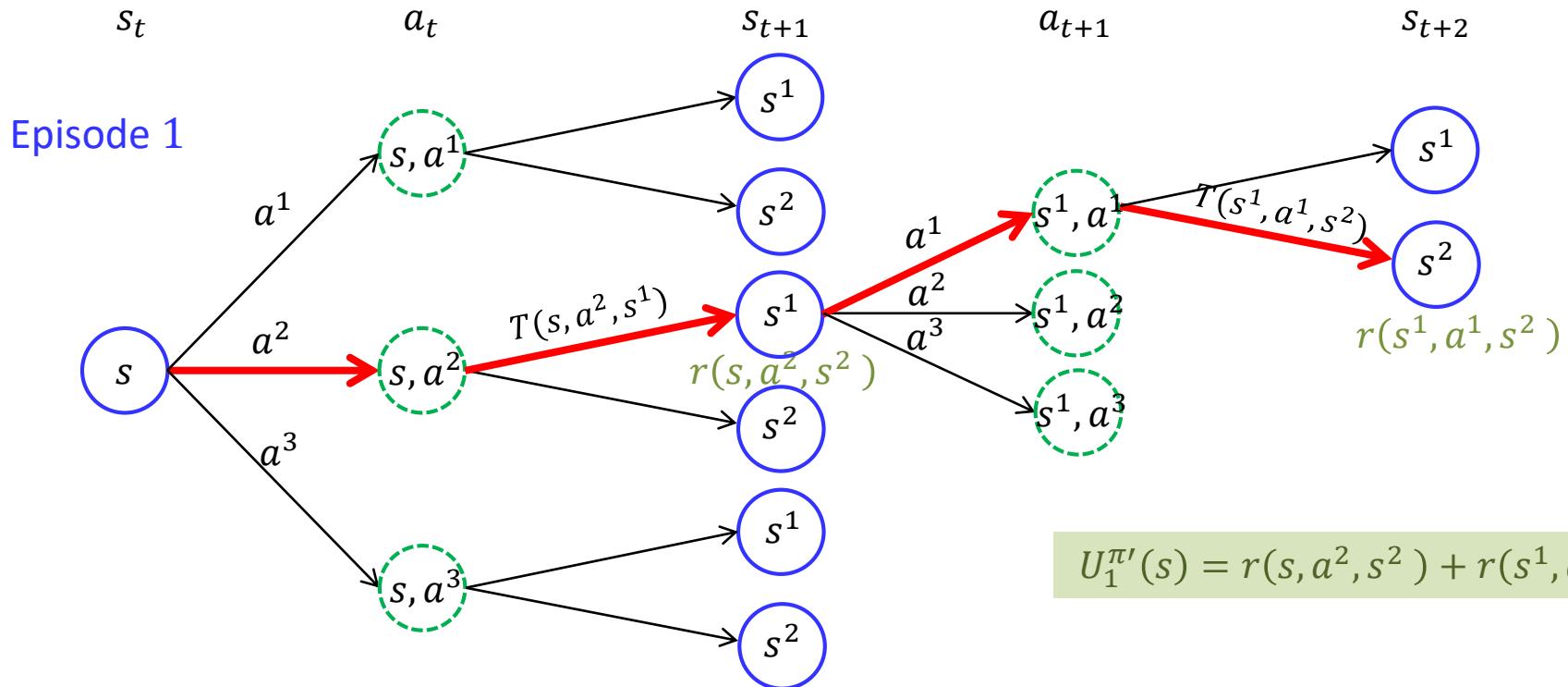
### Key Idea : Off-policy algorithm

- Follows the behavior policy while learning about and improving the estimation policy
- *Behavior* policy  $\pi'(s, a)$ 
  - ✓ The policy used to generate behavior
  - ✓ Requires that the behavior policy have a nonzero probability of selecting all actions that might be selected by the estimation policy (e.g.,  $\epsilon$  – soft policy )
- *Estimation* policy  $\pi(s, a)$ 
  - ✓ The policy that is evaluated and improved
  - ✓  $\pi$  can be deterministic
  - ✓  $\pi$  can be the greedy policy with respect to  $Q$  (an estimation  $Q^\pi$ )

### Disadvantages

→ Learning can be slow

## Off-policy Monte Carlo Control :Evaluate One Policy While Following Another

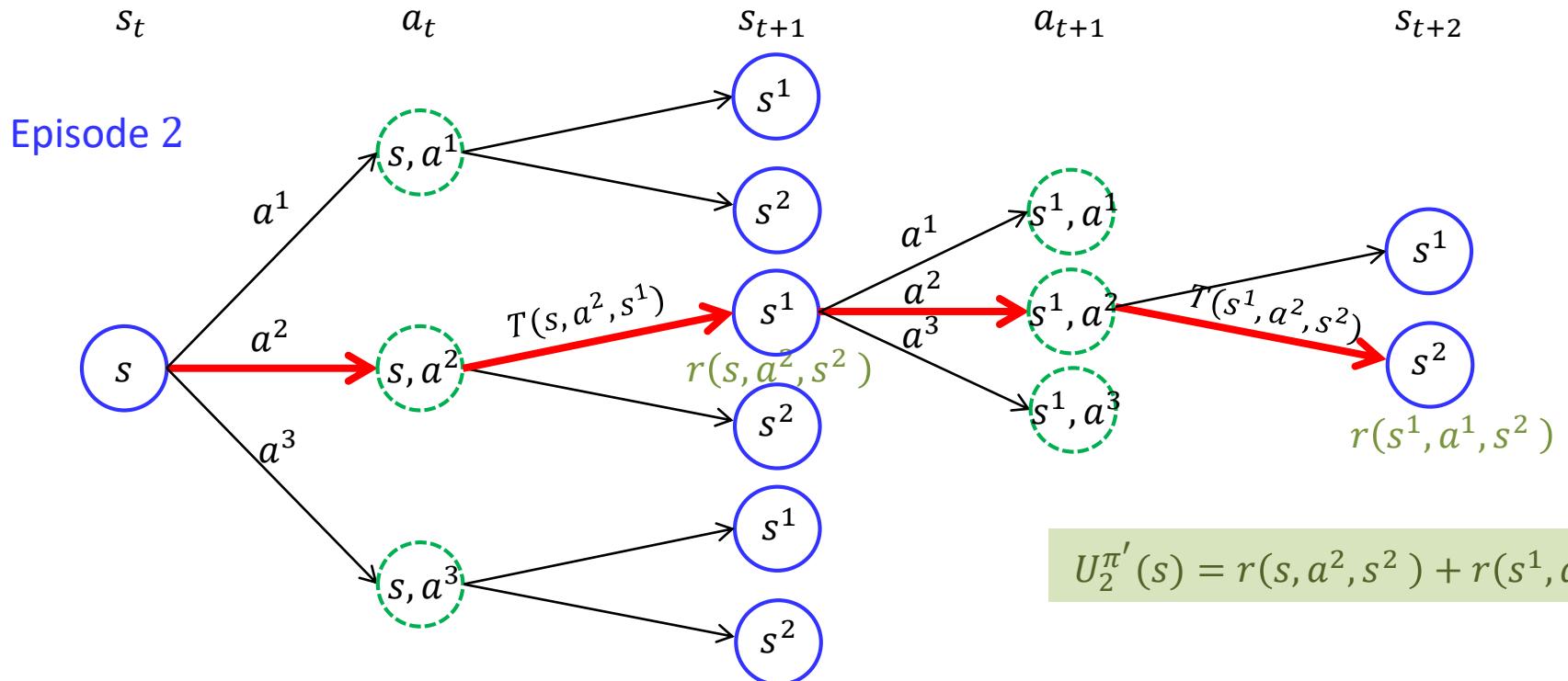


$$p_1(s_t = s) = \prod_{k=t}^{T_1(s)-1} \pi(s_k, a_k) T(s_k, a_k, s_{k+1}) = \pi(s, a^2) T(s, a^2, s^1) \pi(s^1, a^1) T(s^1, a^1, s^2)$$

$$p'_1(s_t = s) = \prod_{k=t}^{T_1(s)-1} \pi'(s_k, a_k) T(s_k, a_k, s_{k+1}) = \pi'(s, a^2) T(s, a^2, s^1) \pi'(s^1, a^1) T(s^1, a^1, s^2)$$

$$\frac{p_1(s)}{p'_1(s)} = \prod_{k=t}^{T_1(s)-1} \frac{\pi(s_k, a_k) T(s_k, a_k, s_{k+1})}{\pi'(s_k, a_k) T(s_k, a_k, s_{k+1})} = \prod_{k=t}^{T_1(s)-1} \frac{\pi(s_k, a_k)}{\pi'(s_k, a_k)} = \frac{\pi(s, a^2) \pi(s^1, a^1)}{\pi'(s, a^2) \pi'(s^1, a^1)}$$

## Off-policy Monte Carlo Control :Evaluate One Policy While Following Another

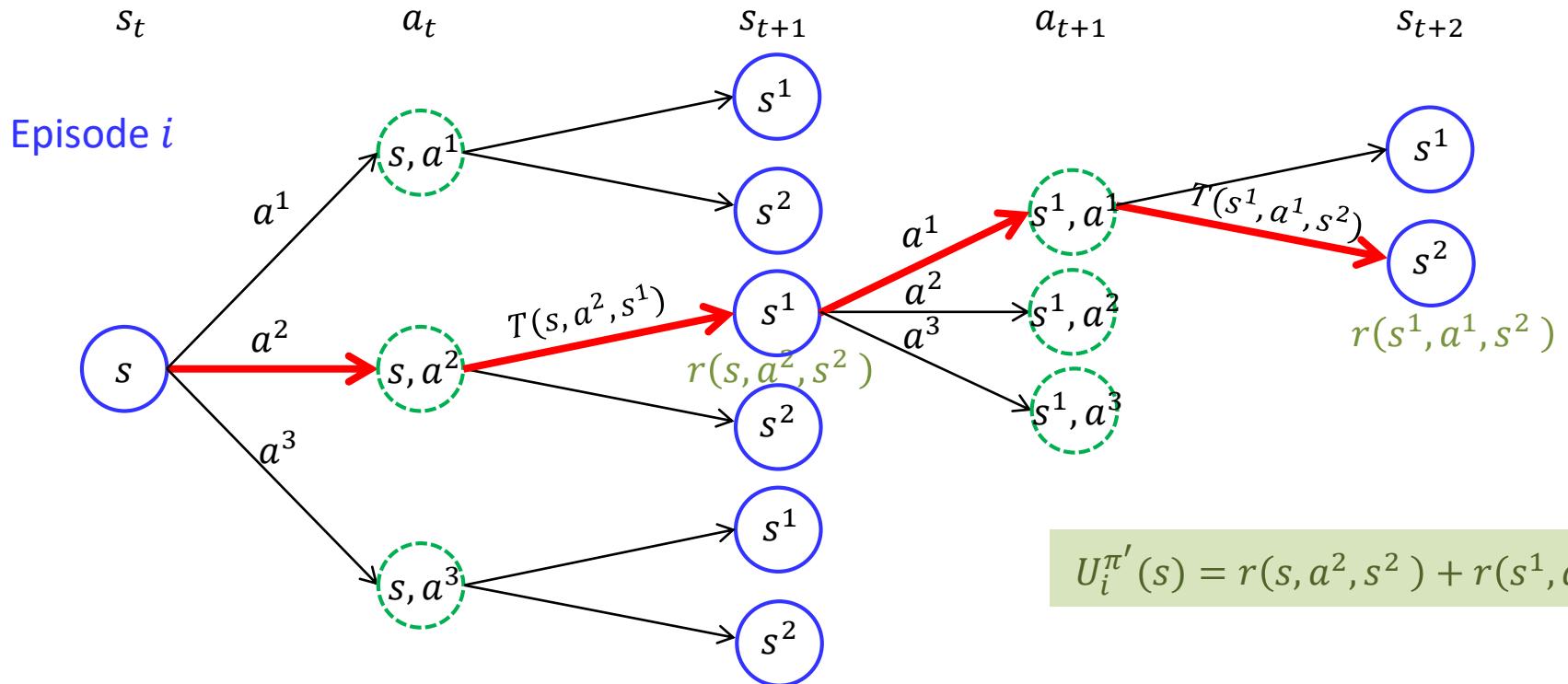


$$p_2(s_t = s) = \prod_{k=t}^{T_2(s)-1} \pi(s_k, a_k) T(s_k, a_k, s_{k+1}) = \pi(s, a^2) T(s, a^2, s^1) \pi(s^1, a^2) T(s^1, a^2, s^2)$$

$$p'_2(s_t = s) = \prod_{k=t}^{T_2(s)-1} \pi'(s_k, a_k) T(s_k, a_k, s_{k+1}) = \pi'(s, a^2) T(s, a^2, s^1) \pi'(s^1, a^2) T(s^1, a^2, s^2)$$

$$\frac{p_2(s)}{p'_2(s)} = \prod_{k=t}^{T_2(s)-1} \frac{\pi(s_k, a_k) T(s_k, a_k, s_{k+1})}{\pi'(s_k, a_k) T(s_k, a_k, s_{k+1})} = \prod_{k=t}^{T_2(s)-1} \frac{\pi(s_k, a_k)}{\pi'(s_k, a_k)} = \frac{\pi(s, a^2) \pi(s^1, a^2)}{\pi'(s, a^2) \pi'(s^1, a^2)}$$

## Off-policy Monte Carlo Control :Evaluate One Policy While Following Another



$$p_i(s_t = s) = \prod_{k=t}^{T_i(s)-1} \pi(s_k, a_k) T(s_k, a_k, s_{k+1}) = \pi(s, a^2) T(s, a^2, s^1) \pi(s^1, a^1) T(s^1, a^1, s^2)$$

$$p'_i(s_t = s) = \prod_{k=t}^{T_i(s)-1} \pi'(s_k, a_k) T(s_k, a_k, s_{k+1}) = \pi'(s, a^2) T(s, a^2, s^1) \pi'(s^1, a^1) T(s^1, a^1, s^2)$$

$$\frac{p_i(s)}{p'_i(s)} = \prod_{k=t}^{T_i(s)-1} \frac{\pi(s_k, a_k) T(s_k, a_k, s_{k+1})}{\pi'(s_k, a_k) T(s_k, a_k, s_{k+1})} = \prod_{k=t}^{T_i(s)-1} \frac{\pi(s_k, a_k)}{\pi'(s_k, a_k)} = \frac{\pi(s, a^2) \pi(s^1, a^1)}{\pi'(s, a^2) \pi'(s^1, a^1)}$$

## Off-policy Monte Carlo Control :Evaluate One Policy While Following Another

Compute the value function  $V^\pi(s)$  using the data generated by  $\pi'$

For  $i = 1: n_s$  ( $n_s$  episodes) with  $s_t = s$

$$\left\{ \begin{array}{l} p_i(s_t = s) = \prod_{k=t}^{T_i(s)-1} \pi(s_k, a_k) T(s_k, a_k, s_{k+1}) \\ p'_i(s_t = s) = \prod_{k=t}^{T_i(s)-1} \pi'(s_k, a_k) T(s_k, a_k, s_{k+1}) \end{array} \right.$$

$$\frac{p_i(s_t)}{p'_i(s_t)} = \prod_{k=t}^{T_i(s)-1} \frac{\pi(s_k, a_k)}{\pi'(s_k, a_k)}$$

$$U_i^{\pi'}(s)$$

The desired Monte Carlo estimate after observing  $n_s$  utilities from state  $s_t = s$  is then

$$V^\pi(s) = \frac{\sum_{i=1}^{n_s} \frac{p_i(s)}{p'_i(s)} U_i^{\pi'}(s)}{\sum_{i=1}^{n_s} \frac{p_i(s)}{p'_i(s)}}$$

To compute the weights  $\frac{p_i(s)}{p'_i(s)}$  and estimate the value, we only needs two policies  $\pi(s, a)$  and  $\pi'(s, a)$

## Off-policy Monte Carlo Control :Evaluate One Policy While Following Another

Compute the value function  $Q^\pi(s, a)$  using the data generated by  $\pi'$

For  $i = 1: n_s$  ( $n_s$  episodes) with  $s_t = s$

$$\left\{ \begin{array}{l} p_i(s_t = s, a_t = a) = \prod_{k=t}^{T_i(s)-1} \pi(s_k, a_k) T(s_k, a_k, s_{k+1}) \\ p'_i(s_t = s, a_t = a) = \prod_{k=t}^{T_i(s)-1} \pi'(s_k, a_k) T(s_k, a_k, s_{k+1}) \end{array} \right.$$

$$\frac{p_i(s_t)}{p'_i(s_t)} = \prod_{k=t}^{T_i(s)-1} \frac{\pi(s_k, a_k)}{\pi'(s_k, a_k)}$$

$$U_i^{\pi'}(s, a)$$

The desired Monte Carlo estimate after observing  $n_s$  utilities from state  $s_t = s$  is then

$$Q^\pi(s, a) = \frac{\sum_{i=1}^{n_s} \frac{p_i(s)}{p'_i(s)} U_i^{\pi'}(s, a)}{\sum_{i=1}^{n_s} \frac{p_i(s)}{p'_i(s)}} = \frac{\sum_{i=1}^{n_s} \prod_{k=t}^{T_i(s)-1} \frac{\pi(s_k, a_k)}{\pi'(s_k, a_k)} U_i^{\pi'}(s, a)}{\sum_{i=1}^{n_s} \prod_{k=t}^{T_i(s)-1} \frac{\pi(s_k, a_k)}{\pi'(s_k, a_k)}}$$

To compute the weights  $\frac{p_i(s)}{p'_i(s)}$  and estimate the value, we only needs two policies  $\pi(s, a)$  and  $\pi'(s, a)$

### Key Idea : Off-policy algorithm

- Follows the behavior policy while learning about and improving the estimation policy
- *Behavior* policy  $\pi'(s, a)$ 
  - ✓ The policy used to generate behavior
  - ✓ Requires that the behavior policy have a nonzero probability of selecting all actions that might be selected by the estimation policy (e.g.,  $\epsilon$  – soft policy )
- *Estimation* policy  $\pi(s, a)$ 
  - ✓ The policy that is evaluated and improved
  - ✓  $\pi$  can be deterministic
  - ✓  $\pi$  can be the greedy policy with respect to  $Q$  (an estimation  $Q^\pi$ )

### Disadvantages

→ Learning can be slow

## Off-policy Monte Carlo Control

### Algorithm : An off-policy Monte Carlo Control

Initialize, for all  $s \in \mathcal{S}, a \in \mathcal{A}(s)$

$$Q(s, a) \leftarrow \text{arbitrary}$$

$$N(s, a) \leftarrow 0$$

$$D(s, a) \leftarrow 0$$

$\pi \leftarrow \text{arbitrary deterministic policy}$

Repeat forever:

(a) Select a policy  $\pi'$  and use it to generate an episode:

$$s_0, a_0, r_1, s_1, a_1, r_2, \dots, s_{T-1}, a_{T-1}, r_T, s_T$$

(b) Find  $\tau \leftarrow \text{the latest time at which } a_\tau \neq \pi(s_\tau)$

(c) For each pair  $(s, a)$  appearing in the episodes at time  $\tau$  or later

$t \leftarrow \text{the time of first occurrence of } (s, a) \text{ such that } t \geq \tau$

$$w \leftarrow \prod_{k=t+1}^{T-1} \frac{1}{\pi'(s_k, a_k)}$$

$$Q(s, a) \leftarrow Q(s, a) + w U_t$$

$$D(s, a) \leftarrow D(s, a) + w$$

$$Q(s, a) \leftarrow \frac{N(s, a)}{D(s, a)}$$

(c) For each  $s$  in the episode:

$$\pi(s) \leftarrow \operatorname{argmax}_{a \in \mathcal{A}(s)} Q(s, a)$$

$$Q^\pi(s, a) = \frac{\sum_{i=1}^{n_s} \prod_{k=t}^{T_i(s)-1} \frac{\pi(s_k, a_k)}{\pi'(s_k, a_k)} U_i^{\pi'}(s, a)}{\sum_{i=1}^{n_s} \prod_{k=t}^{T_i(s)-1} \frac{\pi(s_k, a_k)}{\pi'(s_k, a_k)}}$$

## Summary

- The Monte Carlo methods learn value functions and optimal policies from experience in the form of sample episodes
- Advantages are:
  - ✓ Can be used to learn optimal behavior directly from interaction with the environment with no models of environment dynamics
  - ✓ Can be used with simulation or sample models
  - ✓ It is efficient to focus Monte Carlo methods on a small subset of the states
  - ✓ Less harmed by violations of the Markov property. This is because they do not update their value estimates on the basis of the value estimates of successor states (do not use bootstrap)

## **L17. Reinforcement Learning (Temporal Difference Methods)**

1. SARSA
2. Q-learning

# Introduction

## Dynamic Programming (DP) Methods

pros: Update estimates based in part on other learned estimates, without waiting for a final outcome (bootstrap)

cons: Need explicit model

## Monte Carlo (MC) Methods

pros: Learn directly from raw experience without a model

cons: Need to wait until the end of episode to observe expected reward

## Temporal-Difference (TD) Learning

pros: Learn directly from raw experience without a model

MC

+

pros: Update estimates based in part on other learned estimates, without waiting for a final outcome (bootstrap)

DP

Model free

On line  
Incremental

TD Generalized Policy iteration for

TD Policy Evaluation

+

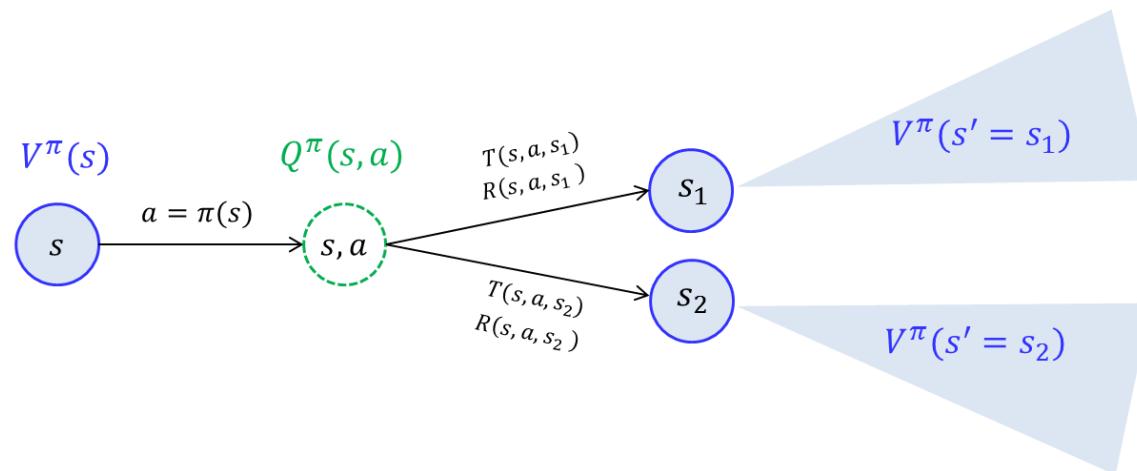
TD Policy Improvement

On-Policy TD Control (SARSA)

Off-Policy Q-learning Control

## Recall : Value function

$$\begin{aligned}
 V^\pi(s) &= \mathbb{E}_\pi(U_t | s_t = s) \\
 &= \mathbb{E}_\pi(r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots | s_t = s) \text{ Complete episode} \\
 &= \mathbb{E}_\pi(\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s) \\
 &= \mathbb{E}_\pi(r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} | s_t = s) \\
 &= \mathbb{E}_\pi(r_{t+1} + \gamma V^\pi(s_{t+1}) | s_t = s) \quad \text{Bootstrapping}
 \end{aligned}$$



## Monte Carlo Policy Evaluation

$$\begin{aligned} V^\pi(s) &= \mathbb{E}_\pi(U_t | s_t = s) \\ &= \mathbb{E}_\pi\left(\sum_{k=0}^T \gamma^k r_{t+k+1} \mid s_t = s\right) \\ &= \mathbb{E}_\pi(r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots \mid s_t = s) \text{ A sampled episode} \end{aligned}$$

Constant- $\alpha$  MC :

After visiting  $s_t$  and receiving utility  $u_t = r_{t+1} + r_{t+2} + r_{t+3} + \dots + r_T$

$$V(s_t) \leftarrow V(s_t) + \alpha[u_t - V(s_t)]$$

$$V(s_t) \leftarrow V(s_t) + \alpha[r_{t+1} + r_{t+2} + r_{t+3} + \dots + r_T - V(s_t)]$$

↳ Target

- The target of update is  $u_t = r_{t+1} + r_{t+2} + r_{t+3} + \dots + r_T$
- A sample reward  $u_t$  from a single episode is used for representing the expected reward. If the episode is long,  $u_t$  will be a lousy estimate (a single initialization)
- This is estimate because we use sampled value instead of expected utility

## Temporal Difference Policy Evaluation

$$\begin{aligned} V^\pi(s) &= \mathbb{E}_\pi(U_t | s_t = s) \\ &= \mathbb{E}_\pi\left(\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s\right) \\ &= \mathbb{E}_\pi\left(r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_t = s\right) \\ &= \mathbb{E}_\pi(r_{t+1} + \gamma V^\pi(s_{t+1}) | s_t = s) \end{aligned} \quad \text{Bootstrapping}$$

### Temporal Difference Policy Evaluation ;TD(0) :

After visiting  $s_t$  and transiting to  $s_{t+1}$  with a single reward  $r_{t+1}$

$$V(s_t) \leftarrow V(s_t) + \alpha [r_{t+1} + \gamma V(s_{t+1}) - V(s_t)]$$

↳ Target

- Bootstrapping: the TD method updates the state value using the previous estimations
- The TD target is an estimate because
  - ✓ it uses the current estimate of  $V(s_t)$ ,
  - ✓ it samples the expected value

$$\mathbb{E}_\pi(r_{t+1} + \gamma V^\pi(s_{t+1}) | s_t = s)$$

## Temporal Difference Policy Evaluation

### Algorithm : Tabular $TD(0)$ for estimating $V^\pi$

Initialize  $V(s)$  arbitrarily,  $\pi$  to the policy to be evaluated

**Repeat** (for each episode):

    Initialize  $s$

**Repeat** (for **each step** of episode)

$a \leftarrow$  action given by  $\pi$  for  $s$

        Take action  $a$ ; observe reward  $r$  and next state  $s'$

$V(s) \leftarrow V(s) + \alpha[r + \gamma V(s') - V(s)]$

$s \leftarrow s'$

    Until  $s$  is terminal

- Simple backups (MC method and TD methods) : Use a single sample success state

Recall:

- Full Backups (DP approach) : Use complete distribution of all possible successors

$$V^\pi(s) = \sum_{s'} T(s, \pi(s), s') \{R(s, \pi(s), s') + \gamma V^\pi(s')\}$$

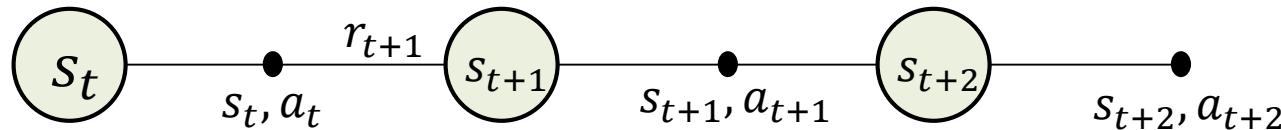
## Advantages of TD Policy Evaluation (prediction)

### What advantages do TD methods have over Monte Carlo and DP methods?

- TD methods learn their estimates on the basis of other estimates (Bootstrap)
- TD methods do not require a model of the environment, i.e., reward and state transition models
- TD methods can be naturally implemented in an **on-line**, fully incremental fashion:
  - ✓ Monte Carlo Method **must wait until the end of an episode**, because only then the return is revealed
  - ✓ TD methods operates with **a single transition of state and action** (a single time step) → advantages for continuous task and learning
- TD methods and Monte Carlo methods converge to  $V^\pi$  in the mean for a constant step-size if it is sufficiently small, and with probability 1 if the step-size parameter decreases
- In practice, TD methods have usually been found to converge faster than  $constnt - \alpha$  MC methods on stochastic tasks

## Temporal Difference Policy Evaluation for Q function

As we estimate state value  $V(s)$ , we can estimate  $Q(s, a)$  using a TD method



### Temporal Difference Policy Evaluation for $Q(s, a)$ function

On each  $(s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1})$  for a single episode:

Note that the action taken is given as data

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - \underbrace{Q(s_t, a_t)}_{\text{Current estimate}}]$$

-----  
Target

### TD Generalized Policy iteration for

TD Policy Evaluation

+

TD Policy Improvement

- On-Policy TD Control (SARSA)
- Off-Policy TD Control (Q-learning)

Estimation and prediction problem

Decision making problems

## Sarsa: On-Policy TD Control

### SARSA Algorithm

Initialize  $Q(s, a)$  arbitrarily

Repeat (for each episode):

    Initialize  $s$

    Choose  $a$  from  $s$  using policy derived from  $Q$  (e.g.,  $\epsilon$  – greedy)

    Repeat (for **each time step** of episode):

        Take action  $a$  given  $s$ , observe  $r, s'$

        Choose  $a'$  from  $s'$  using policy derived from  $Q$  (e.g.,  $\epsilon$  – greedy) **Behavioral policy**

$$Q_\pi(s, a) \leftarrow Q_\pi(s, a) + \eta(r + \gamma Q_\pi(s', a') - Q_\pi(s, a))$$

$s \leftarrow s'; a \leftarrow a'$ ;

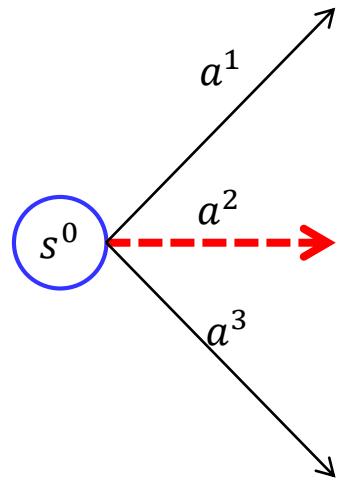
**II**  
**Estimation policy**

Until  $s$  is terminal

- As in all **on-policy methods**, we continually estimate  $Q^\pi$  for the behavioral policy, and the same time change  $\pi$  toward greediness with respect to  $Q^\pi$
- Converges with
  - ✓ All state-action pairs are visited an infinite number of times
  - ✓ The policy converges in the limit to the greedy policy (i.e.,  $\epsilon$  – greedy with  $\epsilon = 1/t$ )

## Sarsa: On-Policy TD Control

$s_t$                    $A_t$                    $R_{t+1}$                    $s_{t+1}$                    $A_{t+1}$                    $s_{t+2}$

