Recap

Today’s plan

First: Recap & Exam info

Time left: Q&A.

Exam: General tips

Focus on understanding the methods and the relationship between them rather than on remembering e.g. update equations

Especially important: know the advantages, disadvantages and limitations of each methods, and the situations where a certain method should be preferred.

The recap will try to sketch the coherence of all lecture topics, nevertheless, we cannot cover all 14 lectures in 90 minutes, and topics outside of the recap can be on the exam, too.

Exam: Cheat sheet

Many algorithms have variants:
Q- and V- version
importance weights
etc

Cheatsheet has most important ones only... Will be on Canvas after lecture to familiarise

Update equations cheat sheet

- DP value iteration: $v_{k+1}(s) = \max_a \sum_{s',r} p(s',r|s,a)\left[r + \gamma v_k(s')\right]$
- DP policy evaluation: $v_{k+1}(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\left[r + \gamma v_k(s')\right]$
- Monte Carlo: $V(S_t) \leftarrow V(S_t) + \alpha\left[G_t - V(S_t)\right]$
- TD(0): $V(S_t) \leftarrow V(S_t) + \alpha\left[R_{t+1} + \gamma V(S_{t+1}) - V(S_t)\right]$
- SARSA: $Q(S_t,A_t) \leftarrow Q(S_t,A_t) + \alpha\left[R_{t+1} + \gamma Q(S_{t+1},A_{t+1}) - Q(S_t,A_t)\right]$
- Expected SARSA: $Q(S_t,A_t) \leftarrow Q(S_t,A_t) + \alpha\left[R_{t+1} + \gamma \sum_a \pi(a|S_{t+1}) Q(S_{t+1},a) - Q(S_t,A_t)\right]$
- Q-learning: $Q(S_t,A_t) \leftarrow Q(S_t,A_t) + \alpha\left[R_{t+1} + \gamma \max_a Q(S_{t+1},a) - Q(S_t,A_t)\right]$
- n-step TD: $V_{t+n}(S_t) \doteq V_{t+n-1}(S_t) + \alpha\left[G_{t:t+n} - V_{t+n-1}(S_t)\right]$
- ~~Tree backups:~~ $\cancel{Q_{t+n}(S_t,A_t) \doteq Q_{t+n-1}(S_t,A_t) + \alpha\left[G^{\text{tree}}_{t:t+n} - Q_{t+n-1}(S_t,A_t)\right]}$ ~~, where~~
  $\cancel{G^{\text{tree}}_{t:t+n} \doteq R_{t+1} + \gamma \sum_{a \neq A_{t+1}} \pi(a|S_{t+1}) Q_{t+n-1}(S_{t+1},a) + \gamma\pi(A_{t+1}|S_{t+1}) G^{\text{tree}}_{t+1:t+n}}$
- Gradient Monte Carlo: $\mathbf{w} \leftarrow \mathbf{w} + \alpha\left[G_t - \hat{v}(S_t,\mathbf{w})\right]\nabla\hat{v}(S_t,\mathbf{w})$
- Semi-gradient TD: $\mathbf{w} \leftarrow \mathbf{w} + \alpha\left[R + \gamma\hat{v}(S',\mathbf{w}) - \hat{v}(S,\mathbf{w})\right]\nabla\hat{v}(S,\mathbf{w})$
- LSTD: $\mathbf{w}_t \doteq \widehat{\mathbf{A}}_t^{-1}\widehat{\mathbf{b}}_t$, where $\widehat{\mathbf{A}}_t \doteq \sum_{k=0}^{t-1} \mathbf{x}_k(\mathbf{x}_k - \gamma\mathbf{x}_{k+1})^\top + \varepsilon\mathbf{I}$ and $\widehat{\mathbf{b}}_t \doteq \sum_{k=0}^{t-1} R_{k+1}\mathbf{x}_k$
- GTD2: $\mathbf{v}_{t+1} \doteq \mathbf{v}_t + \beta\rho_t\left(\delta_t - \mathbf{v}_t^\top\mathbf{x}_t\right)\mathbf{x}_t, w_{t+1} = \mathbf{w}_t + \alpha\rho_t(\mathbf{x}_t - \gamma\mathbf{x}_{t+1})\mathbf{x}_t^\top\mathbf{v}_t$
  (note: in the GTD2 equation, v is a parameter vector, not the value function)

Note: the following methods are given assuming the discount factor $\gamma = 1$.