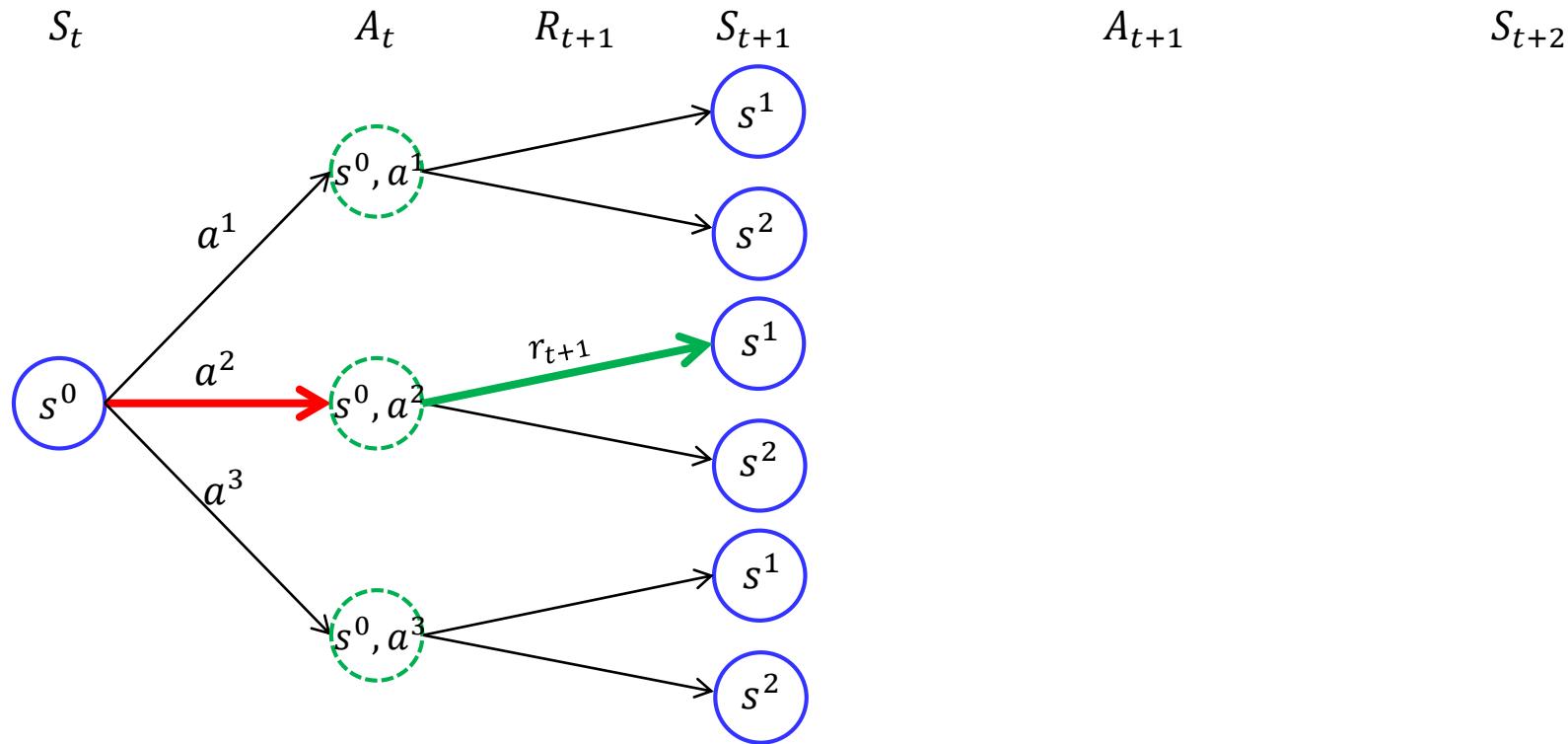


Choose  $a_t$  from  $s_t = s^0$  using current  $Q$

$$a_t = \begin{cases} \underset{a}{\operatorname{argmax}} Q(s_t = s^0, a) & \text{with prob } 1 - \epsilon \\ \text{random action} & \text{with prob } \epsilon \end{cases}$$

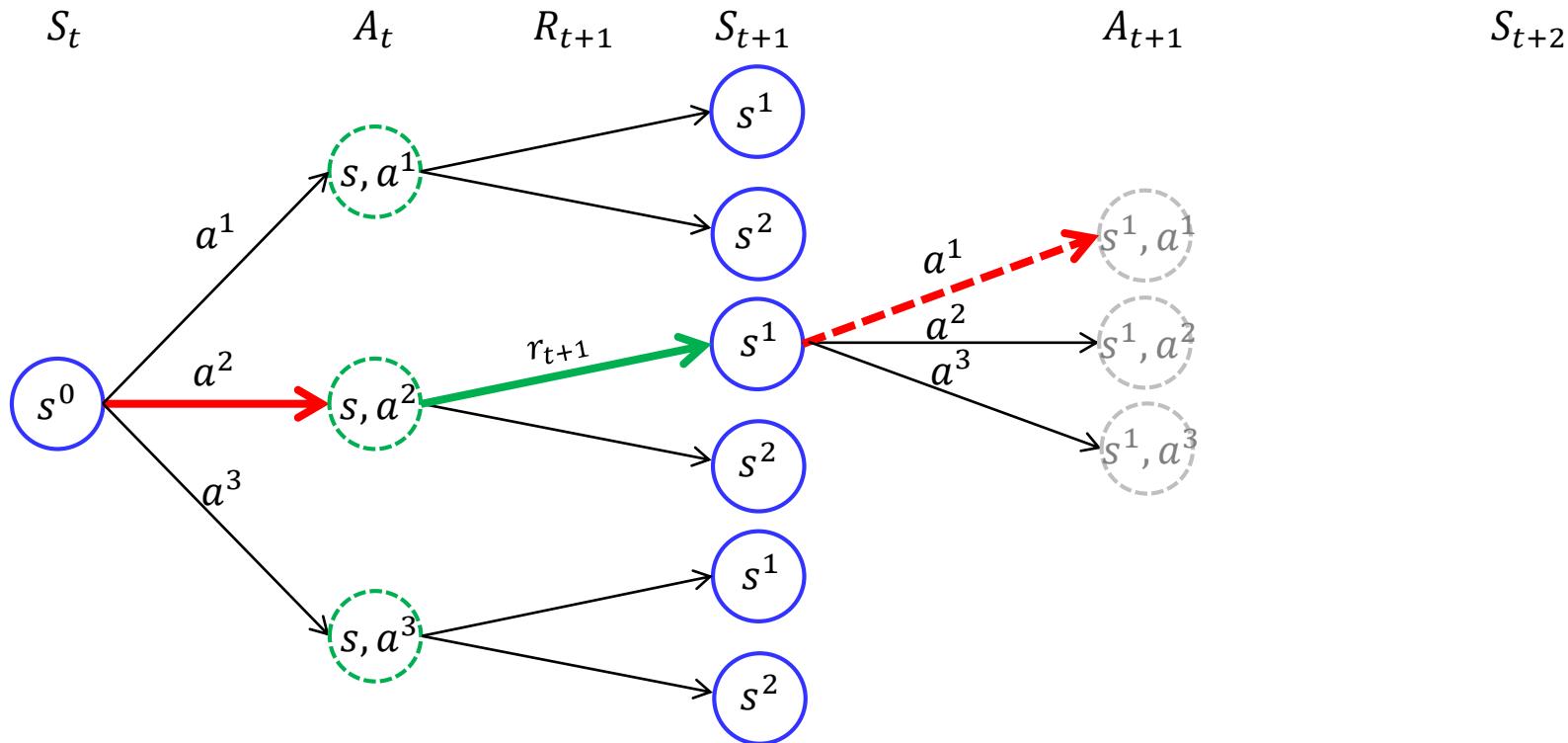
Assume  $a^2$  is chosen

## Sarsa: On-Policy TD Control



Take action  $a_t = a^2$  given  $s_t = s^0$  and observe  $r_{t+1}$  and  $s_{t+1} = s^1$

## Sarsa: On-Policy TD Control

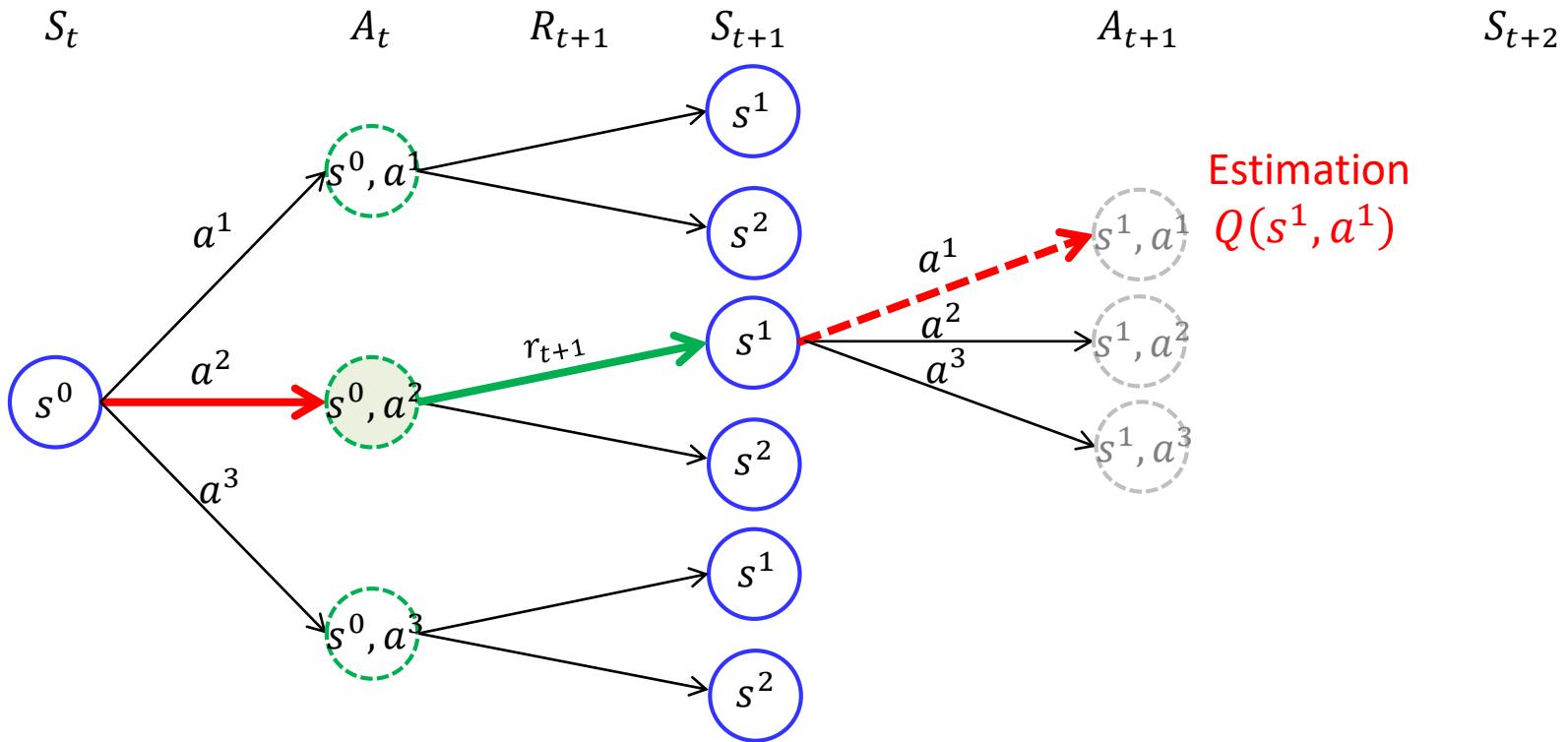


Choose  $a_{t+1}$  from  $s_{t+1} = s^1$  using current  $Q$

$$a_{t+1} = \begin{cases} \underset{a}{\operatorname{argmax}} Q(s_{t+1} = s^1, a) & \text{with prob } 1 - \epsilon \\ \text{random action} & \text{with prob } \epsilon \end{cases}$$

Assume  $a^1$  is chosen

## Sarsa: On-Policy TD Control

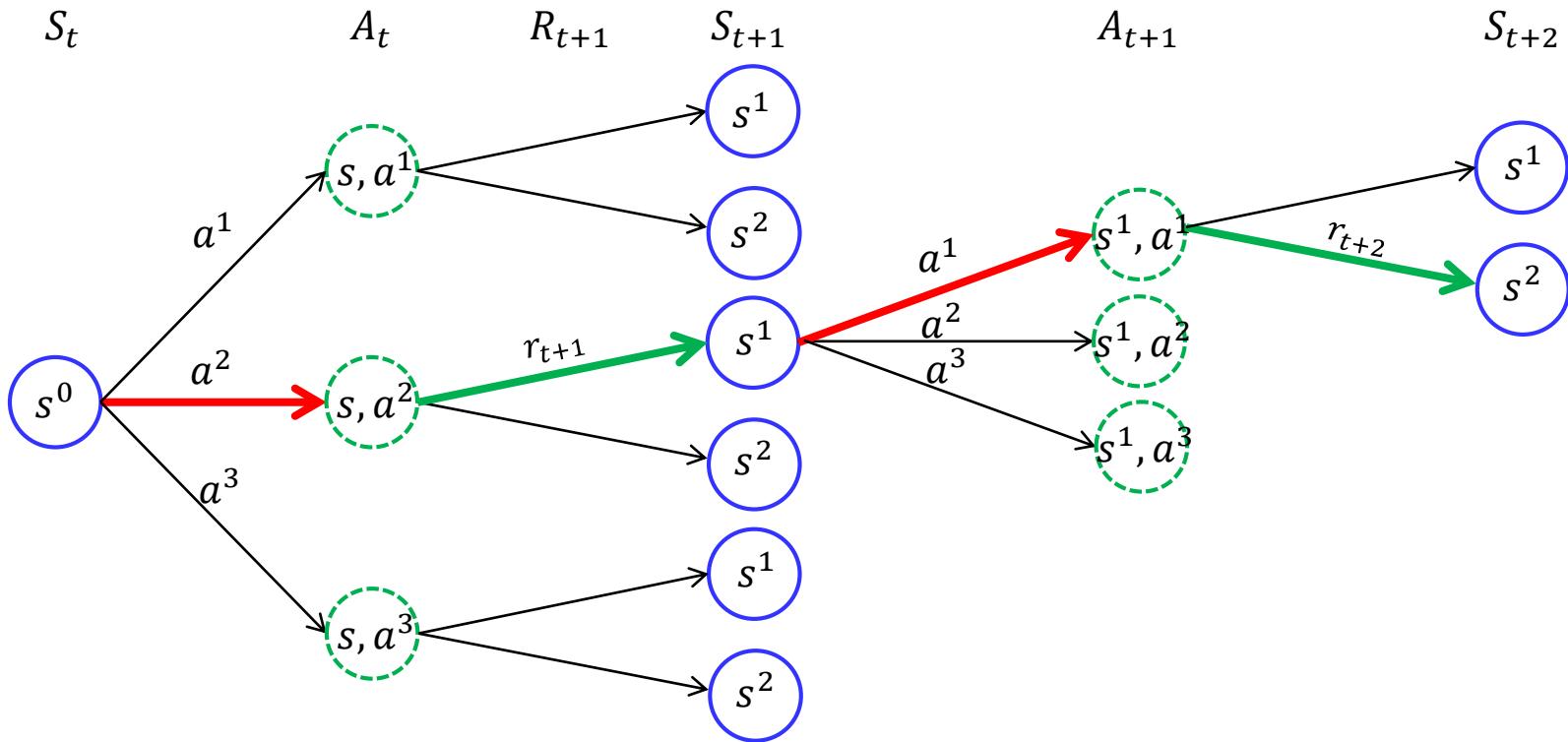


Update  $Q$  function with the estimation  $Q(s_{t+1}, a_{t+1})$

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]$$

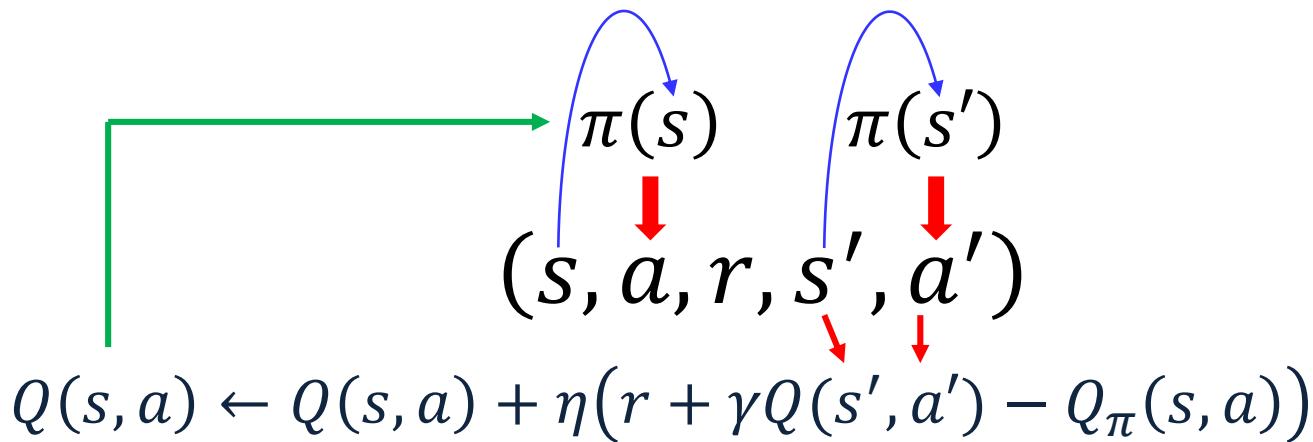
$$\rightarrow Q(s^0, a^2) \leftarrow Q(s^0, a^2) + \alpha[r_{t+1} + \gamma Q(s^1, a^1) - Q(s^0, a^2)]$$

## Sarsa: On-Policy TD Control



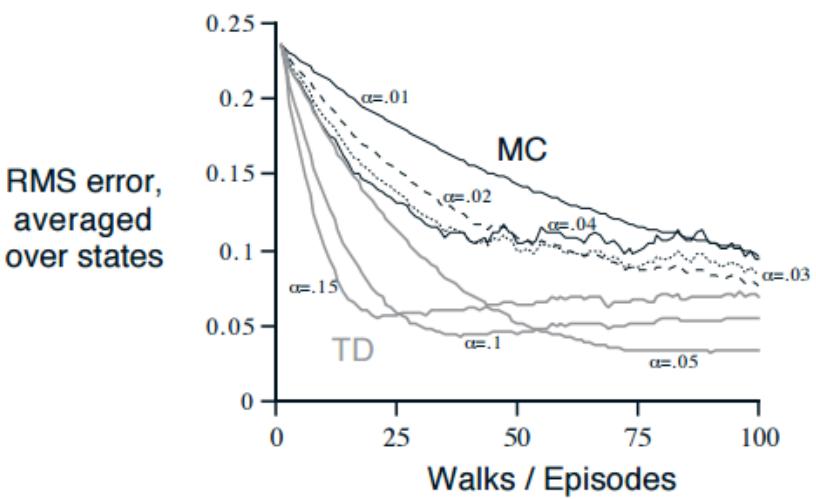
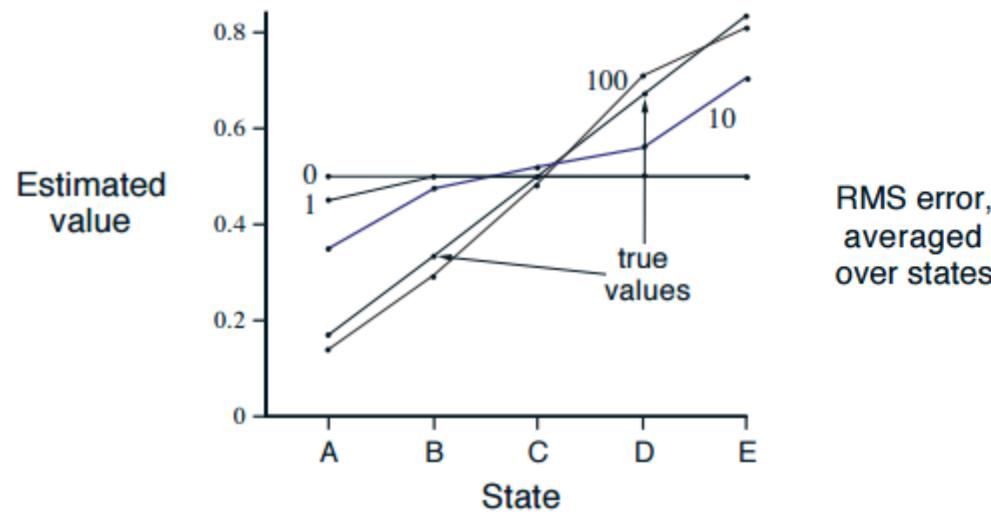
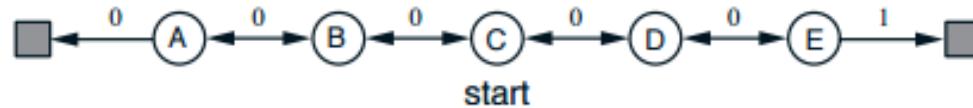
Take action  $a_{t+1} = a^1$  given  $s_{t+1} = s^1$  and observe  $r_{t+2}$  and  $s_{t+2} = s^2$

## Why Q-learning is considered as Off-Policy method



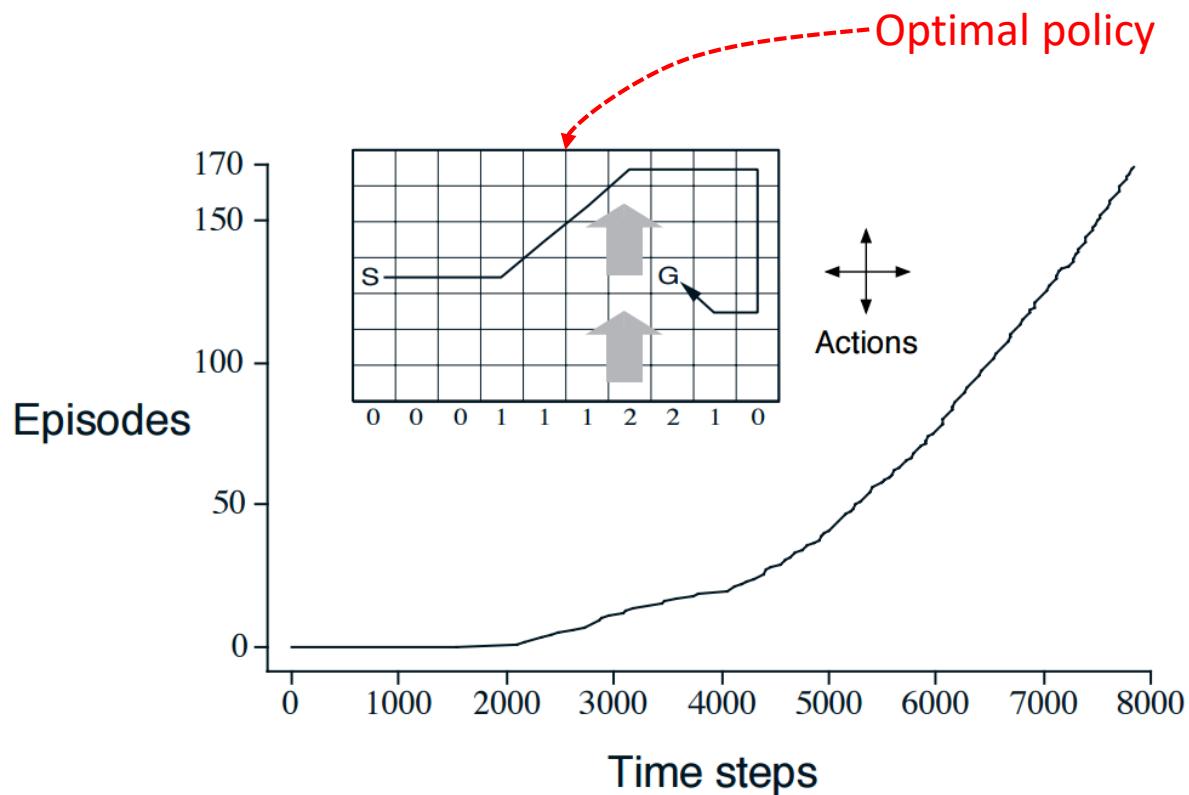
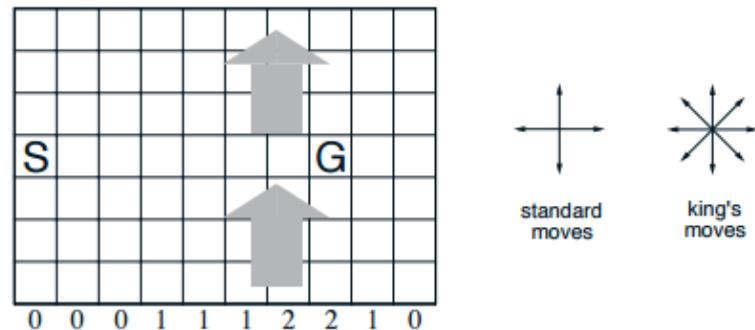
## Sarsa: On-Policy TD Control : Windy Grid world Example

A small Markov process for generating random walk



## Sarsa: On-Policy TD Control : Windy Grid world Example

State transition is encoded by this figure



## Classification of RL

		How to estimate $V^*(s)$ and $Q^*(s, a)$	
		Monte Carlo method	Temporal Difference methods
How to explore ?	Non-Bootstrap	Bootstrap	
	On-policy	On-policy Monte Carlo Control	SARSA
	Off-policy	Off-policy Monte Carlo Control	Q-Learning (SARSmaxA)

• Episodic based      • Single-data-point based

## Q-Learning: Off-Policy TD Control

### On-Policy TD Control (SARSA)

Choose  $a'$  from  $s'$  using policy derived from  $Q$  (e.g.,  $\epsilon$ -greedy)

$$Q(s, a) \leftarrow Q(s, a) + \eta(r + \gamma Q(s', a') - Q(s, a))$$



### Off-Policy TD Control (Q-learning)

$$Q(s, a) \leftarrow Q(s, a) + \eta \left( r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right)$$

- The max over  $a$  rather than taking the  $a$  based on the current policy is the principle difference between Q-learning and SARSA.
- The learned action-value function  $Q$  directly approximates  $Q^*$  independent of the policy being followed
- Converges with
  - ✓ All state-action pairs are visited an infinite number of times
  - ✓ The policy converges in the limit to the greedy policy (i.e.,  $\epsilon$ -greedy with  $\epsilon = 1/t$ )

## Q-Learning: Off-Policy TD Control

### Q learning

Initialize  $Q(s, a)$  arbitrarily

Repeat (for each episode):

    Initialize  $s$

    Repeat (for each time step of episode):

        Choose  $a$  from  $s$  using policy derived from  $Q$  (e.g.,  $\epsilon$  – greedy) **Behavioral policy**

        Take action  $a$ , observe  $r, s'$

$$Q(s, a) \leftarrow Q(s, a) + \eta \left( r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right)$$

$s \leftarrow s'$

    Until  $s$  is terminal

**Estimation policy**

(Always try to estimate the optimal policy)

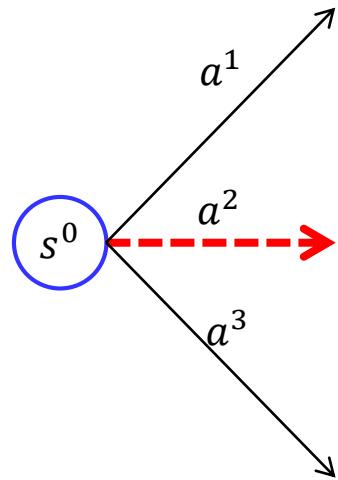
-Estimation can be greedy)

$a^* = \operatorname{argmax}_{a'} Q(s', a)$  is **not** used in the next state!!!

At the next state  $s'$ , Choose  $a$  using policy derived from  $Q$  (e.g.,  $\epsilon$  – greedy)

## Q-Learning: Off-Policy TD Control

$S_t$                    $A_t$                    $R_{t+1}$                    $S_{t+1}$                    $\max A_{t+1}$                    $S_{t+2}$

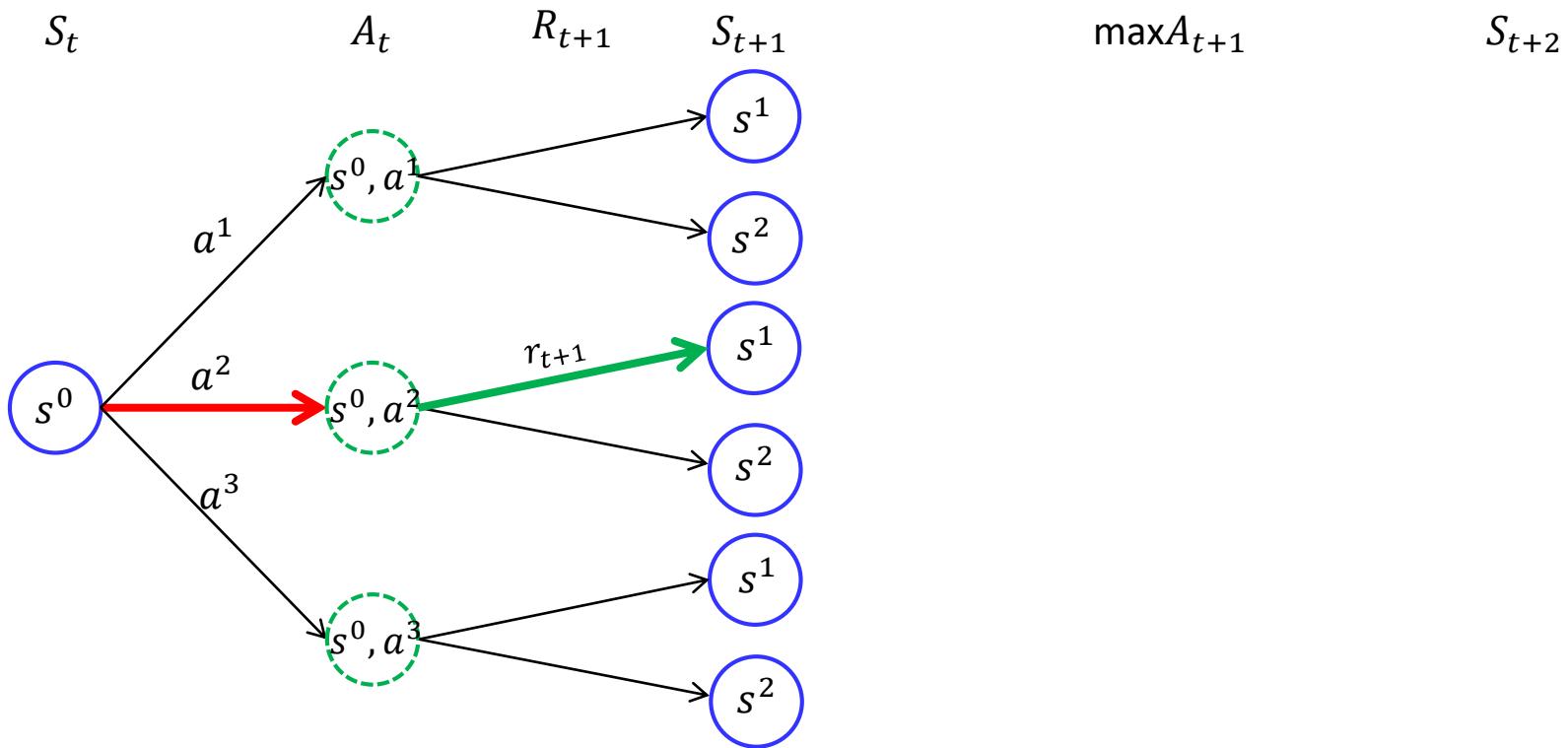


Choose  $a_t$  from  $s_t = s^0$  using current  $Q$

$$a_t = \begin{cases} \underset{a}{\operatorname{argmax}} Q(s_t = s^0, a) & \text{with prob } 1 - \epsilon \\ \text{random action} & \text{with prob } \epsilon \end{cases}$$

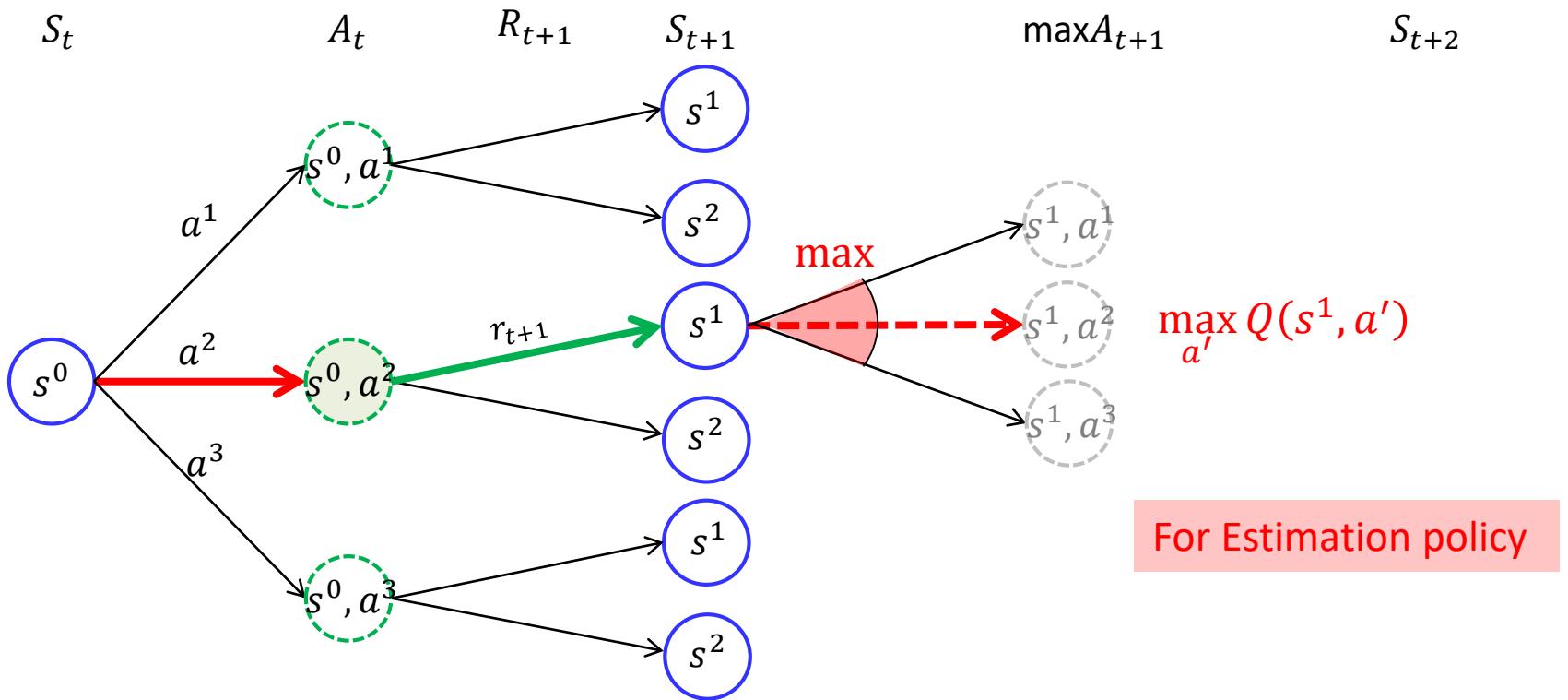
Assume  $a^2$  is chosen

## Q-Learning: Off-Policy TD Control



Take action  $a_t = a^2$  given  $s_t = s^0$  and observe  $r_{t+1}$  and  $s_{t+1} = s^1$

## Q-Learning: Off-Policy TD Control

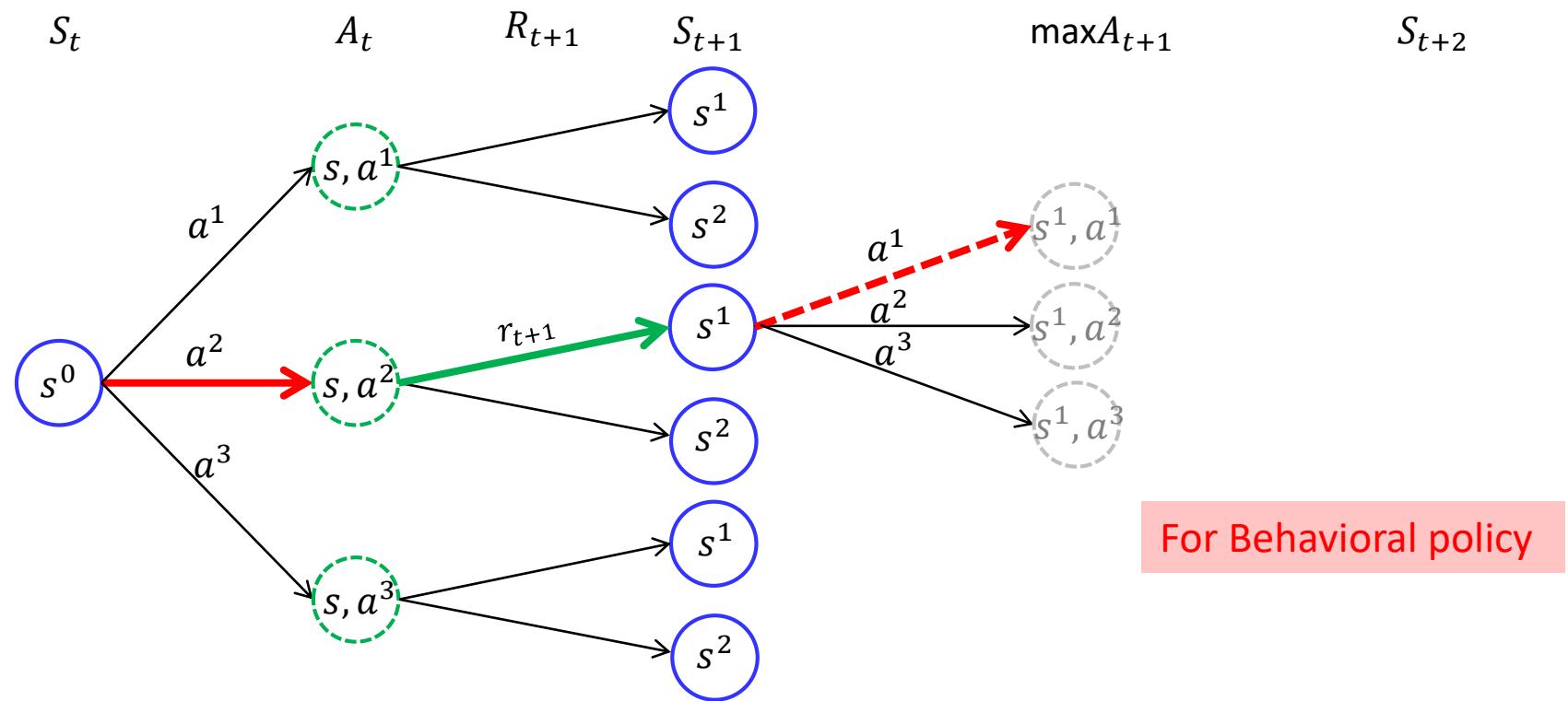


Update  $Q$  function with the  $\max_{a'} Q(s^1, a')$

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[ r_{t+1} + \gamma \max_{a'} Q(s^1, a') - Q(s_t, a_t) \right]$$

$$\rightarrow Q(s^0, a^2) \leftarrow Q(s^0, a^2) + \alpha \left[ r_{t+1} + \gamma \max_{a'} Q(s^1, a') - Q(s^0, a^2) \right]$$

## Q-Learning: Off-Policy TD Control

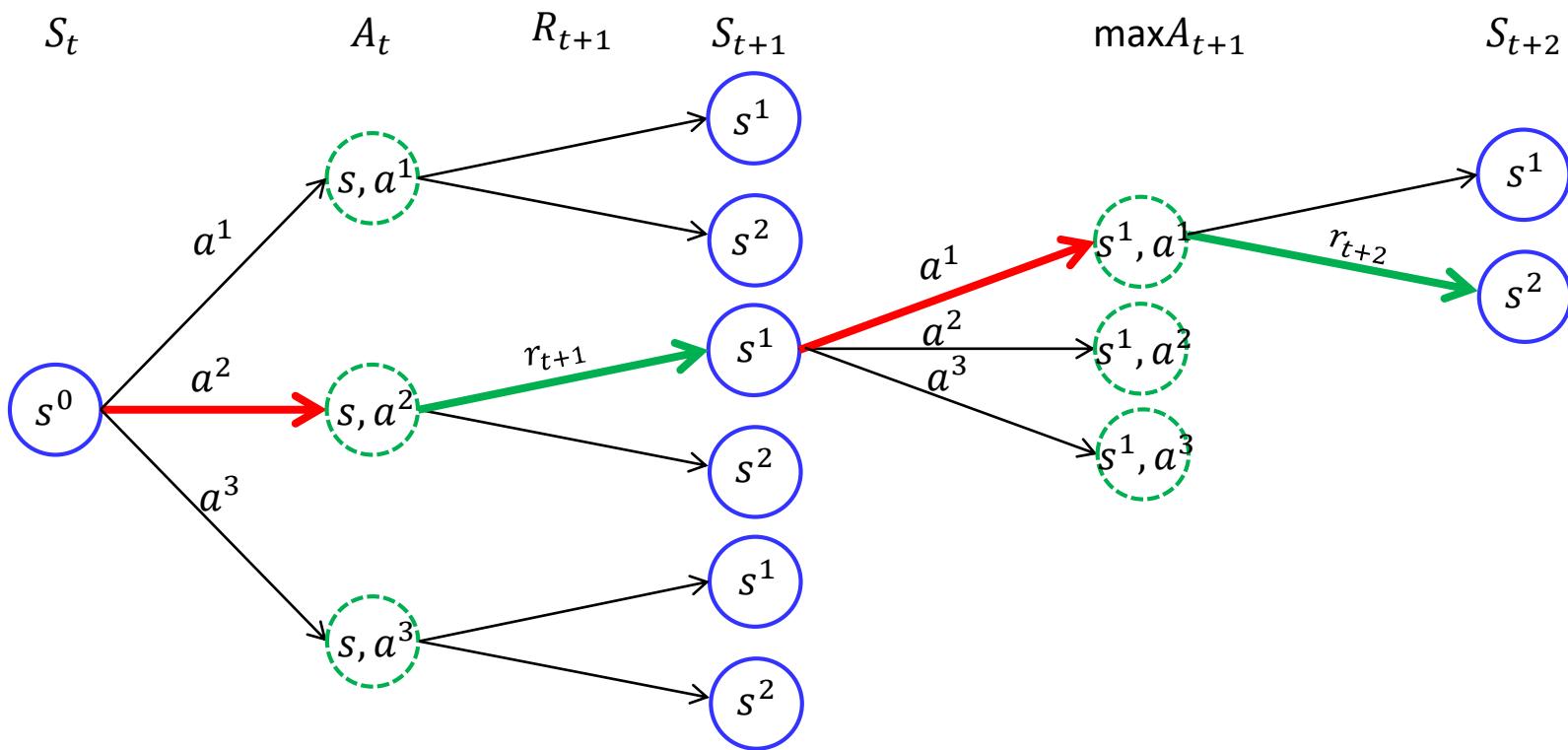


Choose  $a_{t+1}$  from  $s_{t+1} = s^1$  using current  $Q$

$$a_{t+1} = \begin{cases} \underset{a}{\operatorname{argmax}} Q(s_{t+1} = s^1, a) & \text{with prob } 1 - \epsilon \\ \text{random action} & \text{with prob } \epsilon \end{cases}$$

Assume  $a^1$  is chosen

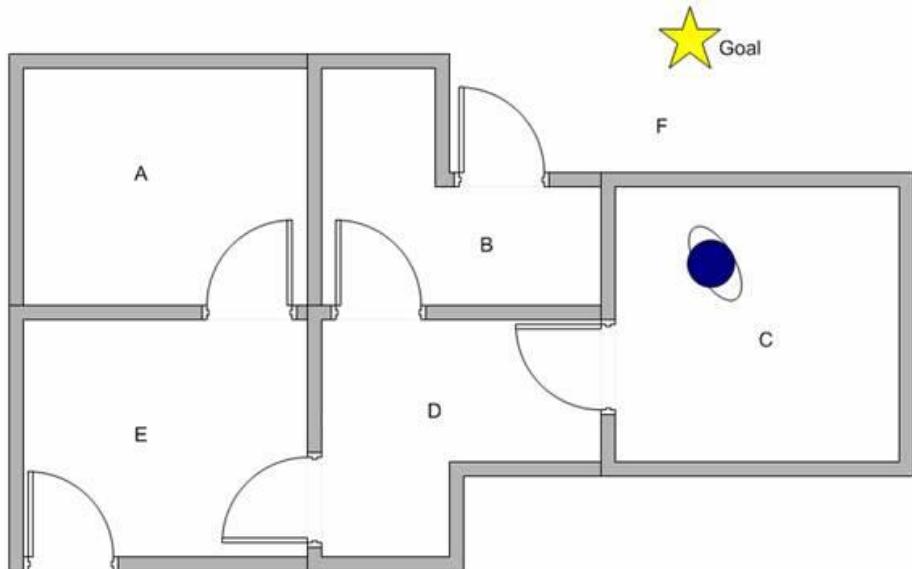
## Q-Learning: Off-Policy TD Control



Take action  $a_{t+1} = a^1$  given  $s_{t+1} = s^1$  and observe  $r_{t+2}$  and  $s_{t+2} = s^2$

## Q-learning Step-by-Step Example

- The agent can pass one room to another but has no knowledge of the building
- That is, it does not know which sequence of doors the agent must pass to go outside the building
- Assume the agent is now in room C, and would like to reach outside the building (state F)



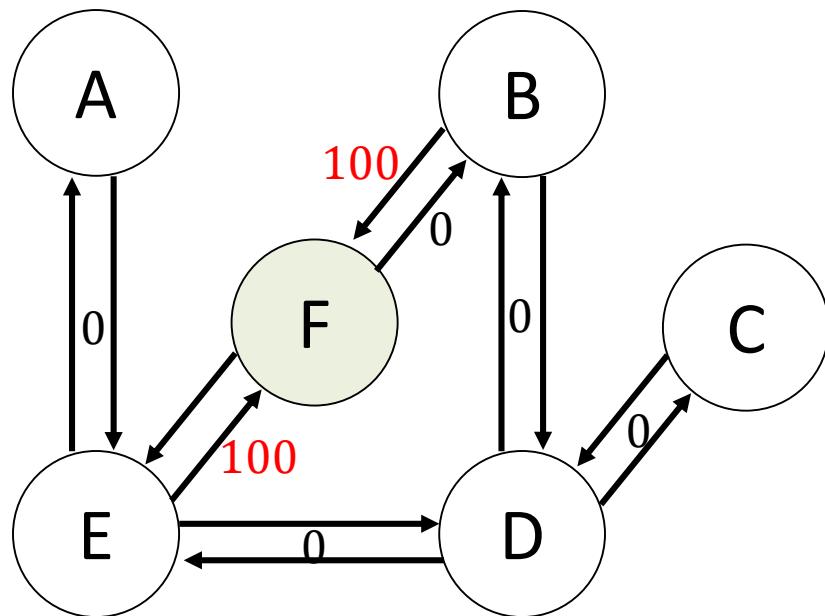
F

- $R(s, a) = \begin{cases} 0 & \text{if move to } a \text{ is allowed and } a \neq F \\ 100 & \text{if move to } a \text{ is allowed and } a = F \end{cases}$
- $\gamma = 0.8$

$$MDP = \{\mathcal{S}, \mathcal{A}, T, R, \gamma\}$$

- $s \in \mathcal{S} = \{A, B, C, D, E, F\}$
- $a \in \mathcal{A} = \{A, B, C, D, E, F\}$   
e.g.,  $\mathcal{A}(s = D) = \{B, C, E\}$
- $T(s, a) = \begin{cases} 1, \text{ if move is allowed} \\ 0, \text{ if move is not allowed} \end{cases}$   
e.g.,  $T(C, D) = 1$

## Q-learning Step-by-Step Example



$$R(s, a) =$$

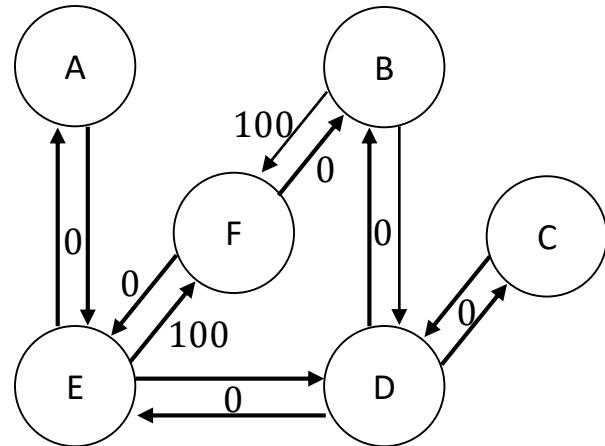
$s \setminus a$	A	B	C	D	E	F
A	-	-	-	-	0	-
B	-	-	-	0	-	100
C	-	-	-	0	-	-
D	-	0	0	-	0	-
E	0	-	-	0	-	100
F	-	0	-	-	0	100

Q Learning update rule:

$$Q(s, a) \leftarrow Q(s, a) + \eta \left( r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right)$$

## Q-learning Step-by-Step Example

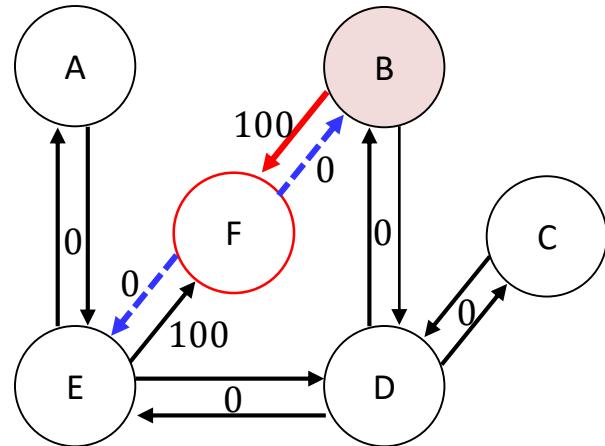
$Q$  Learning update rule:  $Q(s, a) \leftarrow Q(s, a) + \eta \left( r + \gamma \max_a Q(s', a) - Q(s, a) \right)$



$$Q(s, a) = \begin{bmatrix} A & B & C & D & E & F \\ A & 0 & 0 & 0 & 0 & 0 & 0 \\ B & 0 & 0 & 0 & 0 & 0 & 0 \\ C & 0 & 0 & 0 & 0 & 0 & 0 \\ D & 0 & 0 & 0 & 0 & 0 & 0 \\ E & 0 & 0 & 0 & 0 & 0 & 0 \\ F & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

## Q-learning Step-by-Step Example

$Q$  Learning update rule:  $Q(s, a) \leftarrow Q(s, a) + \eta \left( r + \gamma \max_a Q(s', a) - Q(s, a) \right)$



$$Q(s, a) = \begin{bmatrix} A & B & C & D & E & F \\ A & 0 & 0 & 0 & 0 & 0 & 0 \\ B & 0 & 0 & 0 & 0 & 0 & 50 \\ C & 0 & 0 & 0 & 0 & 0 & 0 \\ D & 0 & 0 & 0 & 0 & 0 & 0 \\ E & 0 & 0 & 0 & 0 & 0 & 0 \\ F & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

1. Assume the initial state is  $B$  and take action  $F$  randomly (stochastic policy):

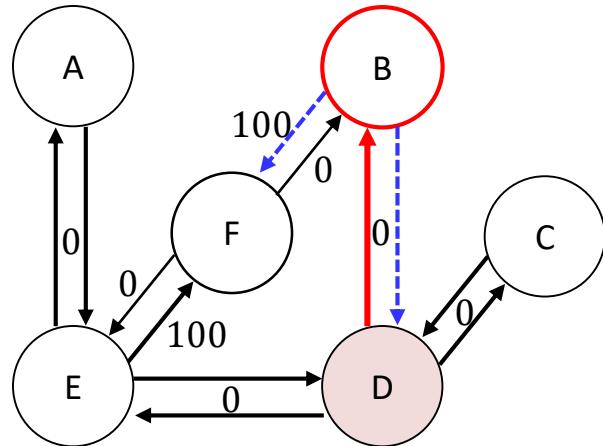
$$Q(B, F) \leftarrow Q(B, F) + 0.5 \left( R(B, F) + 0.8 \max_a \{Q(F, B), Q(F, E)\} - Q(B, F) \right)$$

$$Q(B, F) \leftarrow 0 + 0.5(100 + 0.8 \times 0 - 0) = 50$$

2. Because the state  $F$  is the final state, the episode is over

## Q-learning Step-by-Step Example

$Q$  Learning update rule:  $Q(s, a) \leftarrow Q(s, a) + \eta \left( r + \gamma \max_a Q(s', a) - Q(s, a) \right)$



$$Q(s, a) = \begin{bmatrix} A & B & C & D & E & F \\ A & 0 & 0 & 0 & 0 & 0 & 0 \\ B & 0 & 0 & 0 & 0 & 0 & 50 \\ C & 0 & 0 & 0 & 0 & 0 & 0 \\ D & 0 & 20 & 0 & 0 & 0 & 0 \\ E & 0 & 0 & 0 & 0 & 0 & 0 \\ F & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

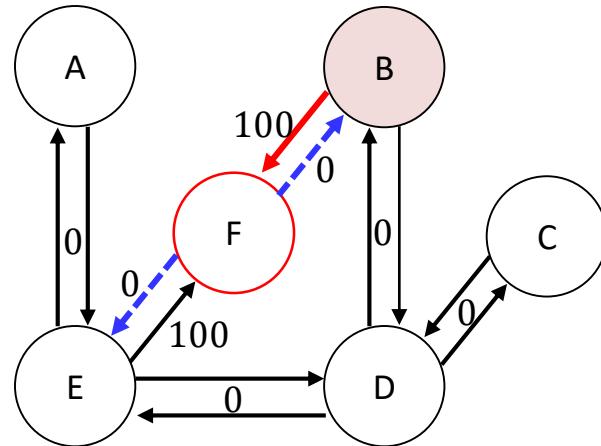
1. Assume the initial state is  $D$  and take action  $B$  randomly (stochastic policy):

$$Q(D, B) \leftarrow Q(D, B) + 0.5 \left( R(D, B) + 0.8 \max_a \{ Q(B, F), Q(B, D) \} - Q(D, B) \right)$$

$$Q(D, B) \leftarrow 0 + 0.5(0 + 0.8 \times 50 - 0) = 20$$

## Q-learning Step-by-Step Example

$Q$  Learning update rule:  $Q(s, a) \leftarrow Q(s, a) + \eta \left( r + \gamma \max_a Q(s', a) - Q(s, a) \right)$



$$Q(s, a) = \begin{bmatrix} A & B & C & D & E & F \\ A & 0 & 0 & 0 & 0 & 0 & 0 \\ B & 0 & 0 & 0 & 0 & 0 & 75 \\ C & 0 & 0 & 0 & 0 & 0 & 0 \\ D & 0 & 20 & 0 & 0 & 0 & 0 \\ E & 0 & 0 & 0 & 0 & 0 & 0 \\ F & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

1. Assume the initial state is  $D$  and take action  $B$  randomly (stochastic policy):

$$Q(D, B) \leftarrow Q(D, B) + 0.5 \left( R(D, B) + 0.8 \max_a \{ Q(B, F), Q(B, D) \} - Q(D, B) \right)$$

$$Q(D, B) \leftarrow 0 + 0.5(0 + 0.8 \times 50 - 0) = 20$$

2. The next state is  $B$  and take an action of  $F$  randomly:

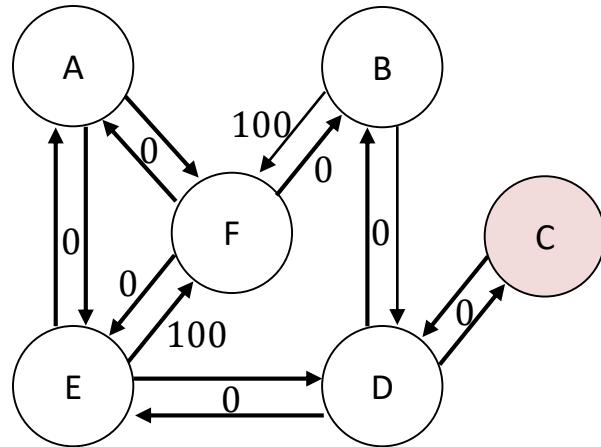
$$Q(B, F) \leftarrow Q(B, F) + 0.5 \left( R(B, F) + 0.8 \max_a \{ Q(F, B), Q(F, E) \} - Q(B, F) \right)$$

$$Q(B, F) \leftarrow 50 + 0.5(100 + 0.8 \times 0 - 50) = 75$$

3. Because the state  $F$  is the final state, the episode is over

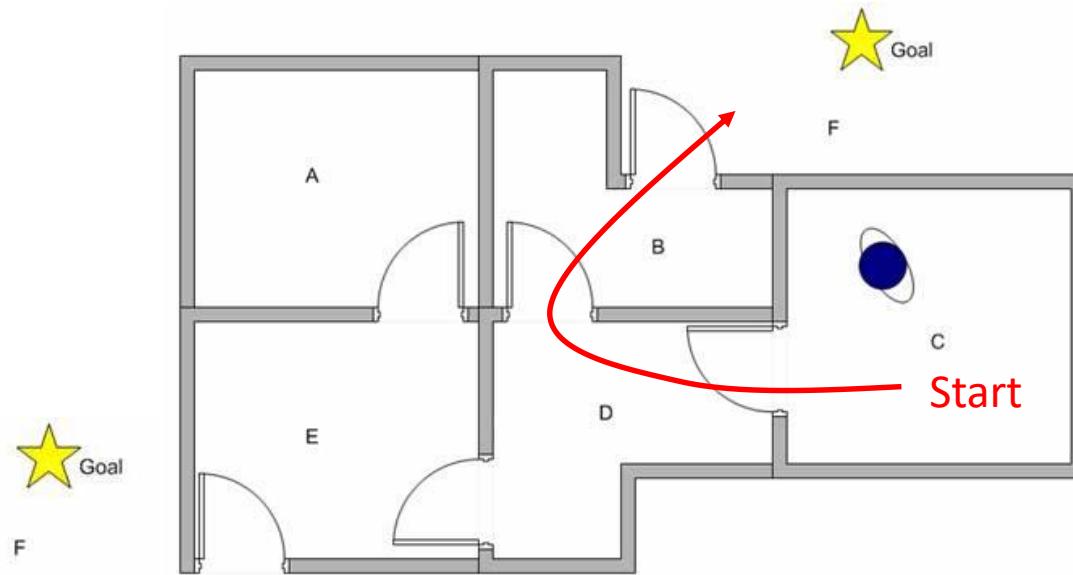
## Q-learning Step-by-Step Example

$Q$  Learning update rule:  $Q(s, a) \leftarrow Q(s, a) + \eta \left( r + \gamma \max_a Q(s', a) - Q(s, a) \right)$

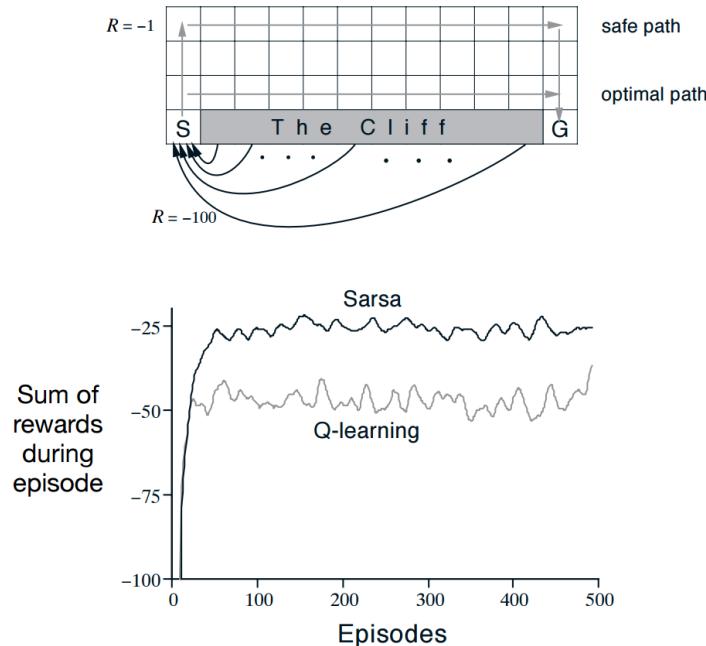


$$Q^*(s, a) = \begin{bmatrix} A & B & C & D & E & F \\ A & 0 & 0 & 0 & 0 & 400 & 0 \\ B & 0 & 0 & 0 & 320 & 0 & 500 \\ C & 0 & 0 & 0 & 320 & 0 & 0 \\ D & 0 & 400 & 256 & 0 & 400 & 0 \\ E & 320 & 0 & 0 & 320 & 0 & 500 \\ F & 0 & 400 & 0 & 0 & 400 & 500 \end{bmatrix}$$

After convergence



## Example 6.6 Cliff Walking



The path away from the cliff

- Take longer
- A wrong action will not hurt you as much

walk near the cliff

- Faster
- a wrong action deterministically causes falling off the cliff.