- Finite difference gradients: $\theta_{k+1} = \theta_k + \alpha\frac{J(\theta_k-\epsilon)-J(\theta_k+\epsilon)}{2\epsilon}$
- REINFORCE: $\theta_{k+1} = \theta_k + \alpha G(\tau)\sum_{t=0}^{T}\nabla_\theta \log\pi_\theta(a_t|s_t)$
- G(PO)MDP: $\theta_{k+1} = \theta_k + \alpha\sum_{t=0}^{T} r_t \sum_{t'=0}^{t}\nabla_\theta \log\pi_\theta(a_t|s_t)$
- PGT Actor-Critic: $\theta_{t+1} = \theta_t + \alpha\hat{q}(s_t,a_t)\nabla_\theta\log\pi_\theta(a_t|s_t)$
- Deterministic policy gradient (DPG): $\theta_{t+1} = \theta_t + \alpha\nabla_\theta\pi_\theta(a_t|s_t)\nabla_a q(s_t,a_t)|_{a=\pi_\theta(s)}$
- Natural policy gradient: $\theta_{t+1} = \theta_t + \alpha F^{-1}(\theta)\nabla_\theta J(\theta)$
  where $\nabla_\theta J(\theta)$ is an estimate of the 'vanilla' policy gradient.

Big picture

Finding optimal policies

Known MDP dynamics

$$p\left(s', r | s, a\right)$$

Dynamic programming

Only **data** from MDP: Reinforcement learning

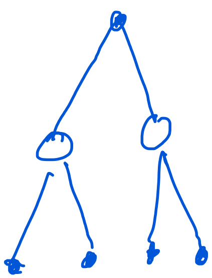
$$D = \left\{\left(\mathbf{s}_i, \mathbf{a}_i, r_i, \mathbf{s}'_i\right)\right\}_{i=1\dots N}$$

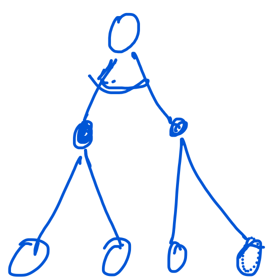
Dynamic programming

Planning Method.

Value Iteration

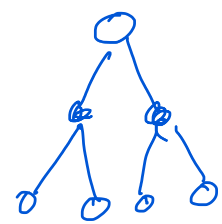
$$V^*_t(s) = \max_a E_{s'}\left[r(s,a) + \gamma V^*_{t+1}(s')\right]$$