- **Sarsa** learns about a policy that sometimes takes optimal actions (as estimated) and sometimes explores other actions (Estimation policy = Behavioral policy)
  - Sarsa will learn to be careful in an environment where exploration is costly
- **Q-learning** learns about the policy that doesn't explore and only takes optimal (as estimated) actions
  - The optimal policy does not capture the risk of exploratory action

The cliff example shows why such a non-optimal policy could be sometimes very useful

## Why Q-learning is considered as Off-Policy method

- Q-learning updates are done regardless to the actual action chosen for next state (behavioral policy)
- That is, for estimation, it just assumes that we are always choosing the argmax one

$$a_{t+1} = \underset{a}{\operatorname{argmax}} Q(s_{t+1} = s^1, a)$$

Behavioral Policy  $\pi_B$

$$a'_B = \pi_B(s)$$

Estimation Policy  $\pi_E$

$$a'_E = \underset{a'}{\operatorname{argmax}} Q(s', a')$$

Take action  $a'_B$  and transit to the next state



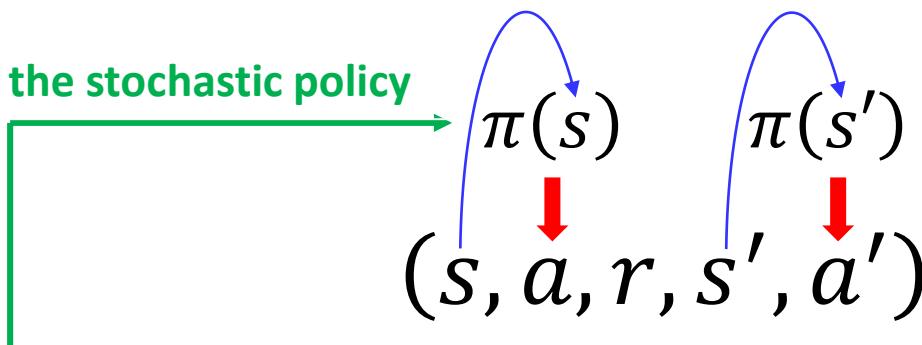
Used to generate data

Used to estimate  $Q(s, a)$

$$Q(s, a) \leftarrow Q(s, a) + \eta \left( r + \gamma \max_{a'} Q(s', a') - Q_\pi(s, a) \right)$$
$$Q(s, a) \leftarrow Q(s, a) + \eta(r + \gamma Q(s', a'_E) - Q_\pi(s, a))$$

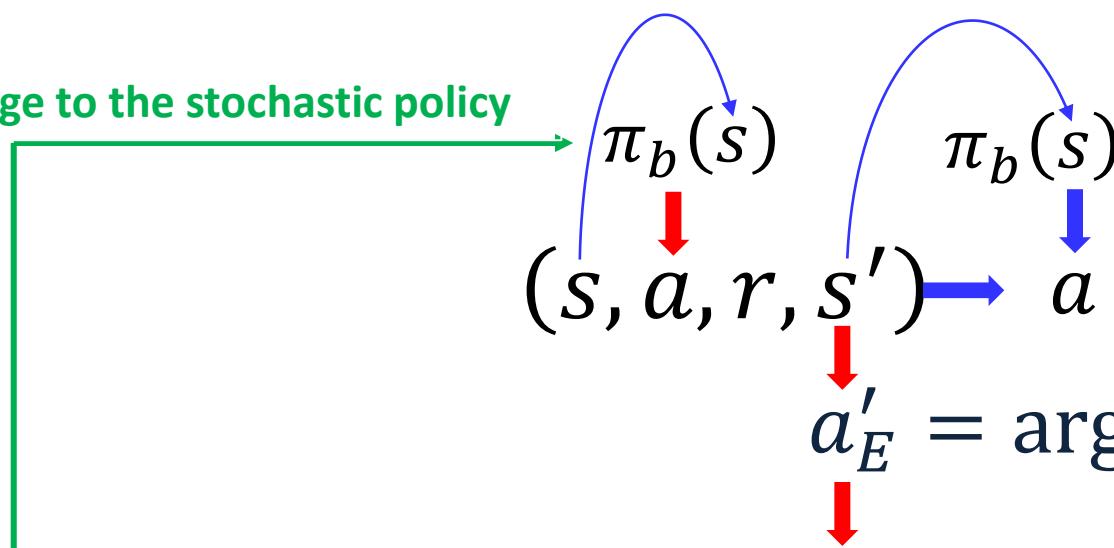
## Why Q-learning is considered as Off-Policy method

Change to the stochastic policy



$$Q(s, a) \leftarrow Q(s, a) + \eta(r + \gamma Q(s', a') - Q_\pi(s, a))$$

Change to the stochastic policy



$$Q(s, a) \leftarrow Q(s, a) + \eta(r + \gamma Q(s', a'_E) - Q_\pi(s, a))$$

Greedy policy (deterministic)

## Why Q-learning is considered as Off-Policy method

### Consider the extreme case:

Suppose you were to take a completely random action on each step (if epsilon greedy exploration is used, set epsilon to 1).

- Sarsa is literally learning the value of the random policy while acting randomly
- Q-learning is learning the value of the optimal policy, but is \*acting\* randomly.

## Summary

### Off-Policy TD Control (Q-learning)

- Based on a single transition, i.e., state-action pair
- Online setting: Learn and take action continuously
- Exploration and Exploitation : Need to learn and optimize at the same time
- Monte Carlo vs. Bootstrapping

**What is the next?**

# **L17. Reinforcement Learning (Extensions)**

## L17. Reinforcement Learning (Extensions)

1. One step → Multiple steps ahead (links to Monte Carlo)
2. Model-free → Include the model of environment (links to Dynamic programming)
3. Tabular → Continuous space (links to supervised learning)
4. Single agent → Multiple Agents (links to Game theory: Stochastic Game)

Dynamic Programming

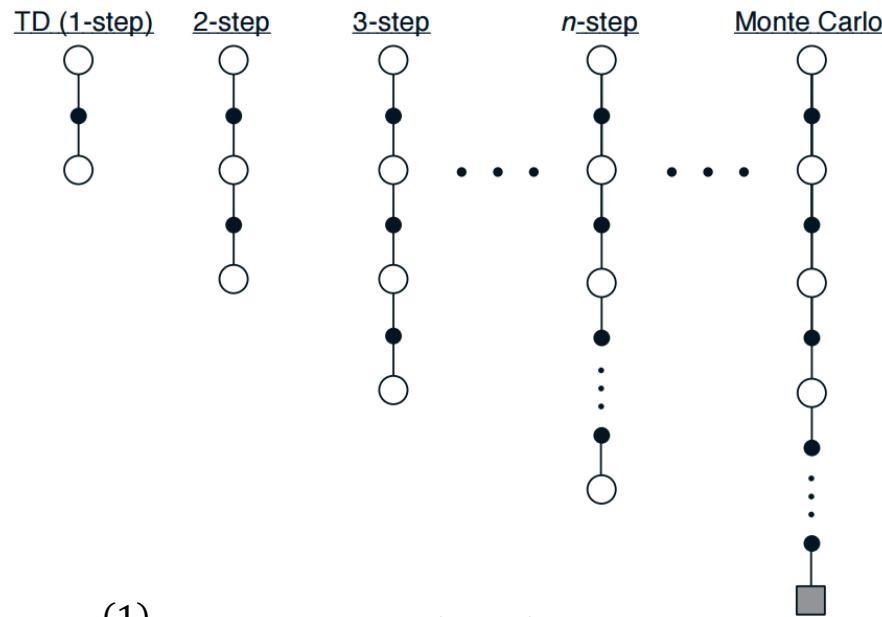
Monte Carlo Control

+

Temporal Difference Control

1. One step → Multiple steps ahead (links to Monte Carlo)
2. Model-free → Include the model of environment (links to Dynamic programming)
3. Tabular → Continuous space (links to supervised learning)
4. Single agent → Multiple Agents (links to Game theory: Stochastic Game)

## n-Steps TD Prediction



TD method

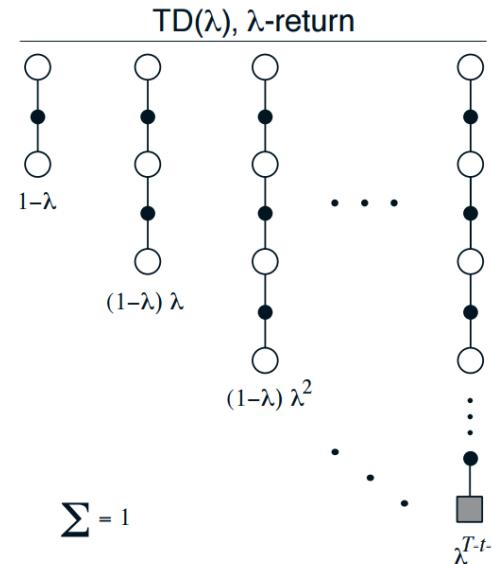


Monte Carlo method

$$\begin{aligned} R_t^{(1)} &= r_{t+1} + \gamma V_t(s_{t+1}) \\ R_t^{(2)} &= r_{t+1} + \gamma r_{t+2} + \gamma^2 V_t(s_{t+2}) \\ R_t^{(3)} &= r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 V_t(s_{t+3}) \\ &\vdots \\ R_t^{(n)} &= r_{t+1} + \gamma V_t(s_{t+1}) + \gamma^2 r_{t+3} + \cdots + \gamma^{n-1} r_{t+n} + \gamma^n V_t(s_{t+n}) \\ &\vdots \\ R_t &= r_{t+1} + \gamma V_t(s_{t+1}) + \gamma^2 r_{t+3} + \cdots + \gamma^{T-t-1} r_T \end{aligned}$$

$$\Delta V_t(s_t) \leftarrow V_t(s_t) + \alpha [R_t^{(n)} - V_t(s_t)]$$

## TD( $\lambda$ ) Method



$$R_t^\lambda = (1 - \lambda) \sum_{n=1}^{\infty} \lambda^{n-1} R_t^{(n)} = (1 - \lambda) \sum_{n=1}^{T-t-1} \lambda^{n-1} R_t^{(n)} + \lambda^{T-t-1} R_t$$

$$\lambda = 0, R_t^{\lambda=0} = R_t^{(1)} = r_{t+1} + \gamma V_t(s_{t+1})$$

TD method

$$\lambda = 1, R_t^{\lambda=1} = R_t$$

Monte Carlo method

$$\Delta V_t(s_t) \leftarrow V_t(s_t) + \alpha [R_t^\lambda - V_t(s_t)]$$

1. One step → Multiple steps ahead (links to Monte Carlo)
2. **Model-free** → Include the model of environment (links to Dynamic programming)
3. Tabular → Continuous space (links to supervised learning)
4. Single agent → Multiple Agents (links to Game theory: Stochastic Game)

## Planning and Learning

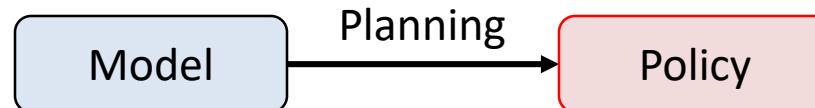
### A Model:

anything that an agent can use to predict how the environment will respond to its actions

- **Distribution models:** produce a description of all possibilities and their probabilities
  - ✓ e.g.,  $T(s, a, s')$
  - ✓ Generate all possible episodes and their probabilities
- **Sample models:** produce just one of the possibility
  - ✓ e.g., Black jack simulator
  - ✓ Generate a an entire episode

### Planning:

Refer to any computational process that takes a model as input and produces or imporves a policy for interacting with the modeled environment

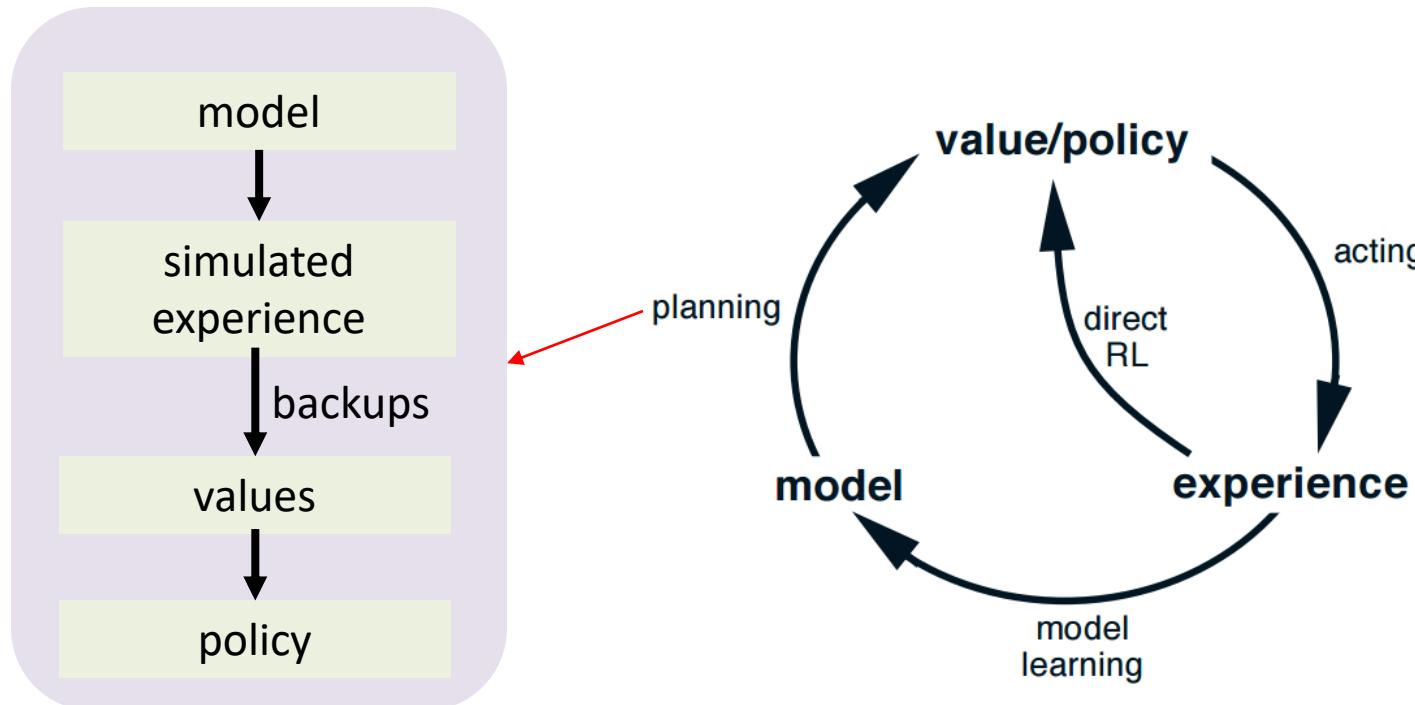


- **State-space planning:** search through the state space for an optimal policy
  - ✓ e.g., Q learning
- **Plan-space planning :** Search through the space of plans
  - ✓ e.g., Route finding, Task ordering

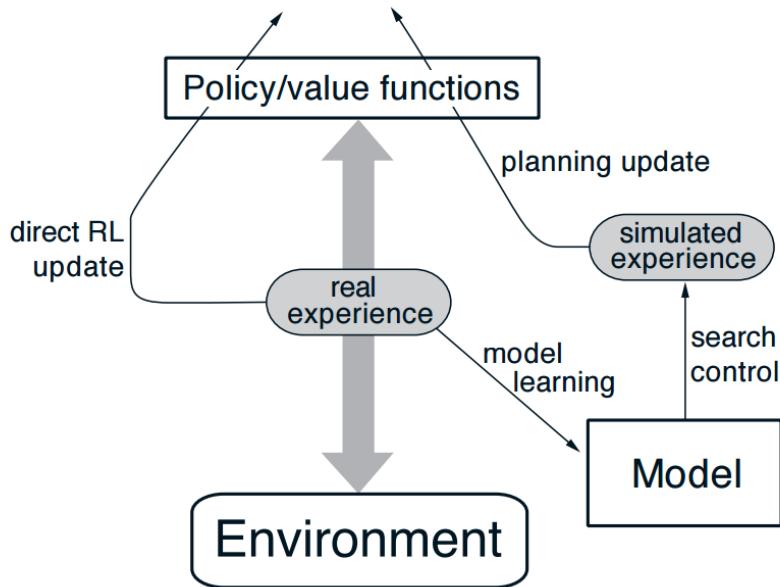
# Integrating Planning, Acting, and Learning

A Planning agent conduct two tasks simultaneously using experience:

- **Model-learning**: use experience to improve the model  
(make it more accurately match the real environment)
- **Direct reinforcement learning** : use experience to directly improve the value function and policy using RL techniques



## Dyna-Q algorithm



Initialize  $Q(s, a)$  and  $Model(s, a)$  for all  $s \in \mathcal{S}$  and  $a \in \mathcal{A}(s)$

Do forever:

- (a)  $S \leftarrow$  current (nonterminal) state
- (b)  $A \leftarrow \epsilon\text{-greedy}(S, Q)$
- (c) Execute action  $A$ ; observe resultant reward,  $R$ , and state,  $S'$
- (d)  $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \max_a Q(S', a) - Q(S, A)]$
- (e)  $Model(S, A) \leftarrow R, S'$  (assuming deterministic environment)
- (f) Repeat  $n$  times:

$S \leftarrow$  random previously observed state

$A \leftarrow$  random action previously taken in  $S$

$R, S' \leftarrow Model(S, A)$

$Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \max_a Q(S', a) - Q(S, A)]$

1. One step → Multiple steps ahead (links to Monte Carlo)
2. Model-free → Include the model of environment (links to Dynamic programming)
- 3. Tabular → Continuous space (links to supervised learning)**
4. Single agent → Multiple Agents (links to Game theory: Stochastic Game)

## Q-learning with function approximation

### Q-learning (stochastic gradient update)

$$Q(s, a) \leftarrow Q(s, a) + \eta \left[ \left( r + \gamma \max_a Q(s', a) \right) - Q(s, a) \right]$$

#### Problem:

- Basic Q-Learning keeps a table of all q-values
- We cannot possibly learn about every single state!
  - Too many states to visit them all in training
  - Too many states to hold the q-tables in memory
- Doesn't generalize to unseen states/actions

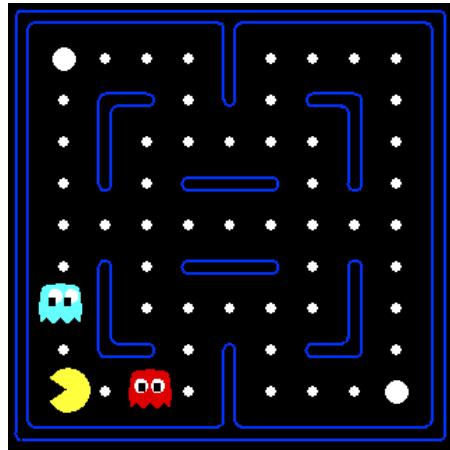
#### Solution:

- Learn about some small number of training states from experience
- Generalize that experience to new, similar situations
  - fundamental idea in machine learning

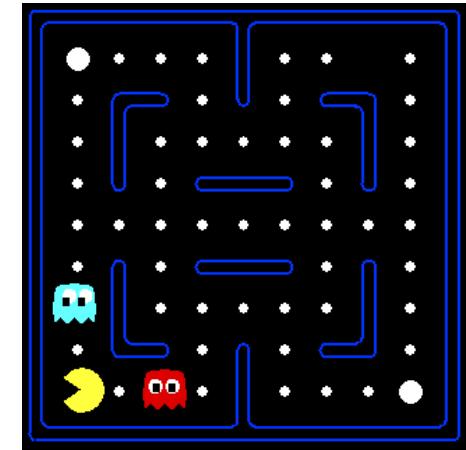
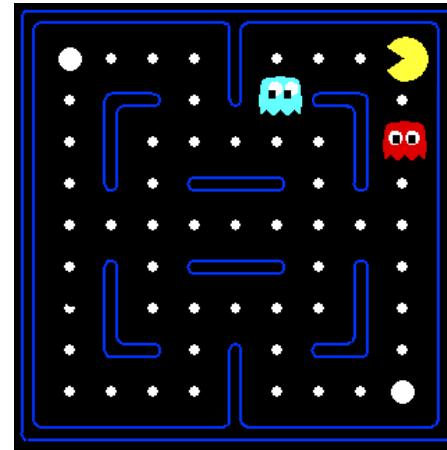
## Q-learning with function approximation

Obtained experience:

This state is bad!



Complete new states



- Are these states good or bad?
- For table representation of  $Q(s, a)$ , we cannot answer
- We can represent  $Q(s, a)$  using the properties of the current state  $s$ :

$$\begin{aligned} Q(s, a; w) &= w_1\phi_1(s, a) + w_2\phi_2(s, a) + \dots + w_n\phi_n(s, a) \\ &= w^T\phi(s, a) \end{aligned}$$

$\phi_i(s, a)$ : distance to closest ghost, number of ghosts, distance to foods

## Q-learning with function approximation

Approximate  $Q(s, a)$  as a function:

$$\begin{aligned} Q(s, a; w) &= w_1 \phi_1(s, a) + w_2 \phi_2(s, a) + \dots + w_n \phi_n(s, a) \\ &= w^T \phi(s, a) \end{aligned}$$

$w$  : weight vector  $\phi(s, a)$  : features vector

- Features,  $\phi(s, a) = \{\phi_1(s, a), \dots, \phi_n(s, a)\}$ , are supposed to be properties of the state-action  $(s, a)$  pair that are indicative of the quality of taking action  $a$  and state  $s$

## Q-learning with function approximation

### Algorithm : Q-learning with function approximation

On each  $(s, a, r, s')$ :

$$w \leftarrow w + \eta \left[ \underbrace{\left( r + \gamma \max_a \hat{Q}(s', a; w) \right)}_{\text{Target}} - \underbrace{\hat{Q}(s, a; w)}_{\text{Estimate}} \right] \phi(s, a)$$

This is equivalent to find the weight  $w$  that maximizes the following objective function

$$\min_{\hat{Q}_\pi} \frac{1}{2} \sum_{(s, a, r, s')} \left( \underbrace{\left( r + \gamma \max_a \hat{Q}(s', a; w) \right)}_{\text{Target}} - \underbrace{\hat{Q}(s, a; w)}_{\text{Estimate}} \right)^2$$

For a single transition  $(s, a, r, s')$ , the error can be expressed as:

$$\begin{aligned} \text{Error}(w) &= \frac{1}{2} \left( \left( r + \gamma \max_a \hat{Q}(s', a; w) \right) - \hat{Q}(s, a; w) \right)^2 \\ &= \frac{1}{2} \left( \left( r + \gamma \max_a \hat{Q}(s', a; w) \right) - \sum_{k=1}^n w_k \phi_k(s, a) \right)^2 \quad \because \hat{Q}(s, a; w) = \sum_{k=1}^n w_k \phi_k(s, a) \end{aligned}$$

$$\frac{\partial \text{Error}(w)}{\partial w_k} = - \left( \left( r + \gamma \max_a \hat{Q}(s', a; w) \right) - \sum_{k=1}^n w_k \phi_k(s, a) \right) \phi_k(s, a)$$

$$w_k \leftarrow w_k + \eta \left[ \left( r + \gamma \max_a \hat{Q}(s', a; w) \right) - \hat{Q}(s, a; w) \right] \phi(s, a)$$

## Deep Reinforcement Learning

Use a neural network for estimating  $Q(s, a)$

Playing Atari [Google DeepMind, 2013]:

$$a^* = \pi(s =$$

move left  
move right  
stroke



Figure 1: Atari Breakout game. Image credit: DeepMind.

- last 4 frames (images)  $\Rightarrow$  3-layer NN  $\Rightarrow$  keystroke (move left, right, stroke)
- $\epsilon$ -greedy, train over 10M frames with 1M replay memory
- Human-level performance on some games (breakout)

## Q-Learning with Neural Network

Screen size :  $84 \times 84$  and convert to grayscale with 256 gray levels

State size =  $256^{84 \times 84 \times 4} \approx 10^{67970}$  more than the number of atoms in the known universe

$$Q(s, a) = \text{[Red Box]} \quad \left\{ \begin{array}{l} \\ \\ \\ \\ \end{array} \right\} 10^{67970}$$

- Need to represent this large Q table using a function approximation (Deep learning)
- Need to estimate Q-values for states that have never been seen before (generalization)

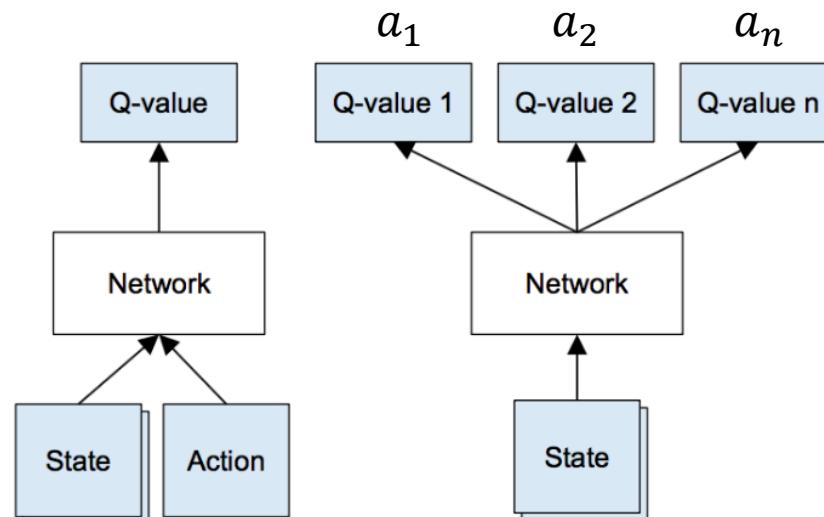


Figure 3: Left: Naive formulation of deep Q-network. Right: More optimized architecture of deep Q-network, used in DeepMind paper.

## Q-Learning with Neural Network

$$L = \frac{1}{2} [\underbrace{r + \max_{a'} Q(s', a')}_{\text{target}} - \underbrace{Q(s, a)}_{\text{prediction}}]^2$$

Given a transition  $\langle s, a, r, s' \rangle$ , the Q-table update rule in the previous algorithm must be replaced with the following:

1. Do a feedforward pass for the current state  $s$  to get predicted Q-values for all actions.
2. Do a feedforward pass for the next state  $s'$  and calculate maximum overall network outputs  $\max_{a'} Q(s', a')$ .
3. Set Q-value target for action to  $r + \gamma \max_{a'} Q(s', a')$  (use the max calculated in step 2). For all other actions, set the Q-value target to the same as originally returned from step 1, making the error 0 for those outputs.
4. Update the weights using backpropagation.

# Deep Q-learning Algorithm

This gives us the final deep Q-learning algorithm with experience replay:

```
initialize replay memory D
initialize action-value function  $Q$  with random weights
observe initial state  $s$ 
repeat
    select an action  $a$ 
        with probability  $\epsilon$  select a random action
        otherwise select  $a = \operatorname{argmax}_{a'} Q(s, a')$ 
    carry out action  $a$ 
    observe reward  $r$  and new state  $s'$ 
    store experience  $\langle s, a, r, s' \rangle$  in replay memory  $D$ 

    sample random transitions  $\langle ss, aa, rr, ss' \rangle$  from replay memory  $D$ 
    calculate target for each minibatch transition
        if  $ss'$  is terminal state then  $tt = rr$ 
        otherwise  $tt = rr + \gamma \max_{a'} Q(ss', aa')$ 
    train the  $Q$  network using  $(tt - Q(ss, aa))^2$  as loss

     $s = s'$ 
until terminated
```

1. One step → Multiple steps ahead (links to Monte Carlo)
2. Model-free → Include the model of environment (links to Dynamic programming)
3. Tabular → Continuous space (links to supervised learning)
- 4. Single agent → Multiple Agents (links to Game theory: Stochastic Game)**

## Normal Form Game

### Prisoner's Dilemma

		Player 2	
		C	D
		(-1, -1)	(-4, 0)
Player 1	C	(-1, -1)	(-4, 0)
	D	(0, -4)	(-3, 3)

- Nash Equilibrium Solution Concept

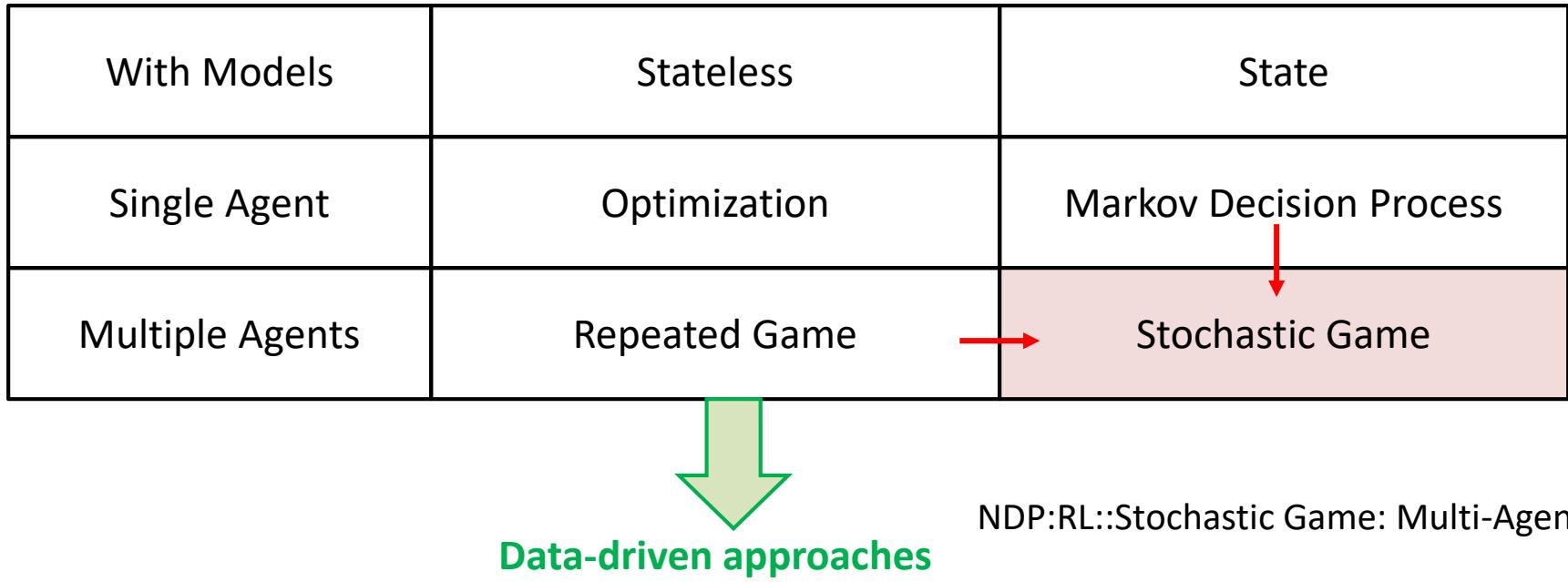
## Repeated Game

What happens when a simple normal-form game is repeated infinitely?

<b>C</b>	<b>D</b>	<b>C</b>	<b>D</b>	<b>C</b>	<b>D</b>	...	<b>C</b>	<b>D</b>
C	(-1, -1)    (-4, 0)	C	(-1, -1)    (-4, 0)	C	(-1, -1)    (-4, 0)		C	(-1, -1)    (-4, 0)
D	(0, -4)    (-3, 3)	D	(0, -4)    (-3, 3)	D	(0, -4)    (-3, 3)		D	(0, -4)    (-3, 3)

- Assume each player does not know the action chosen by other player in the current game, but becomes to know after the single state game is over
- What kind of decision making strategies an agent need to use to maximize the payoff
  - ✓ **Tit-for-Tat:** a strategy in which the players starts by cooperating and thereafter chooses in round  $j + 1$  the action chosen by the other player in round  $j$
  - ✓ **Trigger strategy:** Tit-for-Tat → defect forever once the opponent defects

## Stochastic Games



Without Models	Stateless	State
Single Agent	Model-free Optimization	Reinforcement Learning
Multiple Agents	Learning in Repeated Game	Multi-Agents Reinforcement Learning

## Stochastic Games

**A stochastic game is a generalization of repeated games**

- agents repeatedly play games from a set of normal-form games
- the game played at any iteration depends on the previous game played and on the actions taken by all agents in that game

**A stochastic game is a generalized Markov decision process**

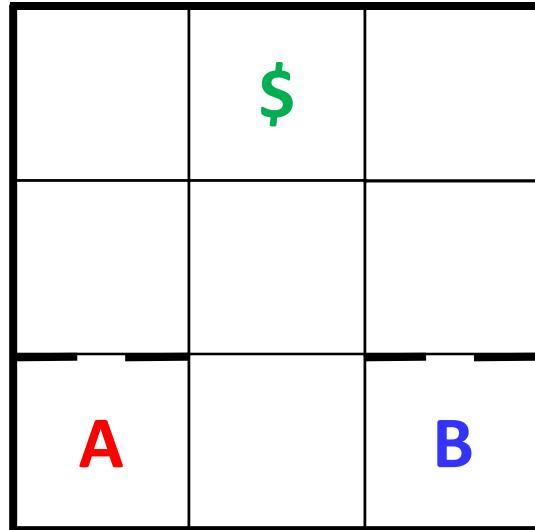
- there are multiple players
- one reward function for each agent
- the state transition function and reward functions depend on the action choices of all of the players

## Stochastic Games

A **stochastic game** is a **tuple**  $(S, N, A, T, R)$ , where

- $S$  is a finite set of states
- $N$  is a finite set of  $n$  players
- $A = A_1 \times \dots \times A_n$ , where  $A_i$  is a finite set of actions available to player  $i$
- $T: S \times A \times S \mapsto [0,1]$  is the transition probability function;
  - ✓  $T(s, a, s')$  is the probability of transitioning from state  $s$  to state  $s'$  after joint action  $a$
- $R = r_1, \dots, r_n$ , where  $r_i: S \times A \mapsto \mathbb{R}$  is a real-valued payoff function for player  $i$ 
  - ✓  $R_i(s, a)$  is the reward function rewarded by taking a joint action  $a$  given state  $s$
- This assumes strategy space is the same in all games otherwise just more notation
- Again we can have average or discounted payoffs.
- Interesting special cases:
  - zero-sum stochastic game
  - single-controller stochastic game transitions (but not payoffs) depend on only one agent

## Stochastic (Markov) Games



- First to reach goal gets \$100
- If both reaches the money at the same time, both win
- Semi wall (50% go through)
- Cannot occupy the same grid
- Coin flip if collide

### 2 players stochastic game

- $S$  (States) :  $s \in S$
- $A_i$  (Actions for player  $i$ ) :  $a_1 \in A_1, a_2 \in A_2$
- $T$  (Transitions) :  $T(s, (a_1, a_2), s')$
- $R_i$  (Rewards for player  $i$ ) :  $R_1(s, (a_1, a_2)), R_2(s, (a_1, a_2))$
- $\gamma$  (Discount factor)

### Narrowing down the definition

- $R_1 = -R_2$ :
- $T(s, (a_1, a_2), s') = T(s, (a_1, a'_2), s')$  for any  $a'_2$
- $R(s, (a_1, a_2)) = T(s, (a_1, a'_2))$  for any  $a'_2$
- $|S| = 1$
- Zero-sum Stochastic Game
- MDP
- Repeated Game

## Stochastic (Markov) Games

### 2 players stochastic game

- Zero-sum Stochastic Game

MDP  $\leftarrow$  Q-learning

Can we employ Q-learning approach to Stochastic Game?

$$Q^*(s, a) = \sum_{s'} T(s, a, s') \left\{ R(s, a, s') + \gamma \max_{a'} Q^*(s', a') \right\}$$

$$Q_i^*(s, (a_1, a_2)) = \sum_{s'} T(s, (a_1, a_2), s') \left\{ R_i(s, (a_1, a_2), s') + \gamma \max_{(a'_1, a'_2)} Q_i^*(s', (a'_1, a'_2)) \right\}$$

- Is this correct?

- This is not correct

because it assumes that joint actions will benefit the agent  $i$  the most.

## Stochastic (Markov) Games

### 2 players stochastic game

- Zero-sum Stochastic Game

MDP  $\leftarrow$  Q-learning

Can we employ Q-learning approach to Stochastic Game?

$$Q^*(s, a) = \sum_{s'} T(s, a, s') \left\{ R(s, a, s') + \gamma \max_{a'} Q^*(s', a') \right\}$$

$$Q_i^*(s, (a_1, a_2)) = \sum_{s'} T(s, (a_1, a_2), s') \left\{ R_i(s, (a_1, a_2), s') + \gamma \max_{(a'_1, a'_2)} Q_i^*(s', (a'_1, a'_2)) \right\}$$

$$Q_i^*(s, (a_1, a_2)) = \sum_{s'} T(s, (a_1, a_2), s') \left\{ R_i(s, (a_1, a_2), s') + \gamma \min_{(a'_1, a'_2)} \max Q_i^*(s', (a'_1, a'_2)) \right\}$$

## Stochastic (Markov) Games

### 2 players stochastic game

- Zero-sum Stochastic Game

$$Q_i^*(s, (a_1, a_2)) = \sum_{s'} T(s, (a_1, a_2), s') \left\{ R_i(s, (a_1, a_2), s') + \gamma \minmax_{(a'_1, a'_2)} Q_i^*(s', (a'_1, a'_2)) \right\}$$

$$Q_i^*(s, (a_1, a_2)) \leftarrow r_i + \gamma \minmax_{(a'_1, a'_2)} Q_i^*(s', (a'_1, a'_2))$$

e.g., for player 1

$$Q_1^*(s, (a_1, a_2)) \leftarrow r_1 + \gamma \min_{a'_2} \max_{a'_1} Q_2^*(s', (a'_1, a'_2))$$

#### Comments:

- Value iteration works
- $\minmax_{(a'_1, a'_2)} Q_i^*(s', (a'_1, a'_2))$  converges
- Unique solution to  $Q_i^*$
- Policy can be computed independently
- $Q_i^*$  is sufficient to optimally behave

### 2 players stochastic game

- General-sum Stochastic Game

$$Q_i^*(s, (a_1, a_2)) = \sum_{s'} T(s, (a_1, a_2), s') \left\{ R_i(s, (a_1, a_2), s') + \gamma \minmax_{(a'_1, a'_2)} Q_i^*(s', (a'_1, a'_2)) \right\}$$

$$\downarrow$$

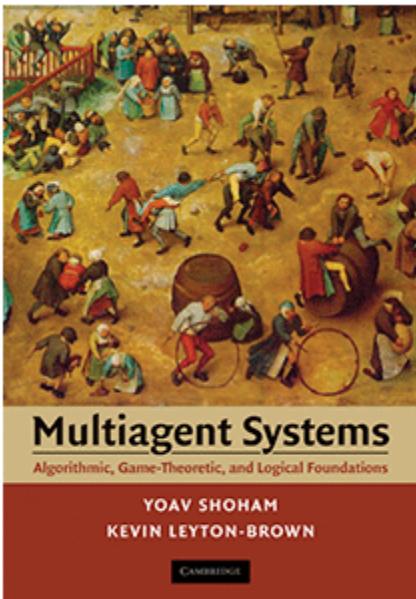
$$Q_i^*(s, (a_1, a_2)) = \sum_{s'} T(s, (a_1, a_2), s') \left\{ R_i(s, (a_1, a_2), s') + \gamma \underset{(a'_1, a'_2)}{\text{Nash}} Q_i^*(s', (a'_1, a'_2)) \right\}$$

$$Q_i^*(s, (a_1, a_2)) \leftarrow r_i + \gamma \underset{(a'_1, a'_2)}{\text{Nash}} Q_i^*(s', (a'_1, a'_2))$$

Comments:

- Value iteration **doesn't** works
- $\minmax_{(a'_1, a'_2)} Q_i^*(s', (a'_1, a'_2))$  **doesn't** converges
- **No** Unique solution to  $Q_i^*$
- Policy can **not** be computed independently
- $Q_i^*$  is **not** sufficient to optimally behave

2017 Spring



### Multiagent Systems Algorithmic, Game-Theoretic, and Logical Foundations

Yoav Shoham  
Stanford University  
Kevin Leyton-Brown  
University of British Columbia

Cambridge University Press, 2009

Order online: [amazon.com](https://www.amazon.com).

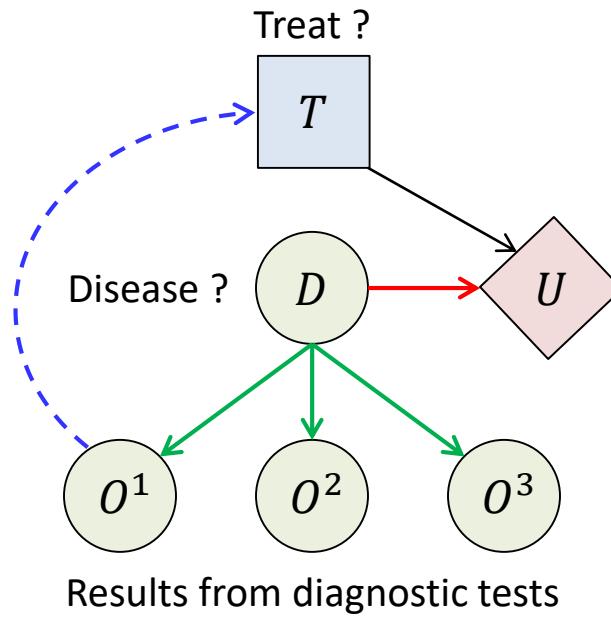
# **Final Exam Reviews**

## L11. Influential Diagram

## Introduction

**Bayesian Network + Decision node + Utility node = Decision network (Influential Diagram)**

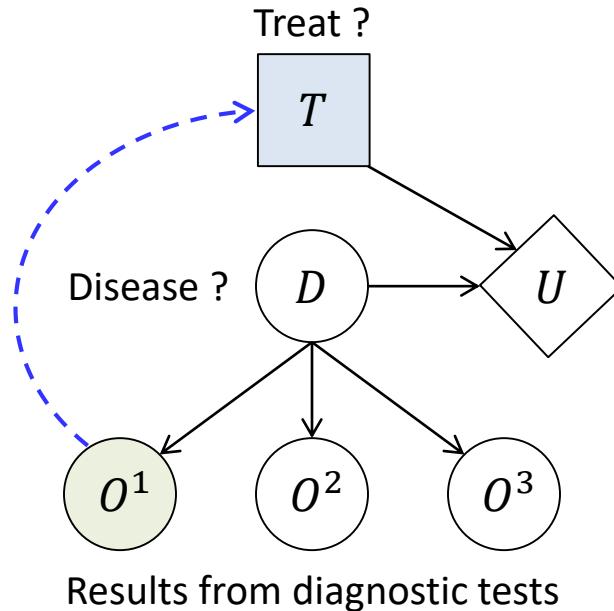
- make rational decisions based on a probabilistic model and utility function



- A chance node** corresponds to a random variable
- A decision node** corresponds to each decision to be made
- A utility node** corresponds to an additive utility component

## Decision Network

Assume we only have a single observation  $O^1 = 1 (= o_1^1)$  from test 1



$$\begin{aligned} EU(t^1 | o_1^1) &= \sum_{o_3} \sum_{o_2} \sum_d P(d, o_2, o_3 | t^1, o_1^1) U(t^1, d, o_1^1, o_2, o_3) \\ &= \sum_d \underbrace{P(d | t^1, o_1^1)}_{\text{Can be computed using many inference methods}} U(t^1, d) \end{aligned}$$

Compare  $EU(t^0 | o_1^1)$  and  $EU(t^1 | o_1^1)$  and chose the treatment that lead maximum EU

## Value of Information

- It may be beneficial to administer additional diagnostic tests to reduce the uncertainty about the decease. Then, how to choose a test type to be conducted?
- Expected utility of optimal action given observation  $o$  :

$$EU^*(o) = \operatorname{argmax}_a EU(a|o)$$

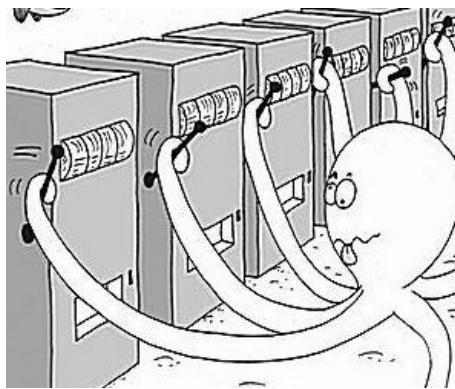
- The value of information (VPI) about new variable  $O^{new}$  (**unobserved**) given the current observation  $o$  (**observed**):

$$\text{VOI}(O^{new}|o) = \left( \sum_{o^{new}} P(o^{new}|o) EU^*(o^{new}, o) \right) - EU^*(o)$$

- The value of information about a variable is the increase in expected utility with the observation of that variable
- VPI can only captures the increase in expected utility → need to consider the cost associated with observing the new information

## L12. Bandit Problem

## An $n$ -Armed Bandit Problem



- There are  $n$  machine
- Each machine  $i$  returns a reward  $r \sim P(\theta_i)$  :  $\theta_i$  is unknown
- $a_t \in \{1, \dots, n\}$  : the choice of machine at time  $t$
- $r_t$  : the reward at time  $t$
- Policy  $\pi$  maps all the history to new action:

$$\pi: [(a_1, r_1), (a_2, r_2), \dots, (a_{t-1}, r_{t-1})] \rightarrow a_t$$

Find the optimal policy  $\pi^*$  that maximizes  $E[\sum_{t=1}^T r_t]$  or  $E[r_T]$

Acquiring new information  
**(exploration)**

trade-off

capitalizing on the information  
available so far  
**(exploitation)**

**Internet add** : Advertising showing strategy for users

**Finance** : Portfolio optimization under unknown return profiles (risk vs mean profit)

**Health care** : Choosing the best treatment among alternatives

**Internet shopping** : Choosing the optimum price (sales v.s. profits)

**Experiment design** : Sequential experimental design (or sequential simulation parameter)

## Action-Value Methods

- If at  $t$ th play, action  $a$  has been chosen  $k_a$  times prior to  $t$ , yielding rewards,  $r_1, r_2, \dots, r_{k_a}$ , the estimated action value for  $a$  is defined as

$$Q_t(a) = \frac{r_1 + r_2 + \dots + r_{k_a}}{k_a} \quad \begin{array}{l} \text{if } k_a = 0, Q_t(a) = 0 \\ \text{if } k_a = \infty, Q_t(a) \rightarrow Q^*(a) \end{array}$$

greedy action selection rule

$$a_t = \operatorname{argmax}_a Q_t(a)$$

- ✓ Always exploits current knowledge to maximize immediate reward
- ✓ spends no time at all sampling apparently inferior actions to see if they might really be better

$\epsilon$  – greedy action selection rule

$$\pi(a) = \begin{cases} 1 - \epsilon + \epsilon/|A(s)| & \text{if } a = a^* \\ \epsilon/|A(s)| & \text{if } a \neq a^* \end{cases}$$

$$\left(1 - \epsilon + \frac{\epsilon}{|A(s)|}\right) \times 1 + \frac{\epsilon}{|A(s)|} (|A(s)| - 1) = 1$$

- ✓ In the limit as the number of plays increases, every  $a$  will be sampled an infinite number of times, guaranteeing  $k_a \rightarrow \infty$  thus  $Q_t(a) \rightarrow Q^*(a)$
- ✓ The probability of selecting the optimum action converges to greater than  $1 - \epsilon$

## Softmax Action Selection

### Disadvantage in $\epsilon$ – greedy action selection:

- When it explores it chooses equally among all actions. That is it is as likely to choose the worst-appearing action as it is to choose the next-to best action

### Solution:

- Vary the action probabilities as a graded function of estimated value
- The greedy action is still given the highest selection probability, but all the others are ranked and weighted according to their values estimates

### Softmax action selection rule

It chooses action  $a$  on the  $t$ th play with probability

$$\frac{e^{Q_t(a)/\tau}}{\sum_{b=1}^n e^{Q_t(b)/\tau}}$$

- ✓  $\tau$  is a positive parameter called the temperature:
  - high  $\tau \rightarrow$  all actions are equiprobable
  - low  $\tau \rightarrow$  greater difference in selection probability for action
  - When  $\tau = 0$ , softmax selection rule become greedy one

Whether softmax action selection or greedy action selection rule is better is unclear and may depend on the task and on human factors (i.e., setting  $\tau$  and  $\epsilon$ )

## Reinforcement Comparison

### Reinforcement Comparison algorithm

- $p_t(a)$  : The preference for each action at time  $t$  (not an actual action value)
- Action determination rule (soft max) :

$$\pi_t(a) = \frac{e^{p_t(a)/\tau}}{\sum_{b=1}^n e^{p_t(b)/\tau}}$$

- After selecting action and observing reward, the preference  $p_t(a)$  is updated :

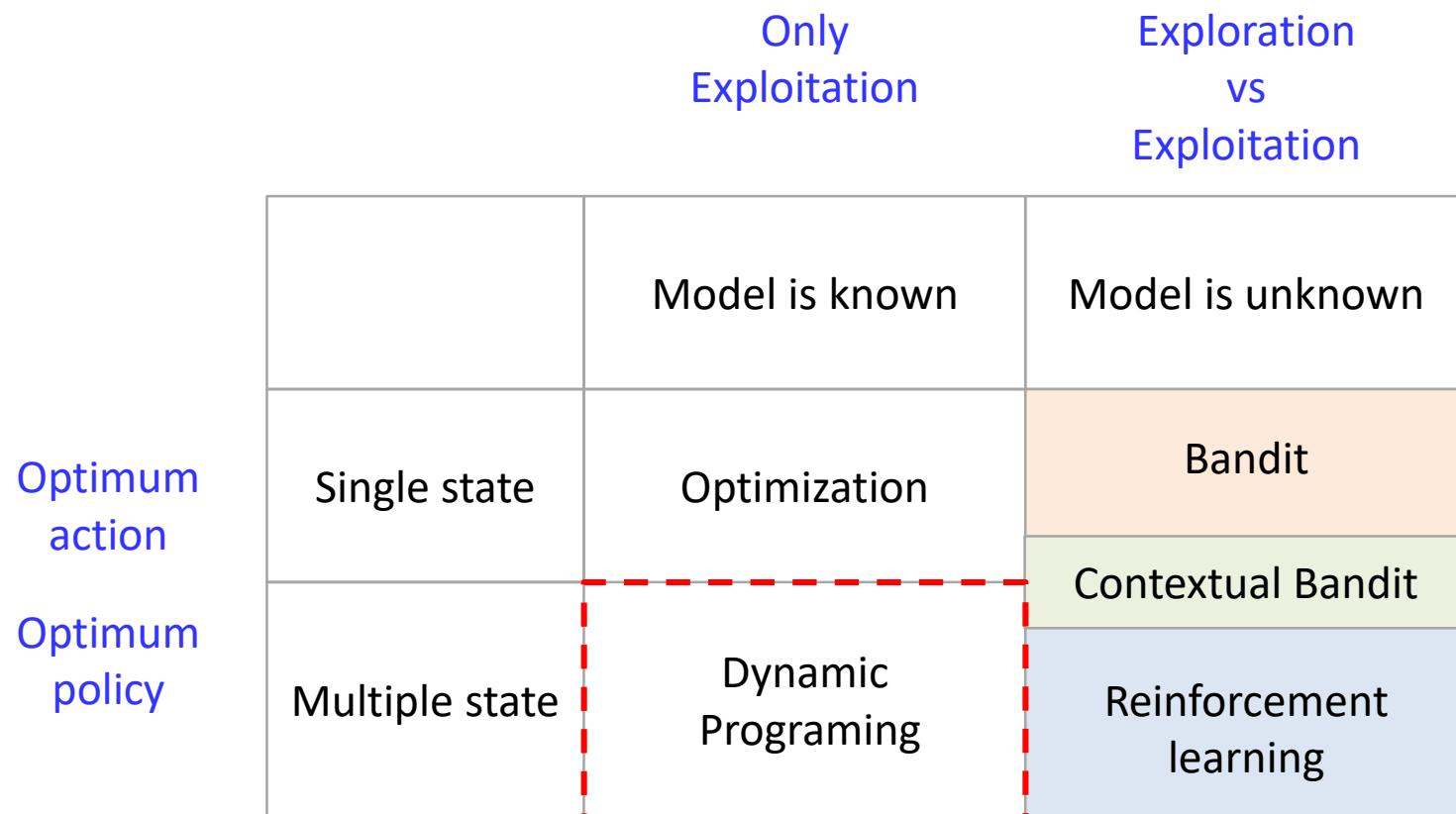
$$p_{t+1}(a_t) = p_t(a_t) + \beta[r_t - \bar{r}_t]$$

- ✓ High rewards should increase the probability of reselecting the action taken
- ✓  $\beta$  is a positive step-size parameter
- ✓ The reference reward  $\bar{r}_t$  (i.e., average reward) is updated using all recently received rewards whichever actions were taken:

$$\bar{r}_{t+1} = \bar{r}_t + \alpha[r_t - \bar{r}_t]$$

Reinforcement comparison method can be very effective sometimes outperforming  $\epsilon$  – greedy method

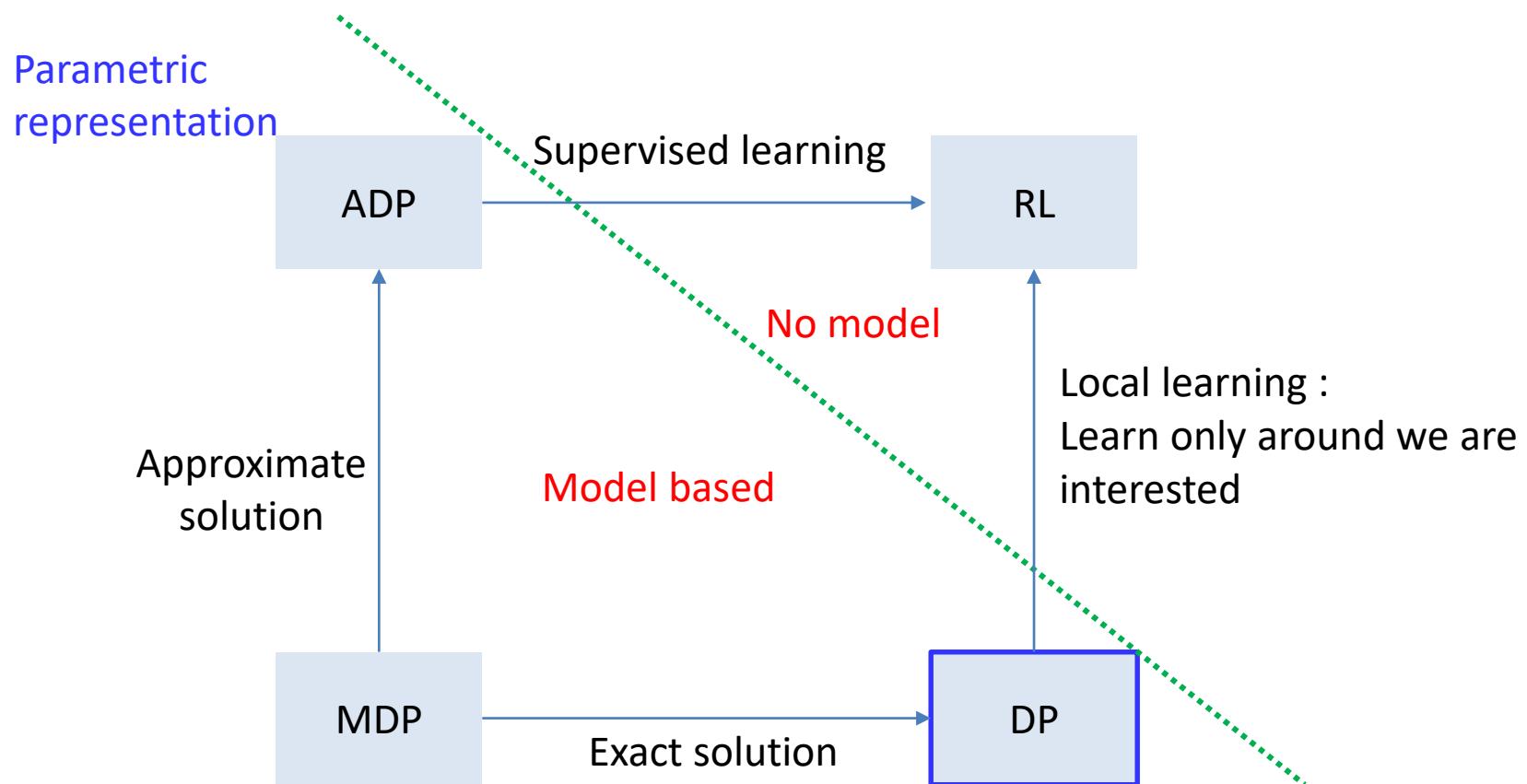
## Relationship with model based approach



- We going to learn Dynamic Programming approach first, and move to Reinforcement learning

## L13. Markov Decision Process (Formulation)

## From MDP to RL



- Usually Tabular representation
- For special case, DP can be used for solving a MDP with continuous space and action

- All other methods can be viewed as attempts to achieve much the same effect as DP, only with less computation and without assuming a perfect model of the environment

## Markov Decision Processes

Finite Markov Decision Process (MDP) :The state and action space are finite

An MDP is defined by:

- A set of states  $s \in \mathcal{S}$
- A set of actions  $a \in \mathcal{A}$
- A transition function  $T(s, a, s') = P(S_{t+1} = s' | S_t = s, A_t = a) = P(s' | s, a)$ 
  - ✓ Probability that a from  $s$  leads to  $s'$  when taking action  $a$
  - ✓ Also called the model or the dynamics
- A reward function  $R(s, a, s')$ 
  - ✓  $r_t = R(s_t, a_t, s_{t+1})$  or  $r_{t+1} = R(s_t, a_t)$
  - ✓ If stochastic,  $R(s, a, s') = \mathbb{E}[r_t + r_{t+1}, \dots | S_t = s, A_t = a, S_{t+1} = s']$
- A start state  $s_0 \in \mathcal{S}$
- A terminal state  $s_T \in \mathcal{S}^+$  (for episodic tasks)

## Value Function & Q function

### Value function (**state value function for $\pi$** )

“How good it is for the agent to be in a given state”

$V^\pi(s)$  : The expected utility received by following policy  $\pi$  from state  $s$

$$V^\pi(s) = \mathbb{E}_\pi(U_t | S_t = s) = \mathbb{E}_\pi\left(\sum_{k=0}^{\infty} \gamma^k r_{t+k} \mid S_t = s\right)$$

$\mathbb{E}_\pi$  : not expectation over policy  $\pi$  but all stochastic state transitions associated with  $\pi$

### Q-function (**action-value function for $\pi$** )

“How good it is for the agent to perform a given action in a given state”

$Q^\pi(s, a)$  : The expected utility of taking action  $a$  from state  $s$ , and then following policy  $\pi$

$$Q^\pi(s, a) = \mathbb{E}_\pi(U_t | S_t = s, A_t = a) = \mathbb{E}_\pi\left(\sum_{k=0}^{\infty} \gamma^k r_{t+k} \mid S_t = s, A_t = a\right)$$

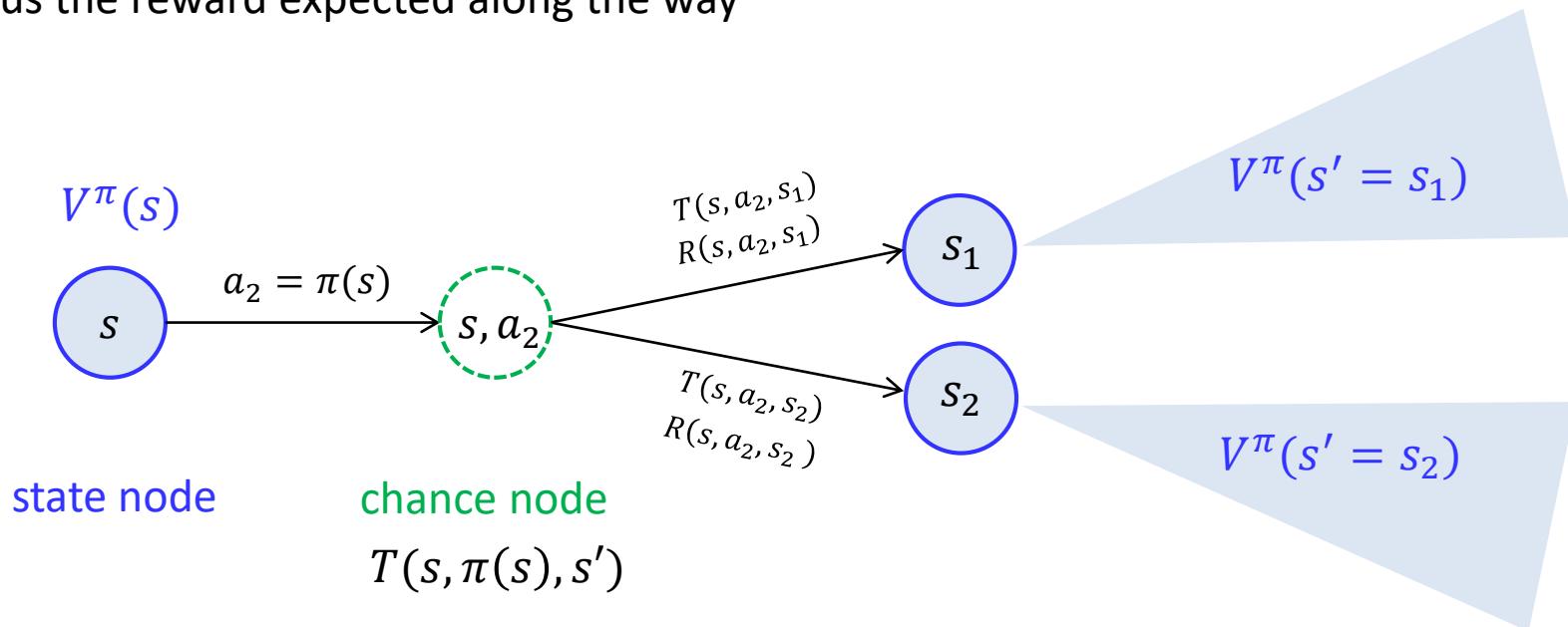
Because the agent can expect to receive in the future depend on what actions it will take  
→ Value and Q functions are defined with respect to a particular policy mapping state  $s \in \mathcal{S}$  to an action  $a \in \mathcal{A}$