Policy Iteration

1) Policy Evaluation

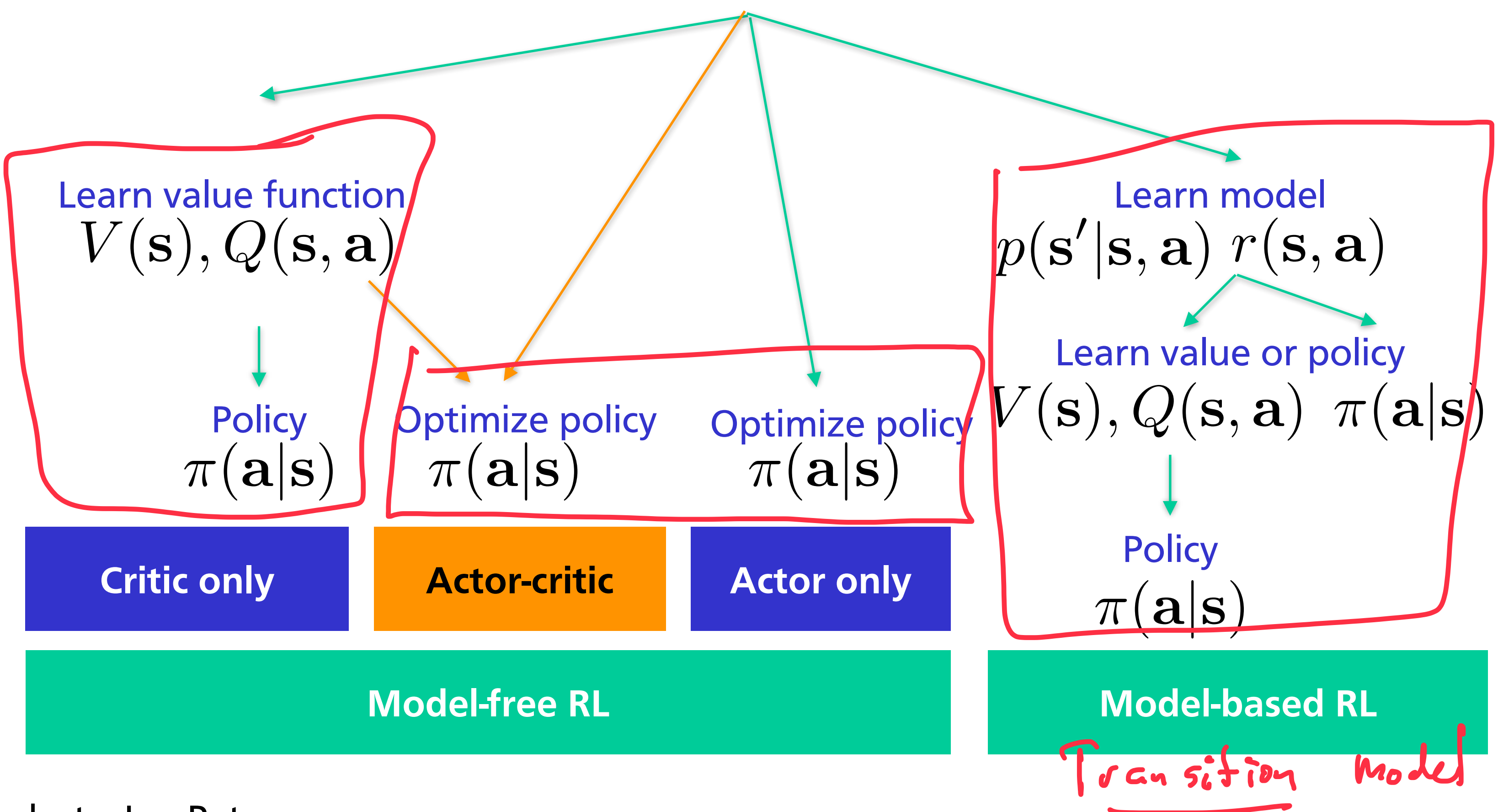


untill convergen V_π

2) $\pi_{new}(s) = \arg\max_a E_{s'} \, r(s,a) + \gamma V_\pi(s')$

Big picture: How to learn policies

$$D = \{(\mathbf{s}_i, \mathbf{a}_i, r_i, \mathbf{s}'_i)\}_{i=1\ldots N}$$



Thanks to Jan Peters

Value-based methods: MC

Lectures 2, 3 & 5

(Critic-only)
Value fc methods

TD

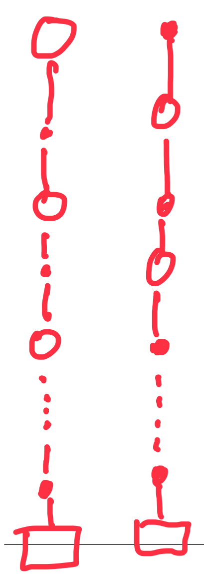
L 2,3

MC

tabular MC

gradient MC

L 5



$$V_\pi(S_t) \leftarrow V_\pi(S_t) + \alpha\left(\underbrace{G_t}_{\text{'target'}} - V_\pi(S_t)\right)$$

+ imp weights.

Mean-squared value error

$$\sum_s \mu(s)\left(V_\pi(s) - \hat{V}_w(s)\right)^2$$

$$w \leftarrow w + \alpha\left(G_t - \hat{V}_w(s)\right) \cdot \nabla_w \hat{V}_w(s)$$

Value based methods: TD learning

Lectures 3, 4, 5 & 6

TD-Learning

on-policy

off-policy

$V(S_t) \leftarrow V(S_t) + \alpha\,(\text{target} - V(S_t))$

$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\,(\text{target} - Q(S_t, A_t))$

TD(0), SARSA

target $= R_{t+1} + \gamma V(S_{t+1})$; $R_{t+1} + \gamma Q(S_{t+1}, A_{t+1})$

expected SARSA

Q-learning

target

$R_{t+1} + \gamma \sum_a \pi(a \mid S_{t+1}) \cdot Q(S_{t+1}, a)$

$R_{t+1} + \gamma \max_a Q(S_{t+1}, a)$

n-step TD, n-step SARSA

$R_{t+1} + \ldots + \gamma^{n-1} R_{t+n} + \gamma^n V(S_{t+n})$

$+ \gamma^n Q(S_{t+n}, A_{t+n})$

(+imp. wts)

Semi-gradient TD(0)

$w \leftarrow w + \alpha(R + \gamma \hat{V}_w(S') - \hat{V}_w(S)) \cdot \nabla_w \hat{V}_w(S)$

GTD2

gradient of the ms projected BE

DQN

- semi-gradient Q-learning
- experience replay, target networks.

double Q learning

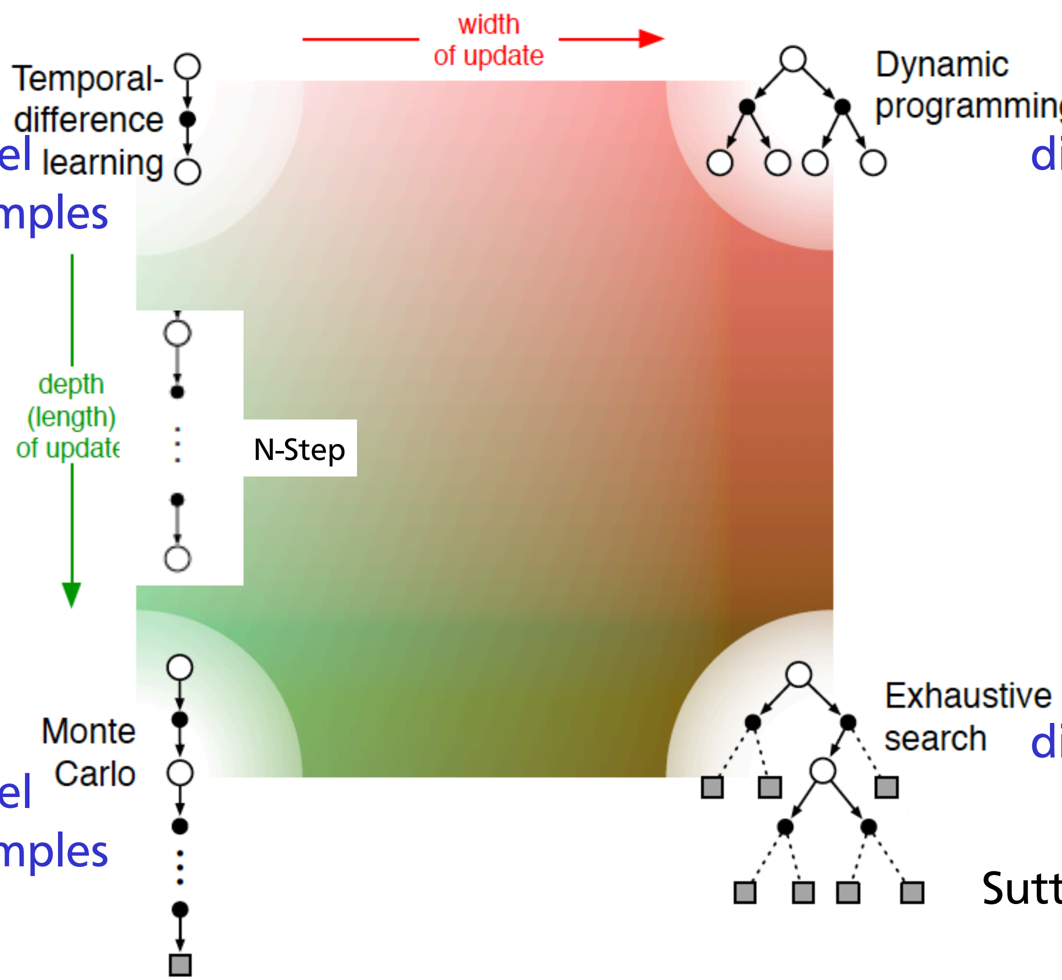
Off-policy learning

Lecture 2 (& throughout)