## The Bellman Equation for Value Function

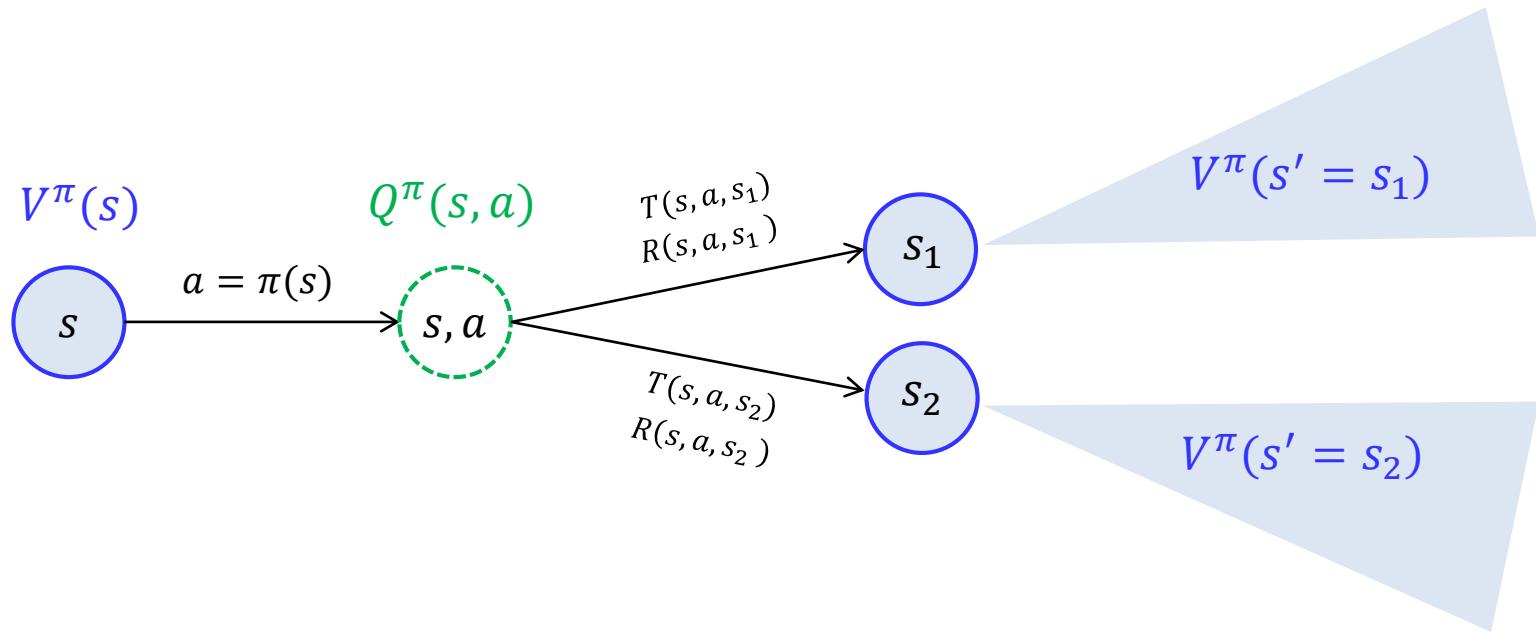
### Recursive Formulation (The Bellman equation)

$$\begin{aligned} V^\pi(s) &= \mathbb{E}_\pi \left( \sum_{k=0}^{\infty} \gamma^k r_{t+k} \mid S_t = s \right) \\ &= \mathbb{E}_\pi \left( r_t + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+1+k} \mid S_t = s \right) \\ &= \sum_{s'} T(s, \pi(s), s') \{ R(s, \pi(s), s') + \gamma \mathbb{E}_\pi \left( \sum_{k=0}^{\infty} \gamma^k r_{t+1+k} \mid S_{t+1} = s' \right) \} \\ &= \sum_{s'} T(s, \pi(s), s') \{ R(s, \pi(s), s') + \gamma V^\pi(s') \} \end{aligned}$$

The value of the start state must equal the (discounted) value of the expected next state, plus the reward expected along the way



## Summary for Value function and Q function



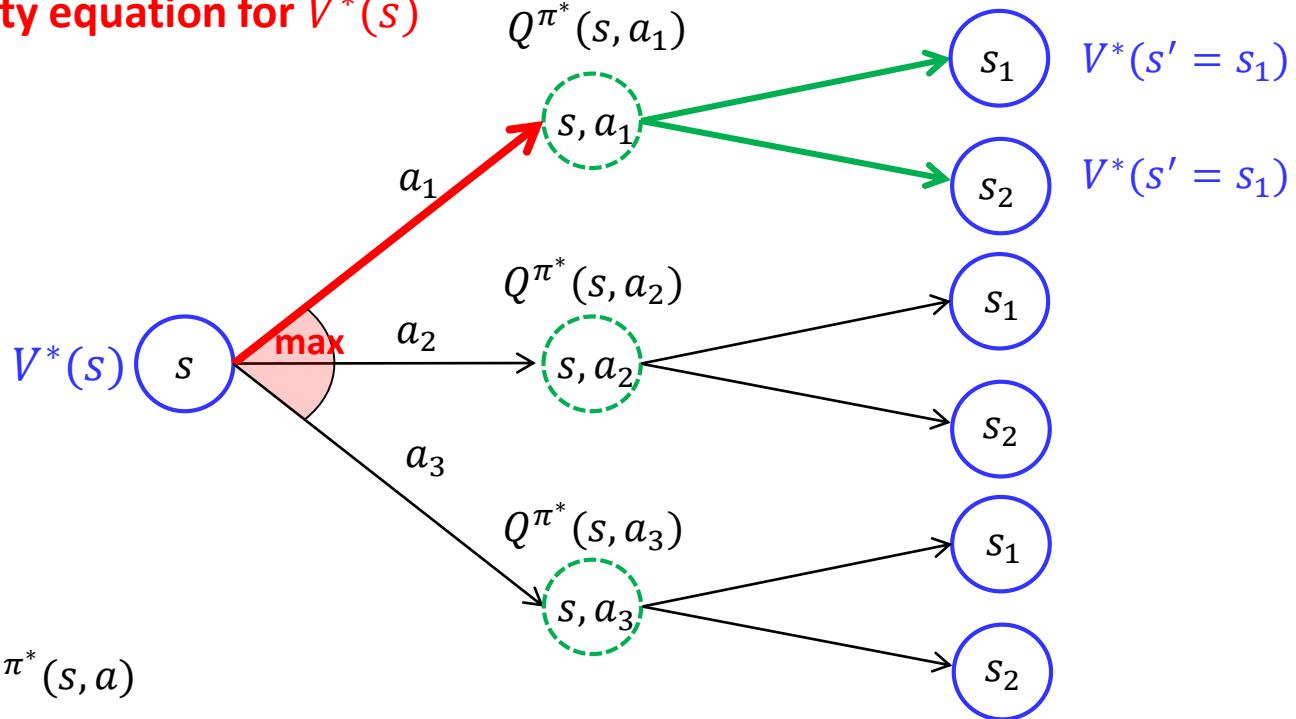
$$V^\pi(s) = \sum_{s'} T(s, \pi(s), s') \{ R(s, \pi(s), s') + \gamma V^\pi(s') \}$$

$$Q^\pi(s, a) = \sum_{s'} T(s, a, s') [R(s, a, s') + \gamma V^\pi(s')]$$

$$V^\pi(s) = Q^\pi(s, \pi(s))$$

## Bellman Optimality Equation for State-Value Function

Bellman optimality equation for  $V^*(s)$



$$V^*(s) = \max_{a \in \mathcal{A}(s)} Q^{\pi^*}(s, a)$$

$$= \max_{a \in \mathcal{A}(s)} \mathbb{E}_{\pi^*} \left( \sum_{k=0}^{\infty} \gamma^k r_{t+k} \mid S_t = s, A_t = a \right)$$

$$= \max_{a \in \mathcal{A}(s)} \mathbb{E}_{\pi^*} \left( r_t + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid S_t = s, A_t = a \right)$$

$\mathbb{E}$  is over transitions associated with  $\pi^*$

$$= \max_{a \in \mathcal{A}(s)} \mathbb{E} (r_t + \gamma V^*(s_{t+1}) \mid S_t = s, A_t = a)$$

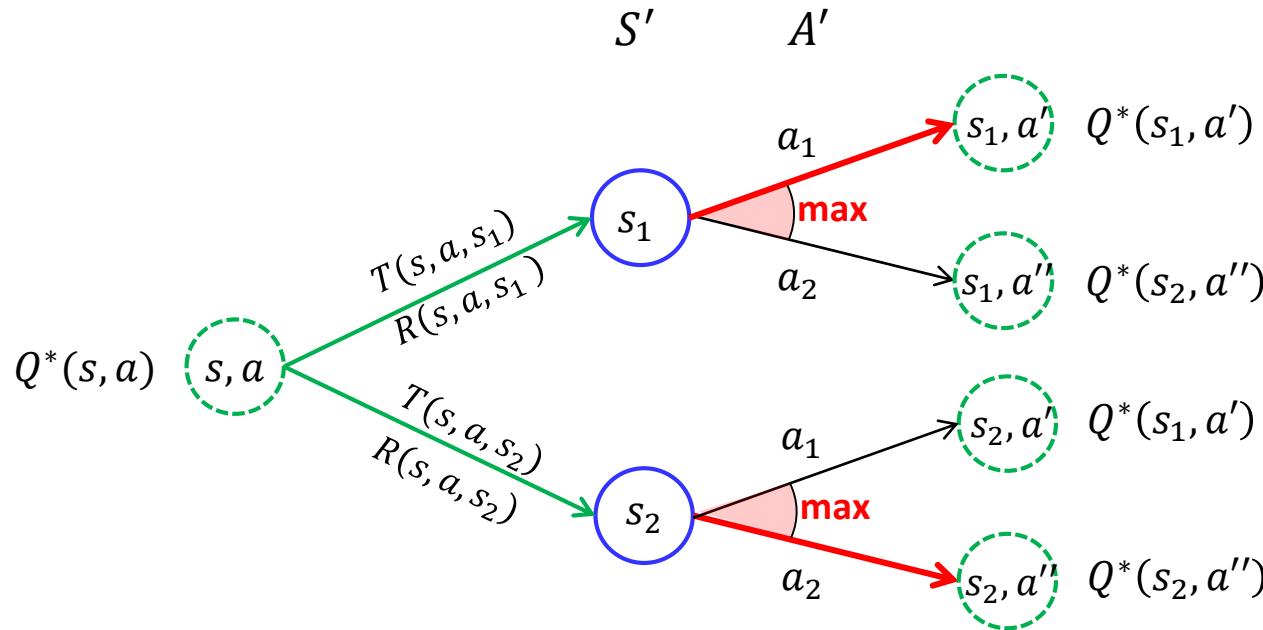
$\mathbb{E}$  is over transitions

$$= \max_{a \in \mathcal{A}(s)} \sum_{s'} T(s, a, s') \{R(s, a, s') + \gamma V^*(s')\}$$

First take optimum action and follow the optimum policy

## Bellman Optimality Equation for State-Action Value Function

Bellman optimality equation for  $Q^*(s, a)$



$$\begin{aligned} Q^*(s, a) &= \mathbb{E} \left\{ r_t + \gamma \max_{a'} Q^*(s', a') | s_t = s, a_t = a \right\} & \mathbb{E} \text{ is over transitions } s' \rightarrow s' \\ &= \sum_{s'} T(s, a, s') \left\{ R(s, a, s') + \gamma \max_{a'} Q^*(s', a') \right\} \end{aligned}$$

First transits by transition probability and take the optimum action for each consequent states

# **L14. Markov Decision Process (Dynamic Programming Approach)**

## Dynamic Programming

- The term **dynamic programming (DP)** refers to a collection of algorithms that can be used to compute optimal policies given **a perfect model of the environment** as a Markov decision process (MDP)
- The key idea of DP (and reinforcement learning) is the use of value functions to organize and structure the search for good policies
- Optimal policies can be derived from the optimal value functions that satisfy the Bellman optimality equations

$$V^*(s) = \max_{a \in \mathcal{A}(s)} \sum_{s'} T(s, a, s') \{R(s, a, s') + \gamma V^*(s')\}$$

$$\begin{aligned} Q^*(s, a) &= \sum_{s'} T(s, a, s') \left\{ R(s, a, s') + \gamma \max_{a'} Q^*(s', a') \right\} \\ &= \sum_{s'} T(s, a, s') \{R(s, a, s') + \gamma V^*(s')\} \quad \because V^*(s') = \max_{a'} Q^*(s', a') \end{aligned}$$



Optimal policy

$$\begin{aligned} \pi^*(s) &= \operatorname{argmax}_a Q^*(s, a) \\ &= \operatorname{argmax}_a \sum_{s'} T(s, a, s') \{R(s, a, s') + \gamma V^*(s')\} \end{aligned}$$

## DP Approaches

### Policy Evaluation

For  $t = 1, \dots$

For each state  $s$ :

$$V_{t+1}^{\pi}(s) \leftarrow \sum_{s'} T(s, \pi(s), s') \{R(s, \pi(s), s') + \gamma V_t^{\pi}(s')\}$$

Policy Iteration

### Policy Improvement

For each state  $s$ :

$$\pi'(s) = \operatorname{argmax}_{a \in \mathcal{A}(s)} \sum_{s'} T(s, a, s') [R(s, a, s') + \gamma V^{\pi}(s')]$$

### Value Iteration

For each state  $s$ :

$$V_{t+1}(s) \leftarrow \max_{a \in \mathcal{A}(s)} \sum_{s'} T(s, a, s') \{R(s, a, s') + \gamma V_t(s)\}$$



### Asynchronous Value iteration

For any single state  $s$ :

$$V(s) \leftarrow \max_{a \in \mathcal{A}(s)} \sum_{s'} T(s, a, s') \{R(s, a, s') + \gamma V(s)\}$$



As long as both processes continue to update all states, the ultimate result is typically the same-convergence to the optimal value function and an optimal policy

## Policy Evaluation

### Policy evaluation :

A method to compute the state-value function  $V^\pi(s)$  for an arbitrary policy  $\pi: \mathcal{S} \rightarrow \mathcal{A}$

$$V^\pi(s) = \sum_{s'} T(s, \pi(s), s') \{R(s, \pi(s), s') + \gamma V^\pi(s')\}$$

- A system of  $|\mathcal{S}|$  simultaneous linear equations in  $|\mathcal{S}|$  unknown

### Algorithm

**Initialize**  $V_{t=0}^\pi(s) \leftarrow 0$  for all states  $s \in S$

**Repeat** (iteration  $t = 0, \dots$ ):

**For each state**  $s$ :

$$V_{t+1}^\pi(s) \leftarrow \sum_{s'} T(s, \pi(s), s') \{R(s, \pi(s), s') + \gamma V_t^\pi(s')\}$$

**Until**  $\max_{s \in \mathcal{S}} |V_{t+1}^\pi - V_t^\pi(s)| \leq \epsilon$

### Full backup:

Each iteration of iterative policy evaluation backs up the value of every state once to produce the new approximate value function  $V_{t+1}^\pi$

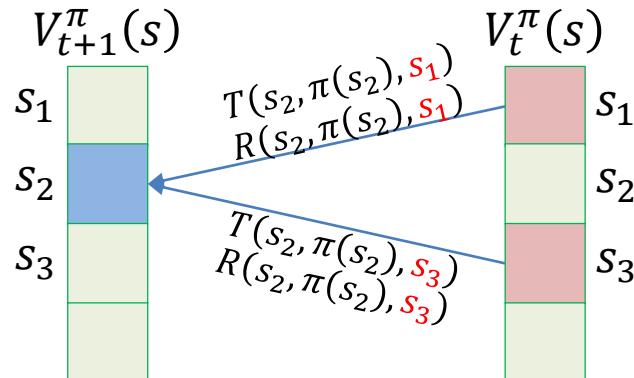
## Policy Evaluation

$$V_{t+1}^{\pi}(s) \leftarrow \sum_{s'} T(s, \pi(s), s') \{R(s, \pi(s), s') + \gamma V_t^{\pi}(s')\}$$

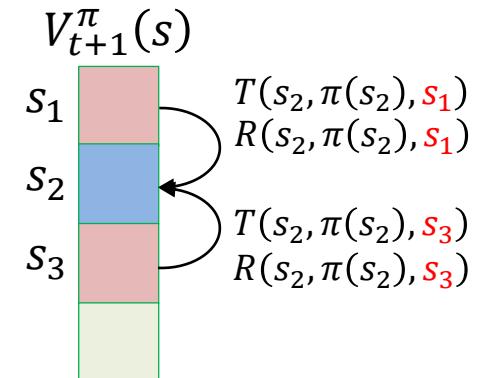
Example:

$$\begin{aligned} V_{t+1}^{\pi}(s_2) &= \sum_{s'} T(s_2, \pi(s_2), s') \{R(s_2, \pi(s_2), s') + \gamma V_t^{\pi}(s')\} \\ &= T(s_2, \pi(s_2), s_1) \{R(s_2, \pi(s_2), s_1) + \gamma V_t^{\pi}(s_1)\} + T(s_2, \pi(s_2), s_3) \{R(s_2, \pi(s_2), s_3) + \gamma V_t^{\pi}(s_3)\} \end{aligned}$$

“Two-arrays” update



“In place” update



Usually faster!  
Less memory

## Policy Improvement

$$\pi'(s) = \operatorname{argmax}_{a \in \mathcal{A}(s)} Q^\pi(s, a)$$

$$\rightarrow Q^{\pi'}(s, \pi'(s)) \geq Q^{\pi}(s, \pi(s)) = V^{\pi}(s)$$

$$x^* = \operatorname{argmax}_x f(x)$$

$\rightarrow f(x^*) \geq f(x)$  for all  $x$

**Improvement criterion** =

Expected reward provided by **changing one step action** and **following the original policy**

## Proof (Policy improvement Theorem)

Policy improvement must give us a strictly better policy  $\pi'(s)$  than the older policy  $\pi(s)$  except when the original policy is already optimal  $\pi(s) = \pi^*(s)$

$$Q^\pi(s, \pi'(s)) \geq V^\pi(s) \rightarrow$$

-----

$$\pi' \geq \pi$$

## Policy Improvement

### Proof (Policy improvement Theorem)

Policy improvement must give us a strictly better policy  $\pi'(s)$  than the older policy  $\pi(s)$  except when the original policy is already optimal  $\pi(s) = \pi^*(s)$

$$Q^\pi(s, \pi'(s)) \geq V^\pi(s) \rightarrow V^{\pi'}(s) \geq V^\pi(s) \text{ for all states } s \in \mathcal{S}$$

$$V^\pi(s) \leq Q^\pi(s, \pi'(s))$$

Given

$$= \mathbb{E}_{\pi'}[r_{t+1} + \gamma V^\pi(s_{t+1}) | s_t = s]$$

$\mathbb{E}_{\pi'}$  is expectation over  $s_{t+1}$  induced by  $\pi'$

$$\leq \mathbb{E}_{\pi'}[r_{t+1} + \gamma Q^\pi(s_{t+1}, \pi'(s_{t+1})) | s_t = s]$$

$\because V^\pi(s_{t+1}) \leq Q^\pi(s_{t+1}, \pi'(s_{t+1}))$

$$= \mathbb{E}_{\pi'}[r_{t+1} + \gamma \mathbb{E}_{\pi'}[r_{t+2} + \gamma V^\pi(s_{t+2})] | s_t = s]$$

$$= \mathbb{E}_{\pi'}[r_{t+1} + \gamma r_{t+2} + \gamma^2 V^\pi(s_{t+2}) | s_t = s]$$

$$\leq \mathbb{E}_{\pi'}[r_{t+1} + \gamma r_{t+2} + \gamma^2 Q^\pi(s_{t+2}, \pi'(s_{t+2})) | s_t = s]$$

$$= \mathbb{E}_{\pi'}[r_{t+1} + \gamma r_{t+2} + \gamma^2 \mathbb{E}_{\pi'}[r_{t+3} + \gamma V^\pi(s_{t+3})] | s_t = s]$$

$$= \mathbb{E}_{\pi'}[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 V^\pi(s_{t+3}) | s_t = s]$$

:

$$= \mathbb{E}_{\pi'}[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 r_{t+4} + \dots | s_t = s]$$

$$= V^{\pi'}(s)$$

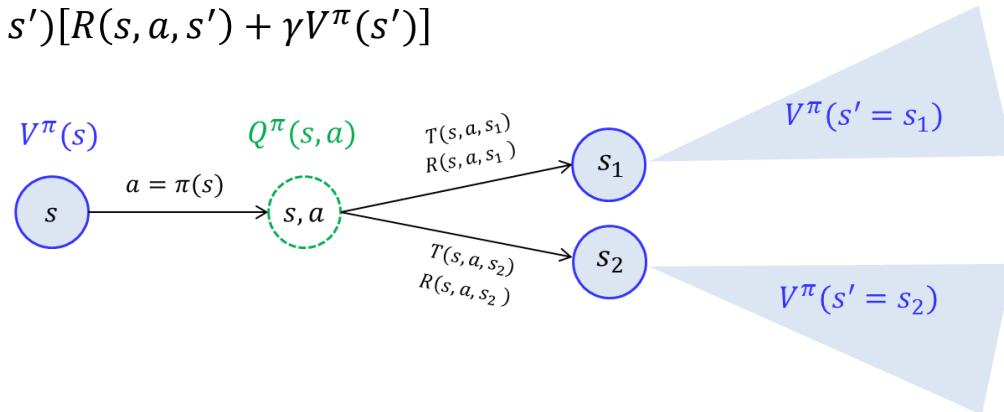
Thus,  $\pi \leq \pi'$

## Policy Improvement

### Policy improvement :

The process of making a new policy  $\pi^{new}$  that improves the original policy  $\pi$ , by making it greedy or nearly greedy with respect of the value function of the original policy

Recall:  $Q^\pi(s, a) = \sum_{s'} T(s, a, s')[R(s, a, s') + \gamma V^\pi(s')]$



### Algorithm

Input : value of policy  $V^\pi(s)$

Output: new policy  $\pi'$

For each state  $s \in \mathcal{S}$

1. Compute  $Q^\pi(s, a) = \sum_{s'} T(s, a, s')[R(s, a, s') + \gamma V^\pi(s')]$  for each  $a$
2. Compute  $\pi'(s) = \operatorname{argmax}_{a \in \mathcal{A}(s)} Q^\pi(s, a)$   
 $= \operatorname{argmax}_{a \in \mathcal{A}(s)} \sum_{s'} T(s, a, s')[R(s, a, s') + \gamma V^\pi(s')]$

## Policy Iteration

### Policy iteration :

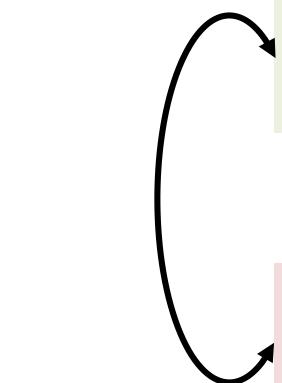
Iterative way of finding the optimum policy through sequence of policy evaluation and policy improvement

Iteration

For  $t = 0, \dots$  until convergence

For each state  $s$ :

$$V_{t+1}^{\pi}(s) \leftarrow \sum_{s'} T(s, \pi(s), s') \{R(s, \pi(s), s') + \gamma V_t^{\pi}(s')\}$$



Converged state value function  $V^{\pi}(s)$

For each state  $s$ :

$$\pi'(s) = \operatorname{argmax}_{a \in \mathcal{A}(s)} Q^{\pi}(s, a)$$

$$= \operatorname{argmax}_{a \in \mathcal{A}(s)} \sum_{s'} T(s, a, s') [R(s, a, s') + \gamma V^{\pi}(s')]$$

## Value Iteration

### Value Iteration:

A method to compute the optimum state-value function  $V^*(s)$  by combining one sweep of policy evaluation and one sweep of policy improvement

### Algorithm

Initialize  $V(s) \leftarrow 0$  for all states  $s \in S$

Repeat

    For each state  $s$ :

$$V(s) \leftarrow \max_a \sum_{s'} T(s, a, s') \{R(s, a, s') + \gamma V(s)\}$$

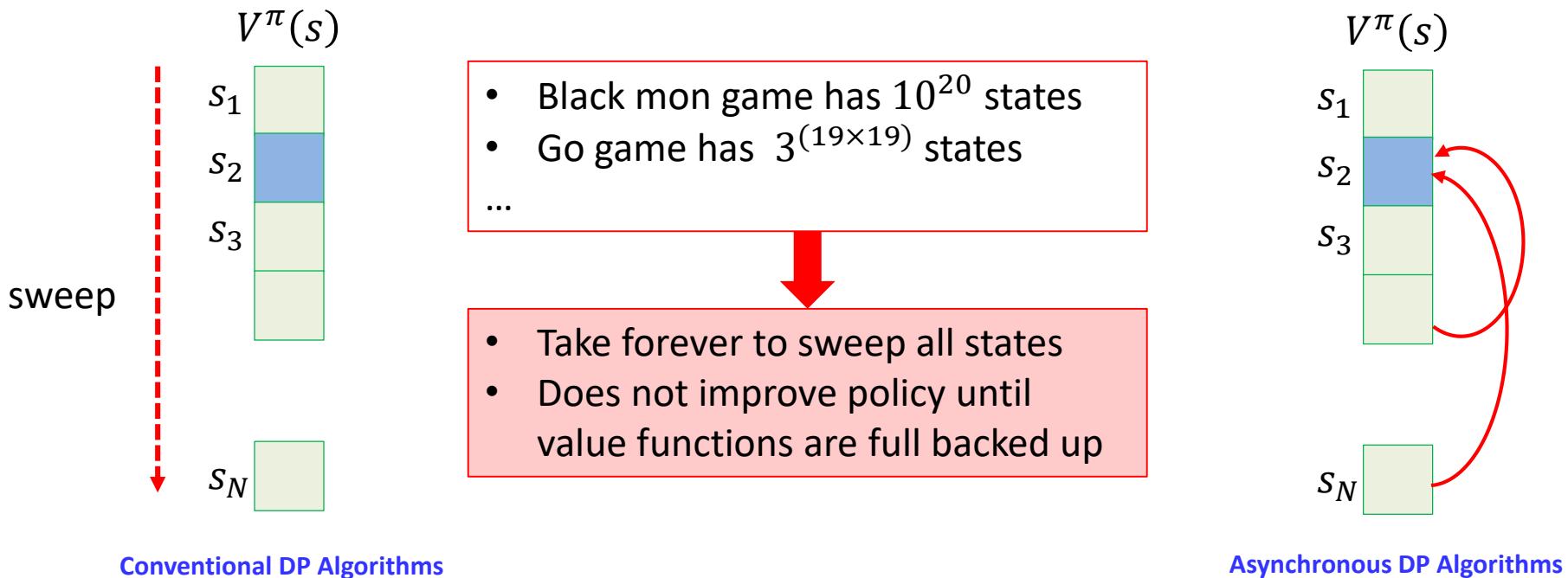
Until  $\max_{s \in S} |V_t(s) - V_{t-1}(s)| \leq \epsilon$

**Optimum policy** can be obtained from the converged  $V^*(s)$ :

$$\pi^*(s) = \operatorname{argmax}_{a \in \mathcal{A}(s)} \sum_{s'} T(s, a, s') \{R(s, a, s') + \gamma V^*(s)\}$$

## Asynchronous DP Algorithms

A major drawback to the DP methods is that they involve operations over the entire state set of the MDP



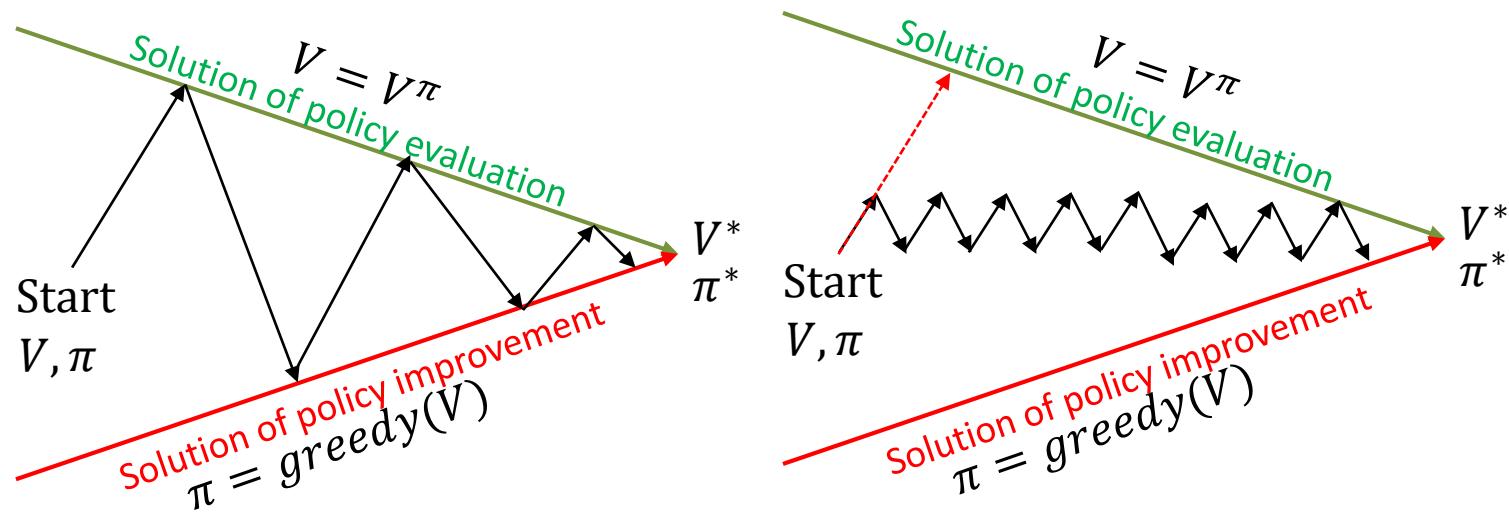
- Back up the values of states in any order whatsoever, using whatever values of other states happen to be available
- Allow great flexibility in selecting states to which backup operations are applied
- Make it easier to intermix computation with real-time interaction: To solve a given MDP, we can run iterative DP algorithm at the same time that an agent is actually experiencing the MDP (Reinforcement Learning !!!!)