On-policy	Off-policy
Simpler	More complex
Specific case	More general
Often converges faster	Often large variance or slow convergence
Only data gathered with current policy	Can reuse data, use data from other source
Generally needs non-greedy policy	Allows greedy target policy

Evaluation methods

Lectures 2,3,4,5,6



distribution model
or real/simulated samples

distribution model

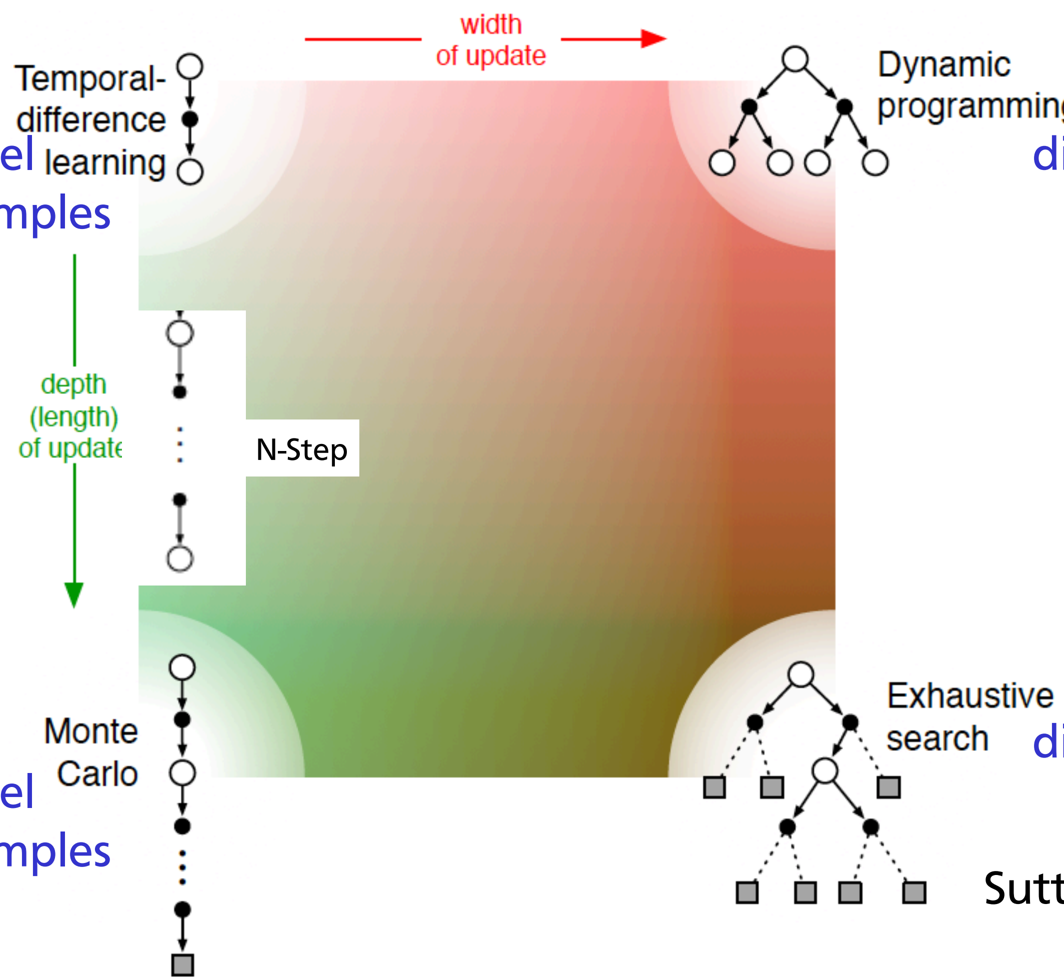
distribution model
or real/simulated samples