## Generalized Policy Iteration

### Generalized Policy Improvement (GPI)

Dynamic Programming  
“full backup”

Asynchronous  
Dynamic Programming



Asynchronous Dynamic Programming is a core concept in Reinforcement learning

# L15. Reinforcement Learning (Monte Carlo Methods)

# From MDP to Reinforcement Learning



## Markov Decision Process (Offline)

- Have mental model of how the world works
- Find policy to collect the maximum rewards

$$\text{Solve } Q^*(s, a) = \sum_{s'} T(s, a, s')[R(s, a, s') + \gamma V^*(s')] \\ \text{Find } \pi^*(s) = \max_a Q^*(s, a)$$



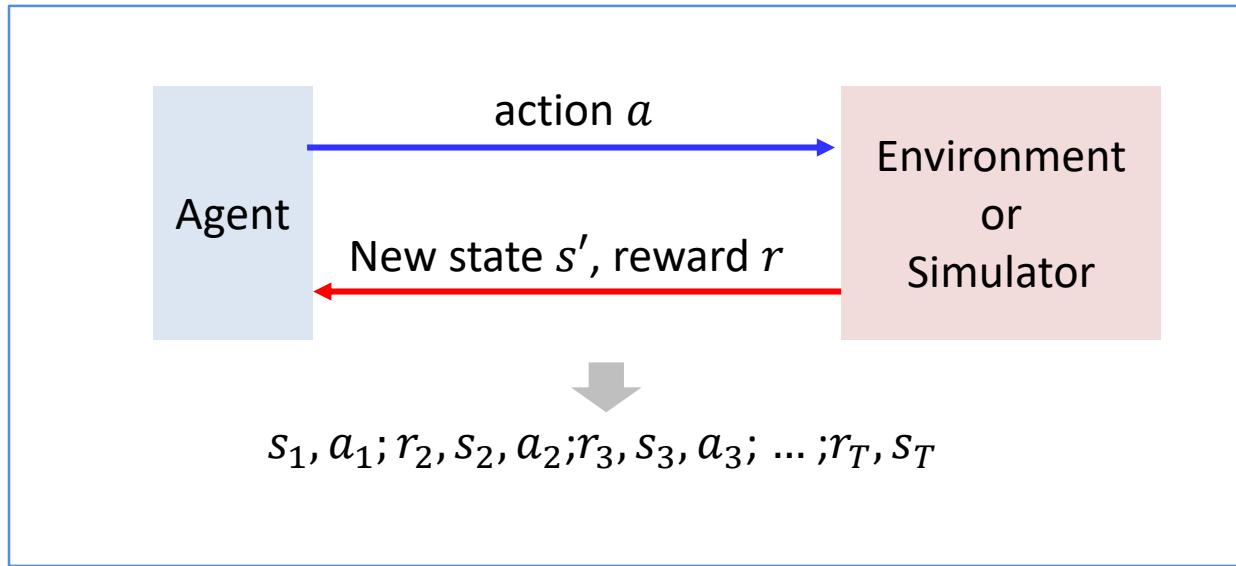
## Reinforcement Learning (Offline & Online)

- Don't know how the world works
- Perform a sequence of actions in the world to maximize the rewards

$$s_1, a_1, r_1; s_2, a_2, r_2; s_3, a_3, r_3; \dots; s_T, a_T, r_T \rightarrow Q^*(s, a) \rightarrow \pi^*(s)$$

- Reinforcement learning is really the way humans work:  
→ we go through life, taking various actions, getting feedback.  
→ We get rewarded for doing well and learn along the way.

## Reinforcement Learning Template



### Template for Reinforcement Learning

For  $t = 1, 2, 3, \dots$

Choose action  $a_t = \pi(s_t)$  (how?) : Decision making

Receive reward  $r_{t+1}$  and observe new state  $s_{t+1}$  (Environment)

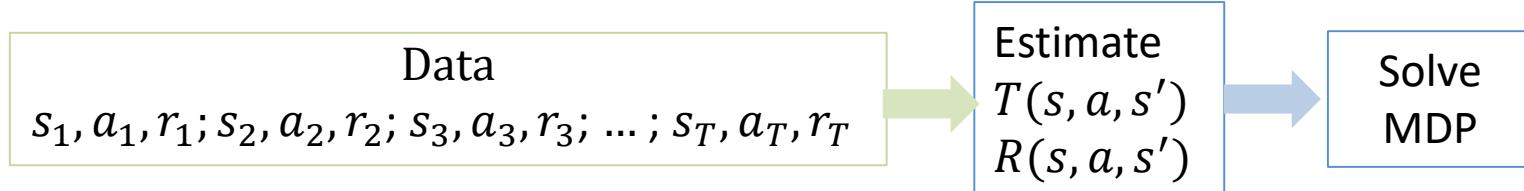
Update parameters associated with  $V(t), Q(s, a)$  (how?) : Learning

## Road Map

- **Monte Carlo Method (Sutton & Barto Ch.5)**
  - Model-Based Monte Carlo method
  - Model-free Monte Carlo method
    - Policy Evaluation
    - Policy Improvement
    - Policy Iteration (Monte Carlo control)
      - ✓ On-policy
      - ✓ Off-policy
- **Temporal Difference Learning (Sutton & Barto Ch.6)**
  - SARSA
  - Q-Learning

## Road Map

- **Model-Based Reinforcement learning**



- **Model-FREE Reinforcement learning**

How to estimate  $V^*(s)$  and  $Q^*(s, a)$

	Monte Carlo method	Temporal Difference methods
How to explore ?	Non-Bootstrap	Bootstrap
On-policy	On-policy Monte Carlo Control	SARSA
Off-policy	Off-policy Monte Carlo Control	Q-Learning

- Episodic based
- Single-data-point based

# Model-Free Monte Carlo Based Methods

How to estimate  $V^*(s)$  and  $Q^*(s, a)$

		Monte Carlo method	Temporal Difference methods
		Non-Bootstrap	Bootstrap
How to explore ?	On-policy	On-policy Monte Carlo Control	SARSA
	Off-policy	Off-policy Monte Carlo Control	Q-Learning

- Episodic based
- Single-data-point based

## Monte Carlo Policy Evaluation

### Key Idea : Monte Carlo Policy Evaluation

- Learn the state-value function  $Q^\pi(s, a)$  for a given policy  $\pi$
- The value of action-state is the expected utility - **expected accumulative future reward** starting from  $s$  and following the policy  $\pi$

$$s_t \in \mathcal{S} = \{s^1, s^2, s^3\}, a_t \in \mathcal{A} = \{a^1, a^2, a^3\}$$

(State, Action, Reward ) pairs generated by policy  $\pi$

Episode 1:  $(s^1, a^2, 1); (s^3, a^1, 5); (s^2, a^3, 3), (s^1, a^3, 10), (s^2, a^2, 2)$

Episode 2:  $(s^1, a^1, 5); (s^2, a^2, 2); (s^1, a^2, 1); (s^2, a^3, 3), (s^1, a^3, 10)$

Episode 3:  $(s^2, a^3, 3); (s^1, a^2, 1); (s^3, a^1, 5); (s^1, a^3, 10), (s^2, a^2, 2)$

$(\gamma = 1)$

Episode 1:  $Q^\pi(s^1, a^2) = 1 + 5 + 3 + 10 + 2 = 21$

Episode 2:  $Q^\pi(s^1, a^2) = 1 + 3 + 10 = 14$

First visit to  $s$

Episode 3:  $Q^\pi(s^1, a^2) = 1 + 5 + 10 + 2 = 19$

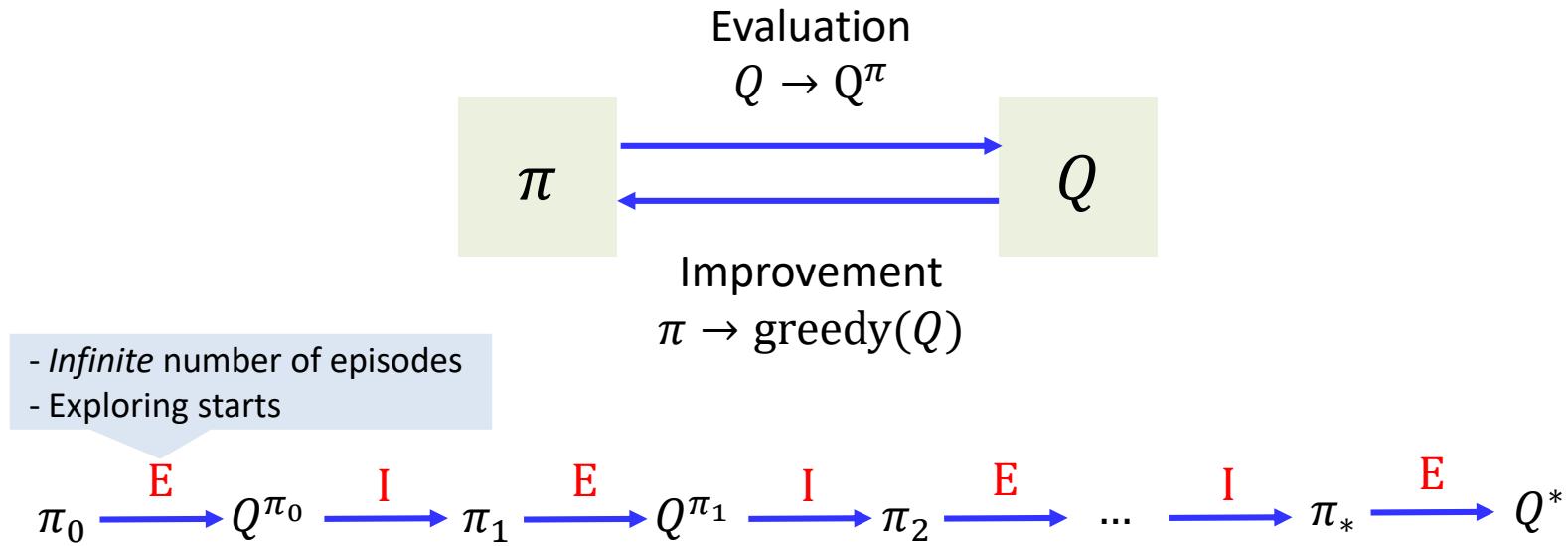
$Q^\pi(s^1, a^2) = \text{average of accumulated reward over all episodes}$

$$= \frac{21+14+19}{3} = 14.6$$

## Monte Carlo Control

### Key Idea : Monte Carlo Control

- Idea of generalized policy iteration (GPI)
- Monte Carlo Policy Evaluation + Policy improvement



- **Policy Evaluation**
  - ✓ The value function is repeatedly altered to more closely approximate the value function for the current policy  $\pi$
- **Policy Improvement**
  - ✓ The policy is repeatedly improved with respect to the current action value function  $Q$

## Monte Carlo control algorithm assuming exploring starts

### Algorithm : Monte Carlo ES Control

Initialize, for all  $s \in S, a \in A(s)$

$$Q(s, a) \leftarrow \text{arbitrary}$$

$$\pi(s) \leftarrow \text{arbitrary}$$

$$U(s, a) \leftarrow \text{empty list}$$

Repeat forever:

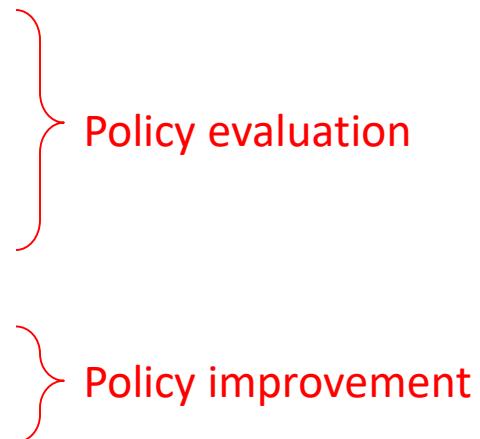
(a) Generate *an episode* using *exploring starts* and  $\pi$

(b) For each pair  $(s, a)$  appearing in the episode:

$U \leftarrow \text{utility following the first occurrence of } s, a$

Append  $U$  to  $U(s, a)$

$Q(s, a) \leftarrow \text{average}(U(s, a))$



(c) For each  $s$  in the episode:

$\pi(s) \leftarrow \underset{a}{\operatorname{argmax}} Q(s, a)$

In a single episode, both Policy evaluation and policy improvement proceed together

## On-policy Monte Carlo Control

### Algorithm : $\epsilon$ – soft On-Policy Monte Carlo Control

Initialize, for all  $s \in \mathcal{S}, a \in \mathcal{A}(s)$

$$Q(s, a) \leftarrow \text{arbitrary}$$

$\pi \leftarrow \text{an arbitrary } \epsilon\text{-soft policy}$

$$U(s, a) \leftarrow \text{empty list}$$

Repeat forever:

(a) Generate *an episode* using  $\pi$

(b) For each pair  $(s, a)$  appearing in the episode:

$U \leftarrow \text{utility following the first occurrence of } s, a$

Append  $U$  to  $U(s, a)$

$Q(s, a) \leftarrow \text{average}(U(s, a))$

$\} \quad \text{Policy evaluation}$

(c) For each  $s$  in the episode:

$a^* \leftarrow \underset{a \in \mathcal{A}(s)}{\operatorname{argmax}} Q(s, a)$

For all  $a \in \mathcal{A}(s)$

$$\pi(s, a) = \begin{cases} 1 - \epsilon + \epsilon / |\mathcal{A}(s)| & \text{If } a = a^* \\ \epsilon / |\mathcal{A}(s)| & \text{If } a \neq a^* \end{cases}$$

$\} \quad \text{Policy improvement}$

# Model-Free Monte Carlo Based Methods

How to estimate  $V^*(s)$  and  $Q^*(s, a)$

Monte Carlo method

Temporal Difference methods

		Non-Bootstrap	Bootstrap
How to explore ?	On-policy	On-policy Monte Carlo Control	SARSA
	Off-policy	Off-policy Monte Carlo Control	Q-Learning

- Episodic based
- Single-data-point based

### Key Idea : Off-policy algorithm

- Follows the behavior policy while learning about and improving the estimation policy
- *Behavior* policy  $\pi'(s, a)$ 
  - ✓ The policy used to generate behavior
  - ✓ Requires that the behavior policy have a nonzero probability of selecting all actions that might be selected by the estimation policy (e.g.,  $\epsilon$  – soft policy )
- *Estimation* policy  $\pi(s, a)$ 
  - ✓ The policy that is evaluated and improved
  - ✓  $\pi$  can be deterministic
  - ✓  $\pi$  can be the greedy policy with respect to  $Q$  (an estimation  $Q^\pi$ )

### Disadvantages

→ Learning can be slow

## Model-Free Monte Carlo Based Methods

How to estimate  $V^*(s)$  and  $Q^*(s, a)$

Monte Carlo method

Temporal Difference methods

		Non-Bootstrap	Bootstrap
How to explore ?	On-policy	On-policy Monte Carlo Control	SARSA
	Off-policy	Off-policy Monte Carlo Control	Q-Learning

- Episodic based
- Single-data-point based

# Introduction

## Dynamic Programming (DP) Methods

pros: Update estimates based in part on other learned estimates, without waiting for a final outcome (bootstrap)

cons: Need explicit model

## Monte Carlo (MC) Methods

pros: Learn directly from raw experience without a model

cons: Need to wait until the end of episode to observe expected reward

## Temporal-Difference (TD) Learning

pros: Learn directly from raw experience without a model

MC

+

pros: Update estimates based in part on other learned estimates, without waiting for a final outcome (bootstrap)

DP

Model free

On line  
Incremental

TD Generalized Policy iteration for

TD Policy Evaluation

+

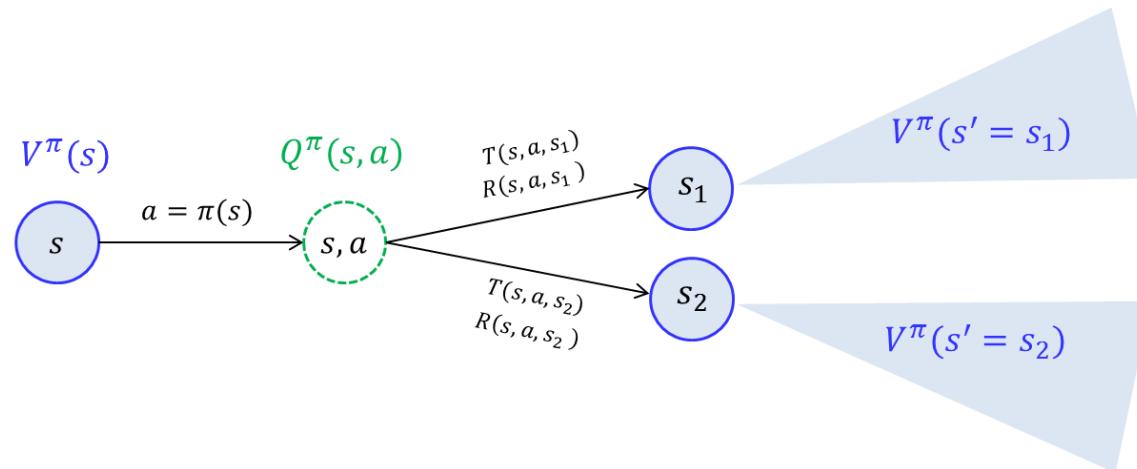
TD Policy Improvement

On-Policy TD Control (SARSA)

Off-Policy Q-learning Control

## Recall : Value function

$$\begin{aligned}
 V^\pi(s) &= \mathbb{E}_\pi(U_t | s_t = s) \\
 &= \mathbb{E}_\pi(r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots | s_t = s) \text{ Complete episode} \\
 &= \mathbb{E}_\pi(\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s) \\
 &= \mathbb{E}_\pi(r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} | s_t = s) \\
 &= \mathbb{E}_\pi(r_{t+1} + \gamma V^\pi(s_{t+1}) | s_t = s) \quad \text{Bootstrapping}
 \end{aligned}$$



## Temporal Difference Policy Evaluation

$$\begin{aligned} V^\pi(s) &= \mathbb{E}_\pi(U_t | s_t = s) \\ &= \mathbb{E}_\pi\left(\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s\right) \\ &= \mathbb{E}_\pi\left(r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_t = s\right) \\ &= \mathbb{E}_\pi(r_{t+1} + \gamma V^\pi(s_{t+1}) | s_t = s) \end{aligned} \quad \text{Bootstrapping}$$

### Temporal Difference Policy Evaluation ;TD(0) :

After visiting  $s_t$  and transiting to  $s_{t+1}$  with a single reward  $r_{t+1}$

$$V(s_t) \leftarrow V(s_t) + \alpha [r_{t+1} + \gamma V(s_{t+1}) - V(s_t)]$$

↳ Target

- Bootstrapping: the TD method updates the state value using the previous estimations
- The TD target is an estimate because
  - ✓ it uses the current estimate of  $V(s_t)$ ,
  - ✓ it samples the expected value

$$\mathbb{E}_\pi(r_{t+1} + \gamma V^\pi(s_{t+1}) | s_t = s)$$

## Temporal Difference Policy Evaluation

### Algorithm : Tabular $TD(0)$ for estimating $V^\pi$

Initialize  $V(s)$  arbitrarily,  $\pi$  to the policy to be evaluated

**Repeat** (for each episode):

    Initialize  $s$

**Repeat** (for **each step** of episode)

$a \leftarrow$  action given by  $\pi$  for  $s$

        Take action  $a$ ; observe reward  $r$  and next state  $s'$

$V(s) \leftarrow V(s) + \alpha[r + \gamma V(s') - V(s)]$

$s \leftarrow s'$

    Until  $s$  is terminal

- Simple backups (MC method and TD methods) : Use a single sample success state

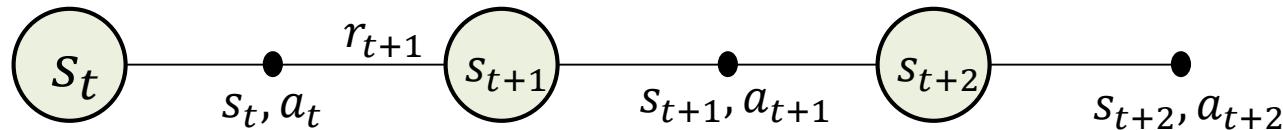
Recall:

- Full Backups (DP approach) : Use complete distribution of all possible successors

$$V^\pi(s) = \sum_{s'} T(s, \pi(s), s') \{R(s, \pi(s), s') + \gamma V^\pi(s')\}$$

## Temporal Difference Policy Evaluation for Q function

As we estimate state value  $V(s)$ , we can estimate  $Q(s, a)$  using a TD method



### Temporal Difference Policy Evaluation for $Q(s, a)$ function

On each  $(s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1})$  for a single episode:

*Note that the action taken is given as data*

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha [r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - \underbrace{Q(s_t, a_t)}_{\text{Current estimate}}]$$

-----  
↳ Target

# Model-Free Monte Carlo Based Methods

How to estimate  $V^*(s)$  and  $Q^*(s, a)$

		Monte Carlo method	Temporal Difference methods
		Non-Bootstrap	Bootstrap
How to explore ?	On-policy	On-policy Monte Carlo Control	SARSA
	Off-policy	Off-policy Monte Carlo Control	Q-Learning

- Episodic based
- Single-data-point based

## Sarsa: On-Policy TD Control

### SARSA Algorithm

Initialize  $Q(s, a)$  arbitrarily

Repeat (for each episode):

    Initialize  $s$

    Choose  $a$  from  $s$  using policy derived from  $Q$  (e.g.,  $\epsilon$  – greedy)

    Repeat (for **each time step** of episode):

        Take action  $a$  given  $s$ , observe  $r, s'$

        Choose  $a'$  from  $s'$  using policy derived from  $Q$  (e.g.,  $\epsilon$  – greedy) **Behavioral policy**

$$Q_\pi(s, a) \leftarrow Q_\pi(s, a) + \eta(r + \gamma Q_\pi(s', a') - Q_\pi(s, a))$$

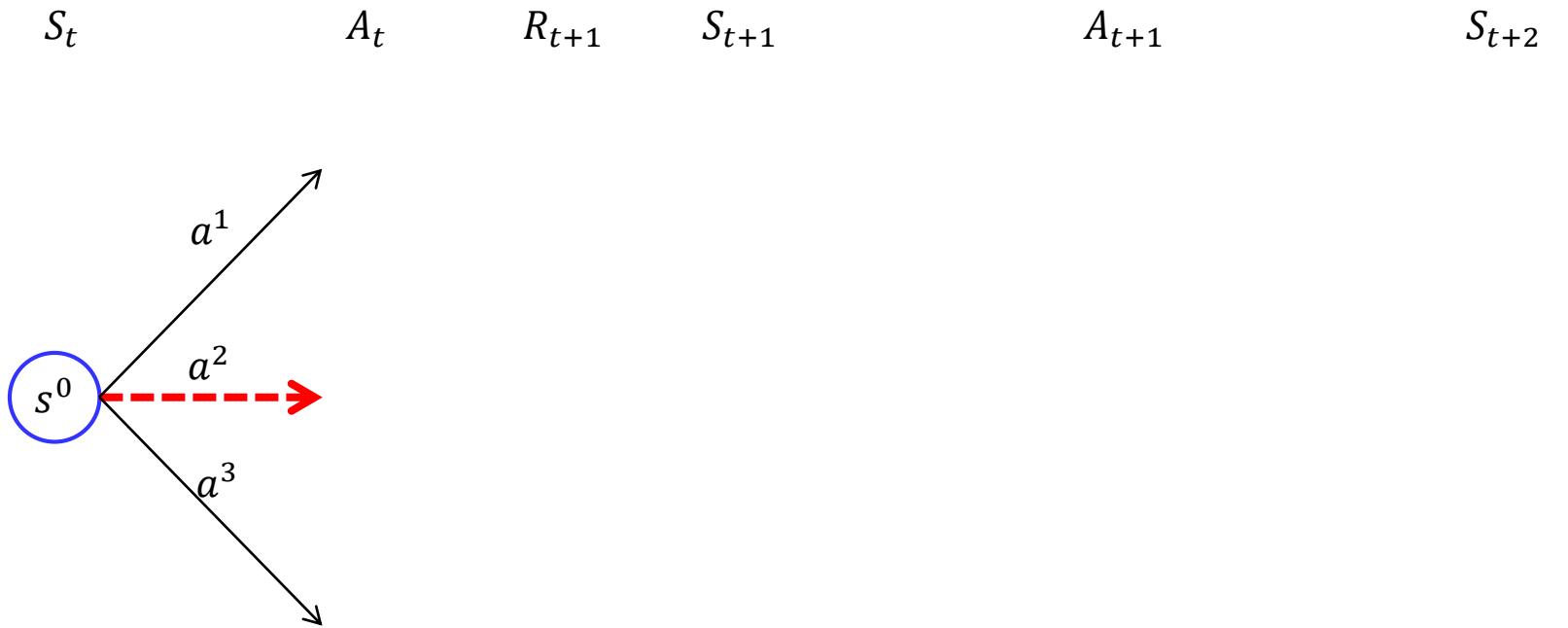
**II**  
**Estimation policy**

$$s \leftarrow s'; a \leftarrow a';$$

Until  $s$  is terminal

- As in all **on-policy methods**, we continually estimate  $Q^\pi$  for the behavioral policy, and the same time change  $\pi$  toward greediness with respect to  $Q^\pi$
- Converges with
  - ✓ All state-action pairs are visited an infinite number of times
  - ✓ The policy converges in the limit to the greedy policy (i.e.,  $\epsilon$  – greedy with  $\epsilon = 1/t$ )

## Sarsa: On-Policy TD Control

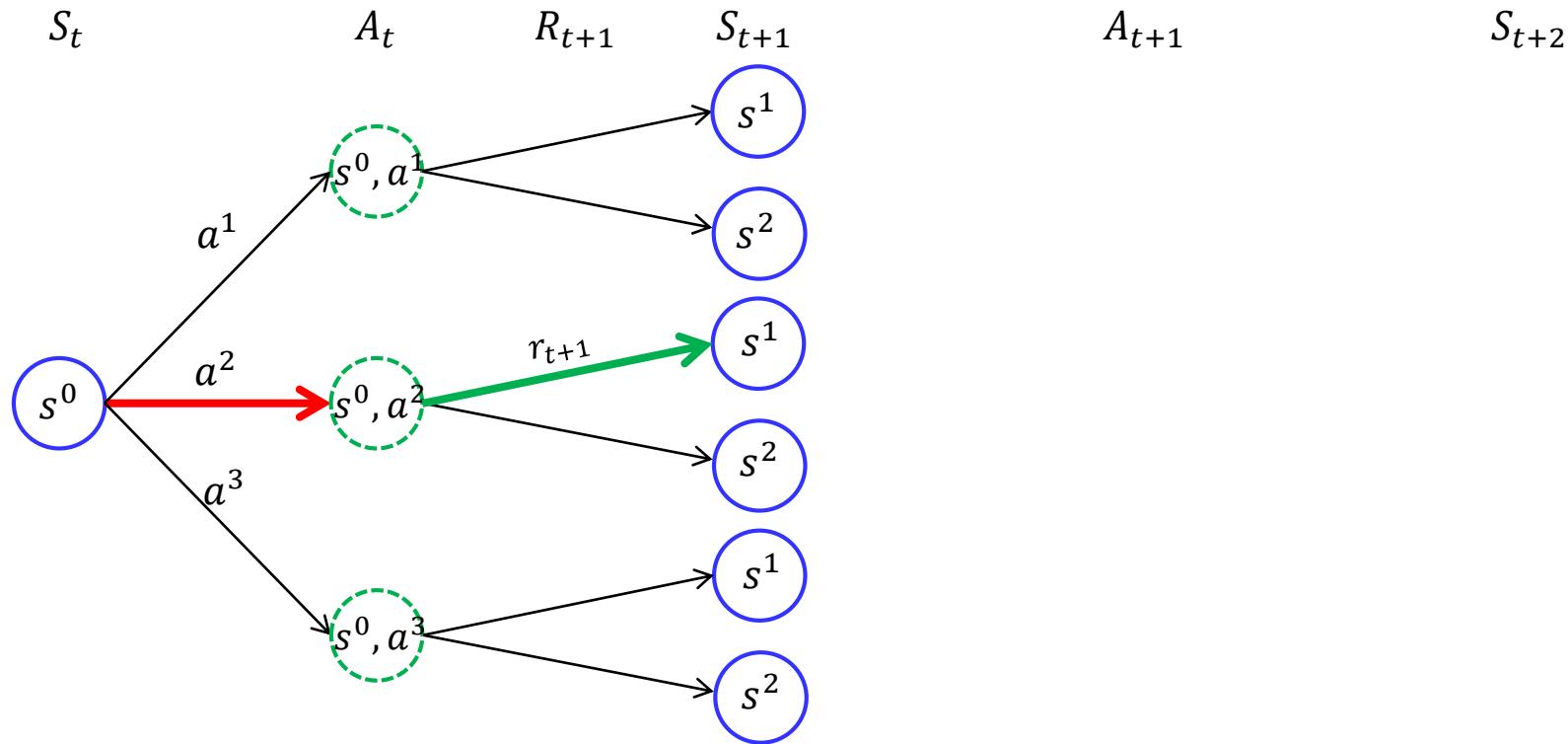


Choose  $a_t$  from  $s_t = s^0$  using current  $Q$

$$a_t = \begin{cases} \underset{a}{\operatorname{argmax}} Q(s_t = s^0, a) & \text{with prob } 1 - \epsilon \\ \text{random action} & \text{with prob } \epsilon \end{cases}$$

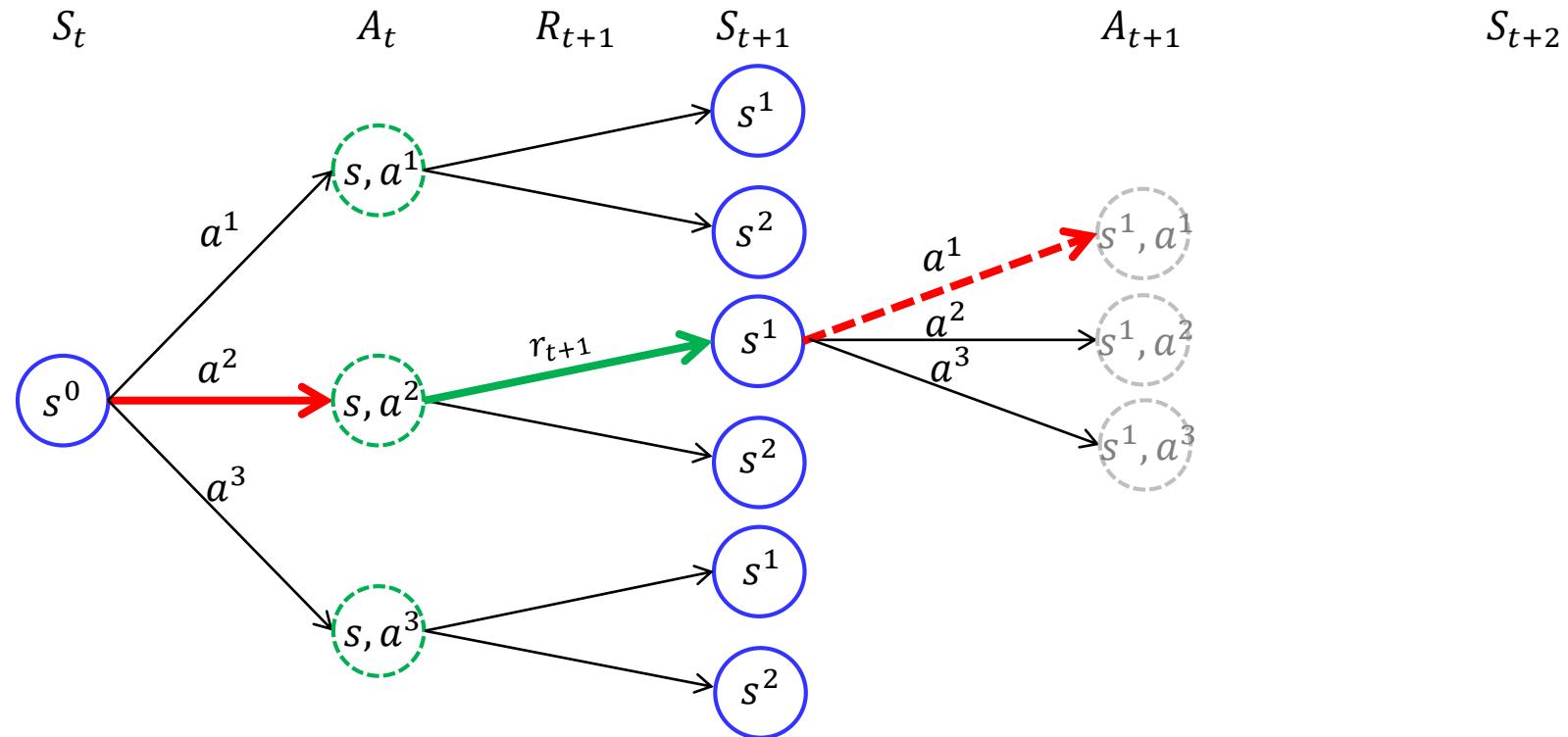
Assume  $a^2$  is chosen

## Sarsa: On-Policy TD Control



Take action  $a_t = a^2$  given  $s_t = s^0$  and observe  $r_{t+1}$  and  $s_{t+1} = s^1$

## Sarsa: On-Policy TD Control

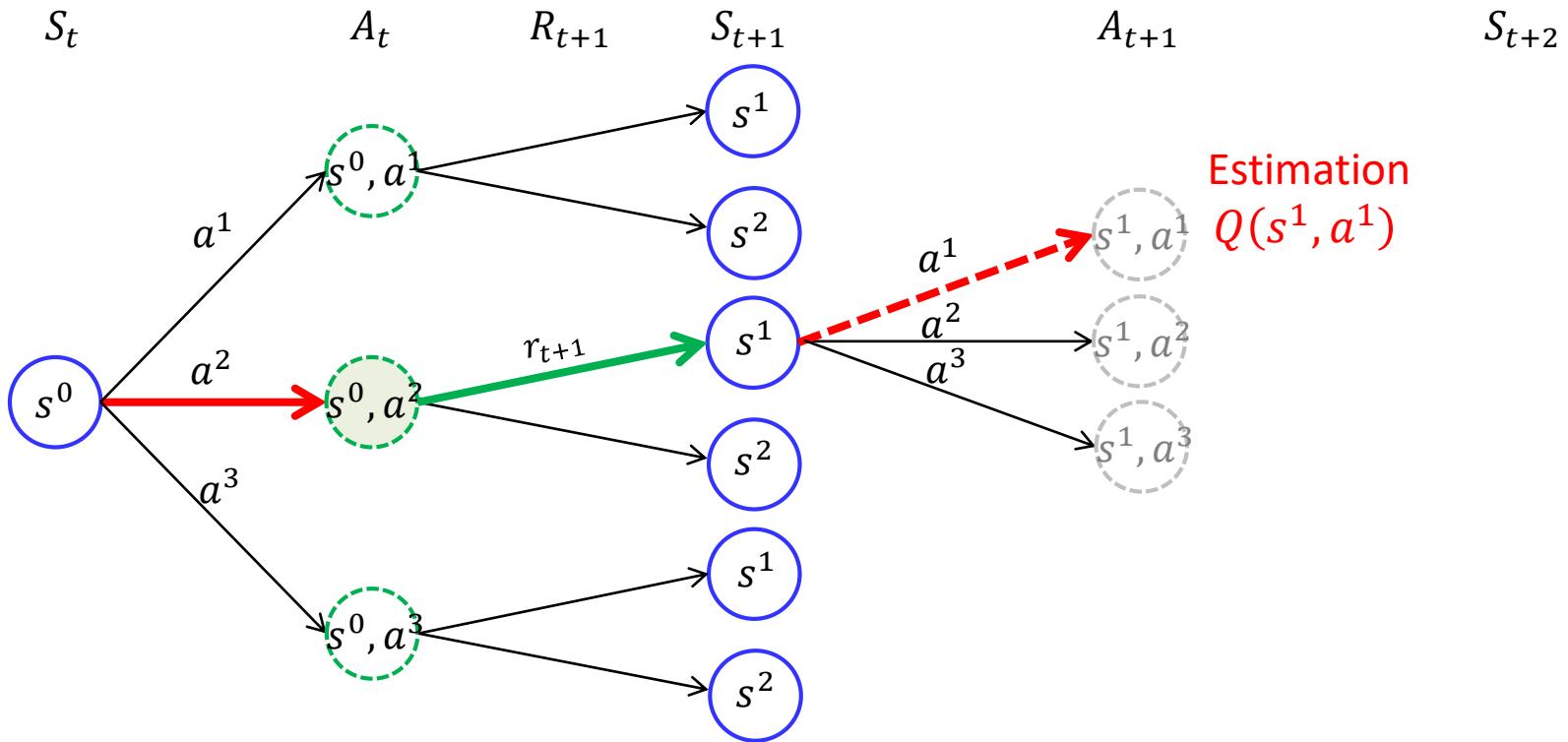


Choose  $a_{t+1}$  from  $s_{t+1} = s^1$  using current  $\textcolor{blue}{Q}$

$$a_{t+1} = \begin{cases} \underset{a}{\operatorname{argmax}} \textcolor{blue}{Q}(s_{t+1} = s^1, a) & \text{with prob } 1 - \epsilon \\ \text{random action} & \text{with prob } \epsilon \end{cases}$$

Assume  $a^1$  is chosen

## Sarsa: On-Policy TD Control

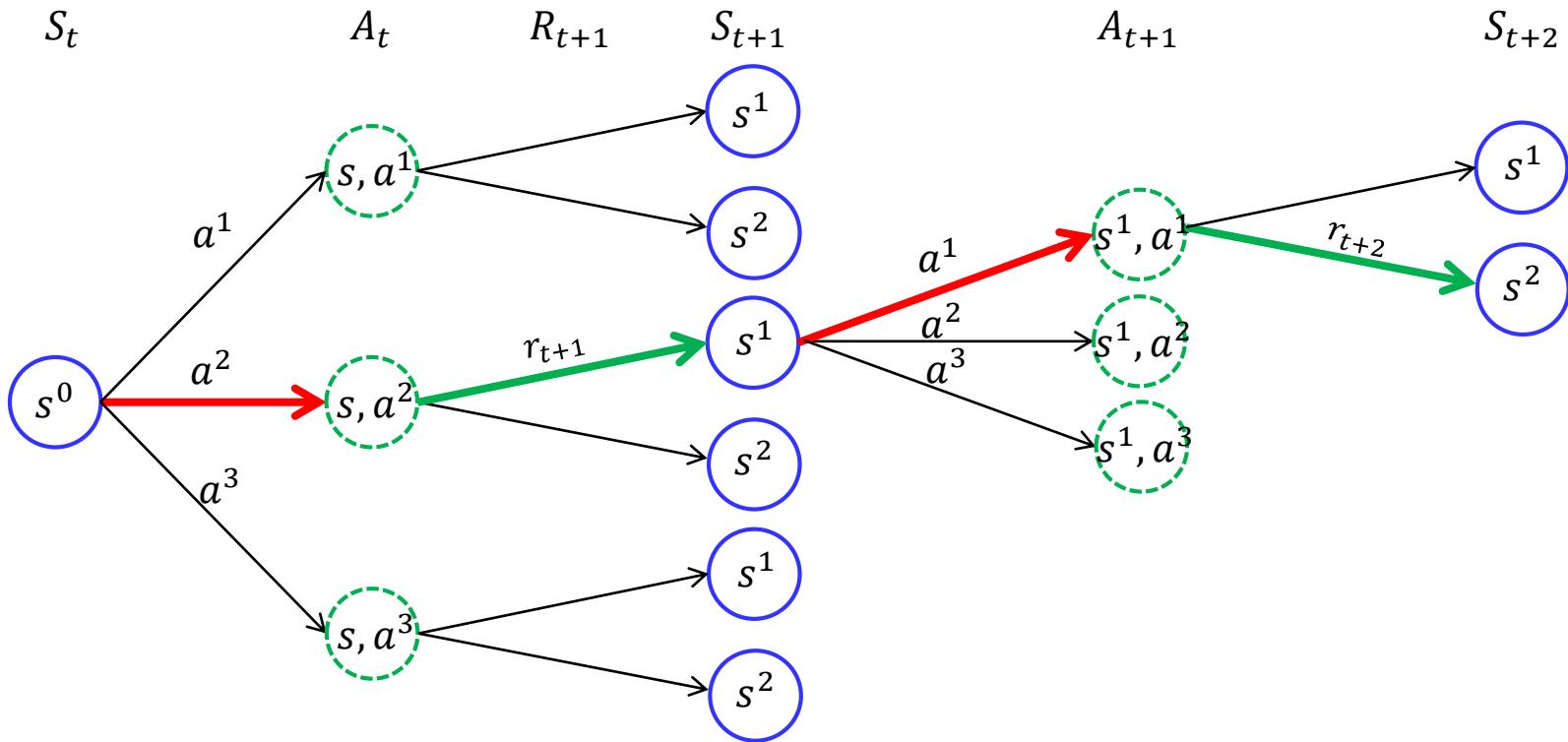


Update  $Q$  function with the estimation  $Q(s_{t+1}, a_{t+1})$

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]$$

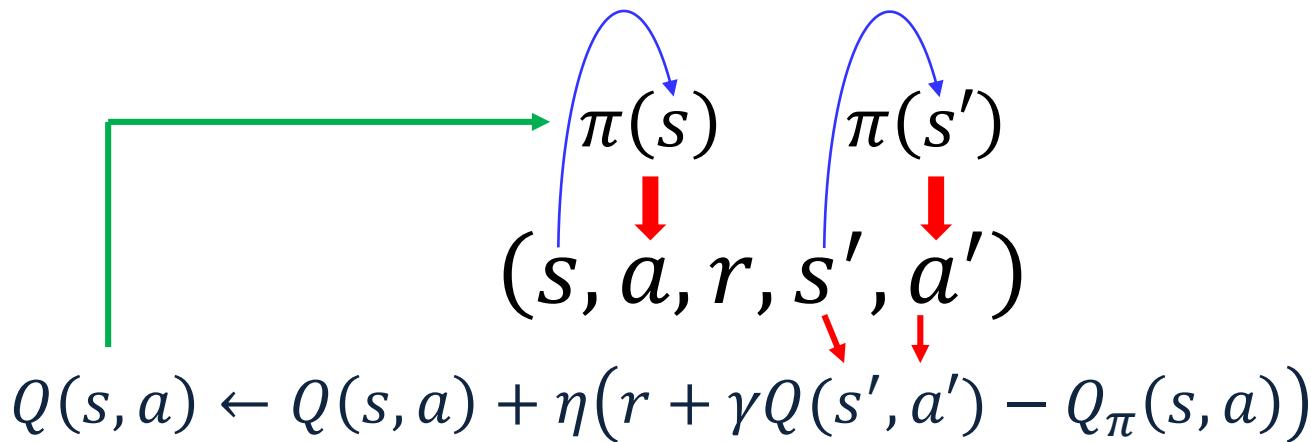
$$\rightarrow Q(s^0, a^2) \leftarrow Q(s^0, a^2) + \alpha[r_{t+1} + \gamma Q(s^1, a^1) - Q(s^0, a^2)]$$

## Sarsa: On-Policy TD Control



Take action  $a_{t+1} = a^1$  given  $s_{t+1} = s^1$  and observe  $r_{t+2}$  and  $s_{t+2} = s^2$

## Why Q-learning is considered as Off-Policy method



## Classification of RL

		How to estimate $V^*(s)$ and $Q^*(s, a)$	
		Monte Carlo method	Temporal Difference methods
How to explore ?	Non-Bootstrap	Bootstrap	
	On-policy	On-policy Monte Carlo Control	SARSA
	Off-policy	Off-policy Monte Carlo Control	Q-Learning (SARSmaxA)

• Episodic based      • Single-data-point based

## Q-Learning: Off-Policy TD Control

### On-Policy TD Control (SARSA)

Choose  $a'$  from  $s'$  using policy derived from  $Q$  (e.g.,  $\epsilon$ -greedy)

$$Q(s, a) \leftarrow Q(s, a) + \eta(r + \gamma Q(s', a') - Q(s, a))$$



### Off-Policy TD Control (Q-learning)

$$Q(s, a) \leftarrow Q(s, a) + \eta \left( r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right)$$

- The max over  $a$  rather than taking the  $a$  based on the current policy is the principle difference between Q-learning and SARSA.
- The learned action-value function  $Q$  directly approximates  $Q^*$  independent of the policy being followed
- Converges with
  - ✓ All state-action pairs are visited an infinite number of times
  - ✓ The policy converges in the limit to the greedy policy (i.e.,  $\epsilon$ -greedy with  $\epsilon = 1/t$ )

## Q-Learning: Off-Policy TD Control

### Q learning

Initialize  $Q(s, a)$  arbitrarily

Repeat (for each episode):

    Initialize  $s$

    Repeat (for each time step of episode):

        Choose  $a$  from  $s$  using policy derived from  $Q$  (e.g.,  $\epsilon$  – greedy) **Behavioral policy**

        Take action  $a$ , observe  $r, s'$

$$Q(s, a) \leftarrow Q(s, a) + \eta \left( r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right)$$

$s \leftarrow s'$

    Until  $s$  is terminal

**Estimation policy**

(Always try to estimate the optimal policy)

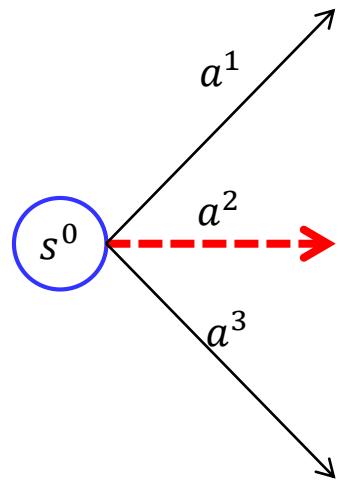
-Estimation can be greedy)

$a^* = \operatorname{argmax}_{a'} Q(s', a)$  is **not** used in the next state!!!

At the next state  $s'$ , Choose  $a$  using policy derived from  $Q$  (e.g.,  $\epsilon$  – greedy)

## Q-Learning: Off-Policy TD Control

$S_t$                    $A_t$                    $R_{t+1}$                    $S_{t+1}$                    $\max A_{t+1}$                    $S_{t+2}$

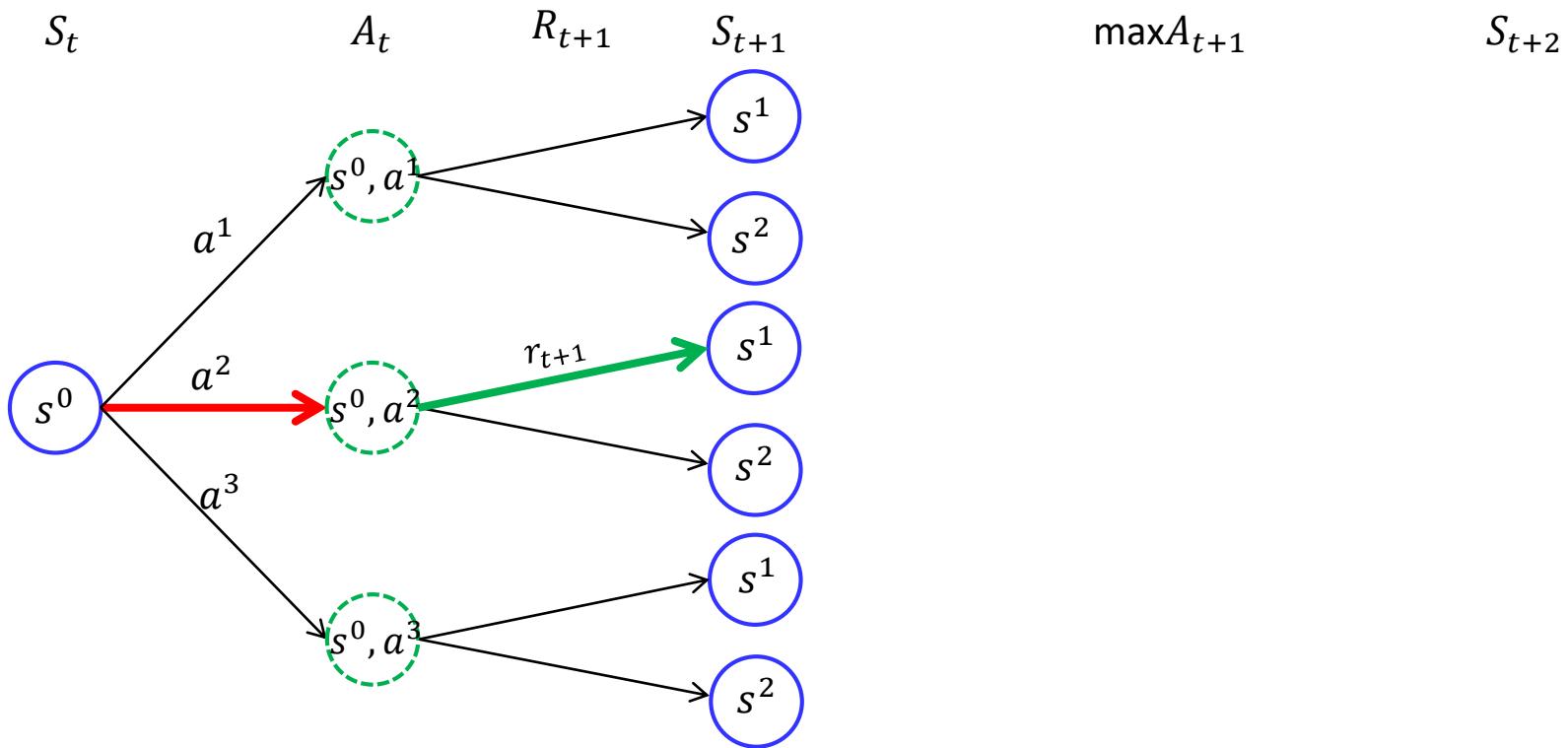


Choose  $a_t$  from  $s_t = s^0$  using current  $Q$

$$a_t = \begin{cases} \underset{a}{\operatorname{argmax}} Q(s_t = s^0, a) & \text{with prob } 1 - \epsilon \\ \text{random action} & \text{with prob } \epsilon \end{cases}$$

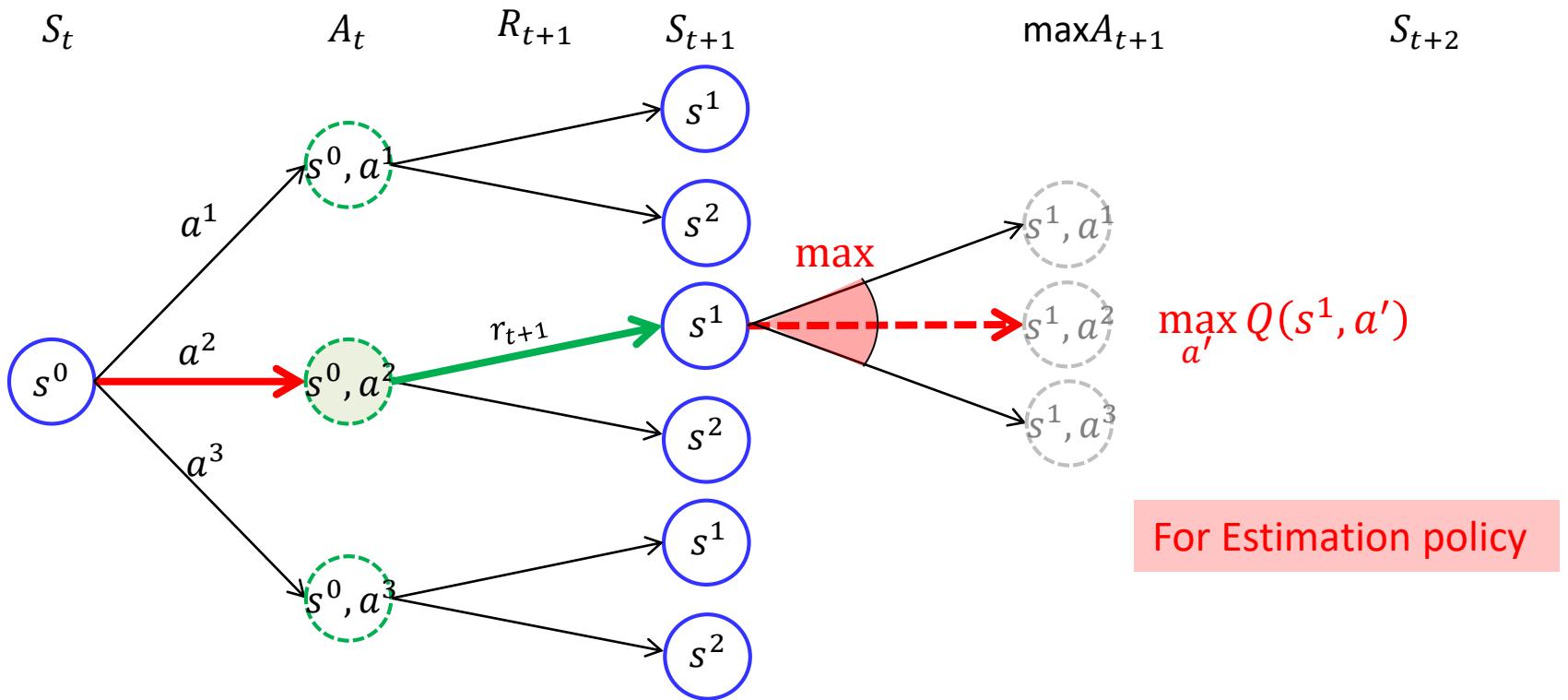
Assume  $a^2$  is chosen

## Q-Learning: Off-Policy TD Control



Take action  $a_t = a^2$  given  $s_t = s^0$  and observe  $r_{t+1}$  and  $s_{t+1} = s^1$

## Q-Learning: Off-Policy TD Control

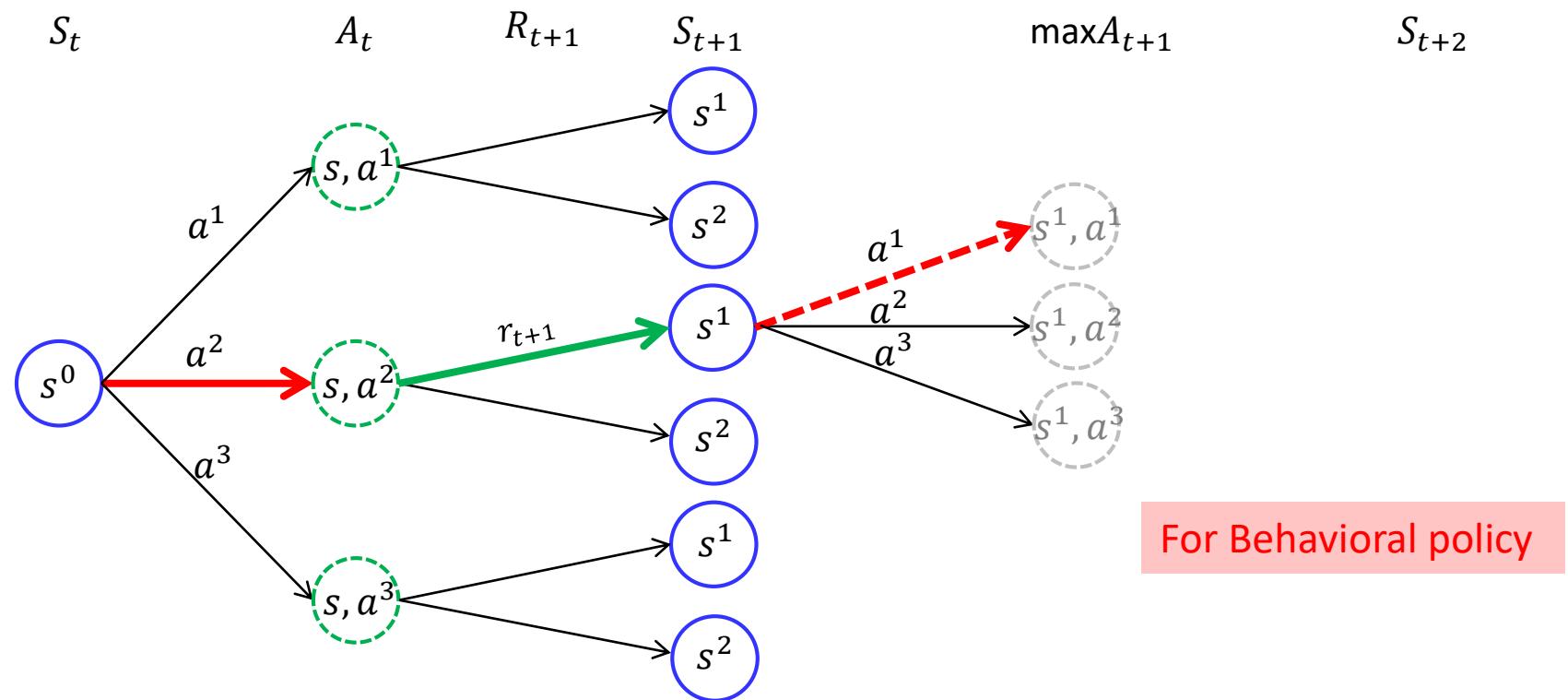


Update  $Q$  function with the  $\max_{a'} Q(s^1, a')$

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[ r_{t+1} + \gamma \max_{a'} Q(s^1, a') - Q(s_t, a_t) \right]$$

$$\rightarrow Q(s^0, a^2) \leftarrow Q(s^0, a^2) + \alpha \left[ r_{t+1} + \gamma \max_{a'} Q(s^1, a') - Q(s^0, a^2) \right]$$

## Q-Learning: Off-Policy TD Control

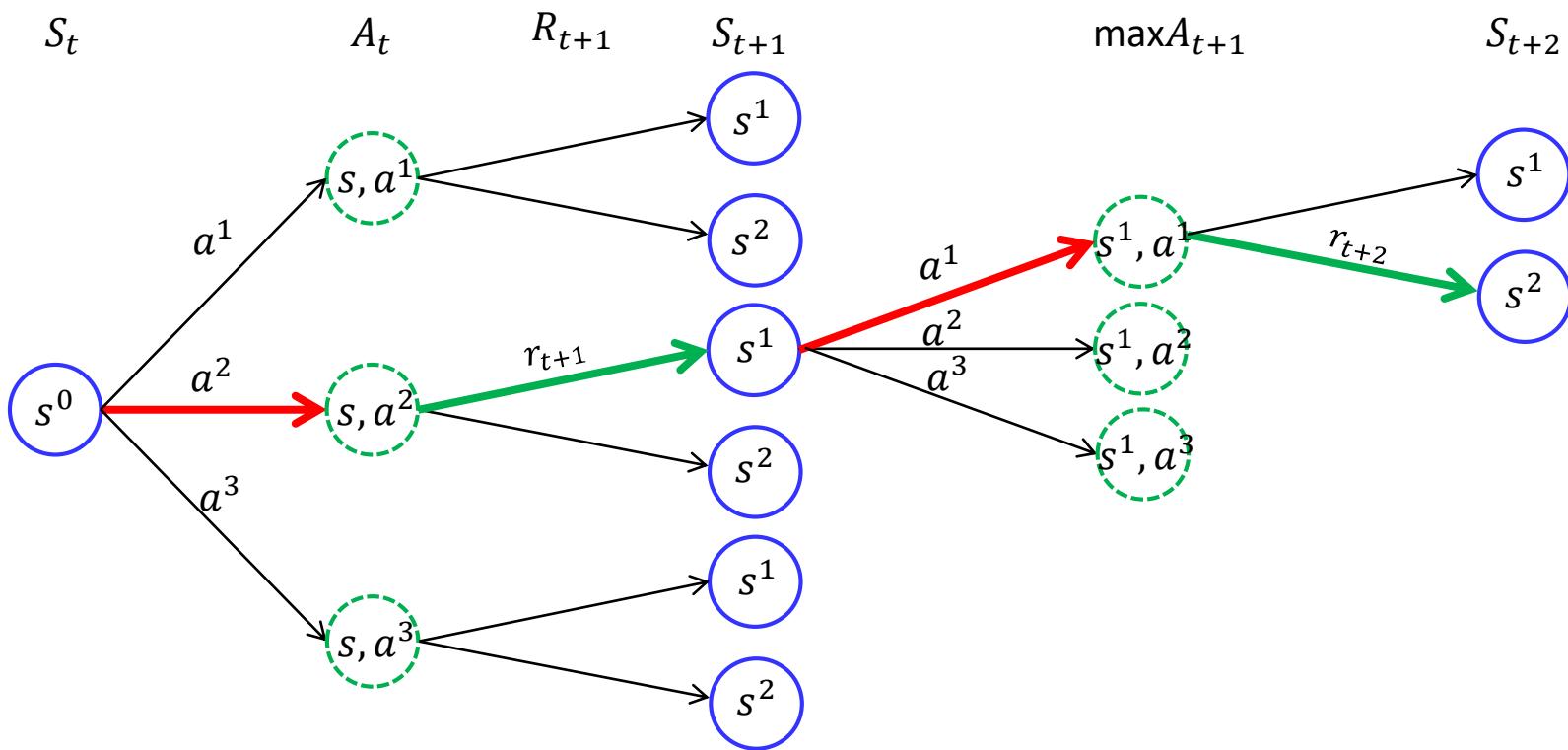


Choose  $a_{t+1}$  from  $s_{t+1} = s^1$  using current  $Q$

$$a_{t+1} = \begin{cases} \underset{a}{\operatorname{argmax}} Q(s_{t+1} = s^1, a) & \text{with prob } 1 - \epsilon \\ \text{random action} & \text{with prob } \epsilon \end{cases}$$

Assume  $a^1$  is chosen

## Q-Learning: Off-Policy TD Control



Take action  $a_{t+1} = a^1$  given  $s_{t+1} = s^1$  and observe  $r_{t+2}$  and  $s_{t+2} = s^2$