distribution model

Figure from
Sutton and Barto RL:AI

Evaluation methods

Lectures 2,3,4,5,6



distribution model
or real/simulated samples

distribution model

gradient
Monte-Carlo

distribution model
or real/simulated samples

distribution model

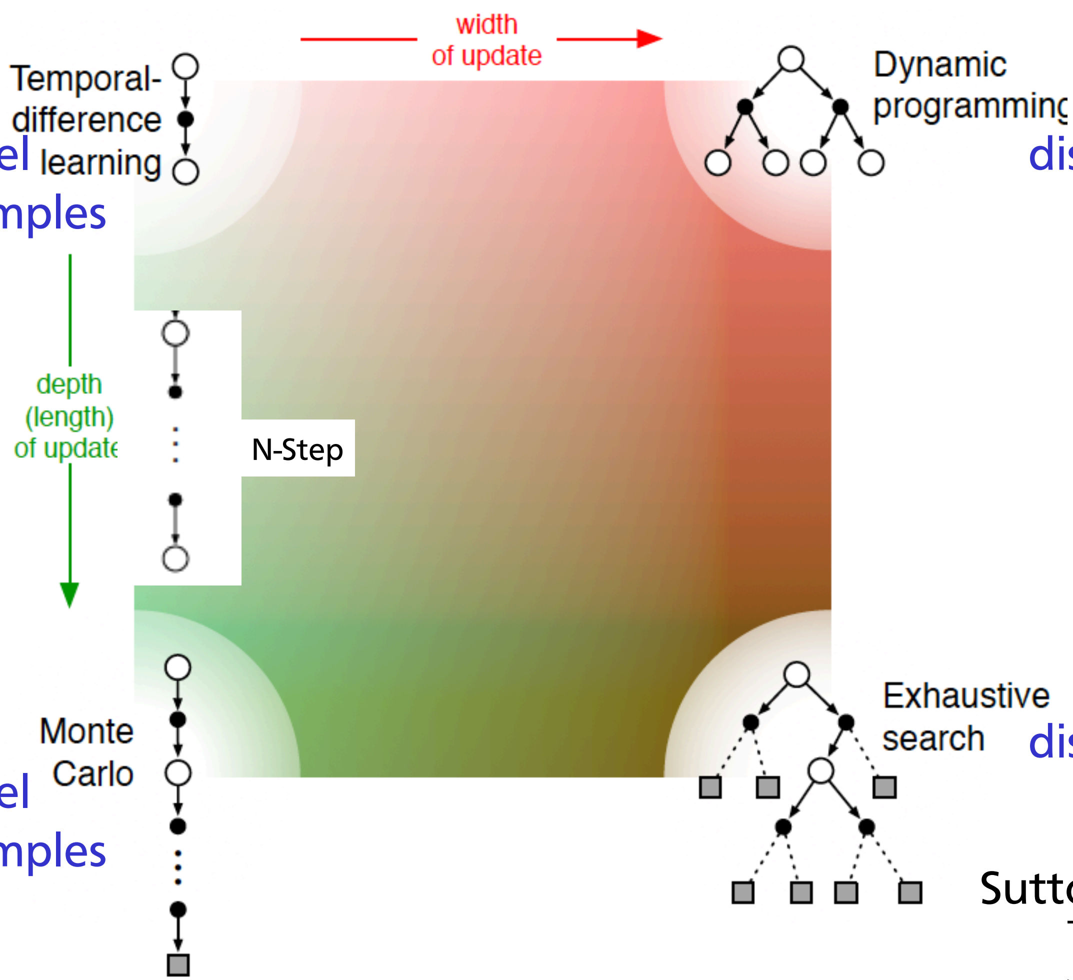
Figure from
Sutton and Barto RL:AI

Evaluation methods

Lectures 2,3,4,5,6

GTD2, semi-gradient TD(0) & LSTD

distribution model or real/simulated samples



distribution model

gradient Monte-Carlo

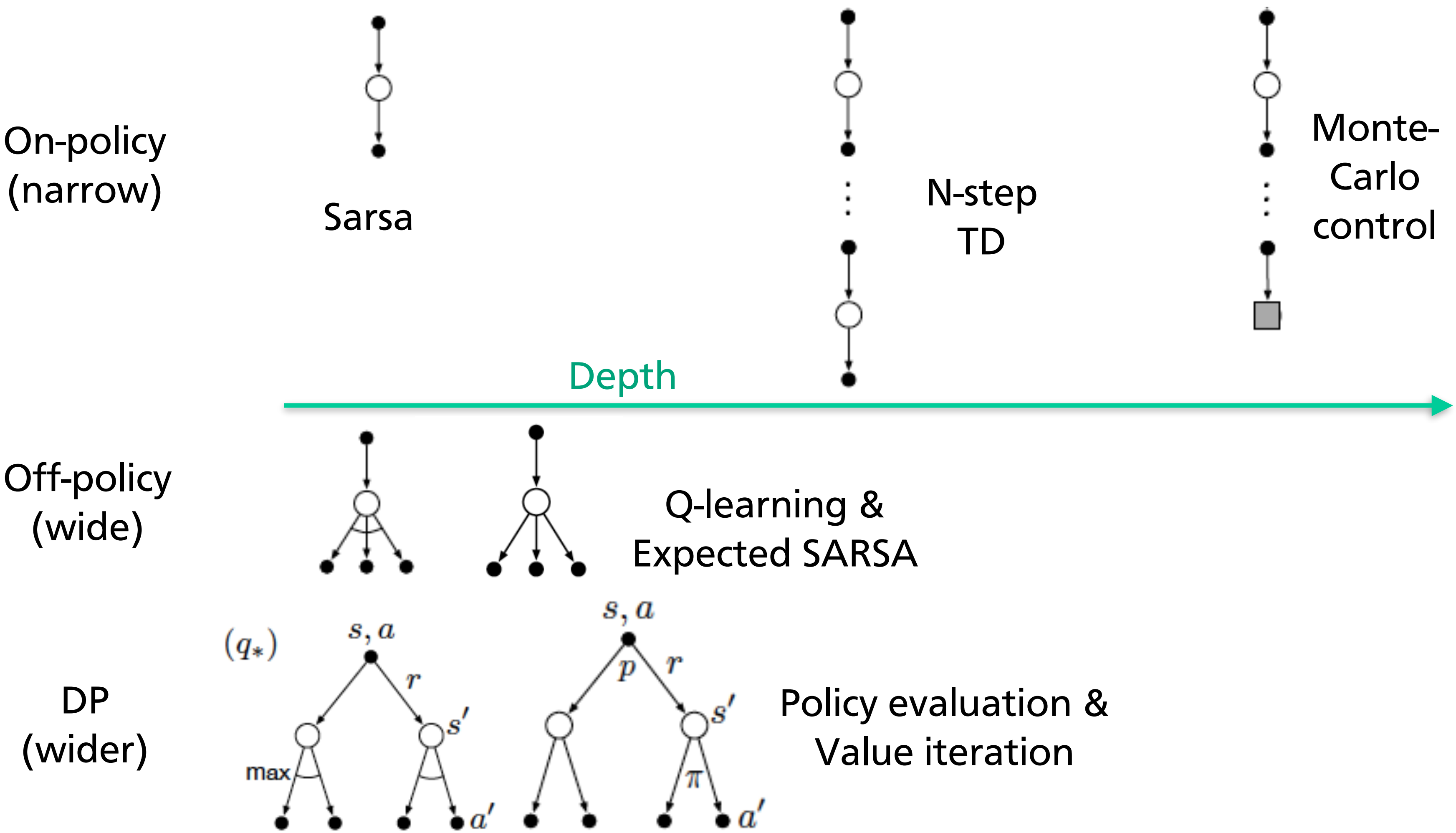
distribution model or real/simulated samples

distribution model

Figure from Sutton and Barto RL:AI

Control methods

Lectures 2, 3, 4



Types of function approximation

For any of the methods (gradient MC / semi-gradient TD/ LSTD / GTD2), choice of function approximation

linear	non-linear
tabular aggregate tiling radial basis function polynomial basis function fourier basis function	e.g. neural network

Convergence with function approximation

Lectures 5&6

	Tabular On/Off	Linear on **	Nonlinear on	Linear off **	Nonlinear off
Gradient MC *					
Semi-gradient TD *			No C!	No C!	No C!
Gradient TD *					
LSTD			N.A.		N.A.

* with appropriate step-size schedule   ** if features independent, single solution

Convergence with function approximation

Lectures 5&6

	Tabular On/Off	Linear on **	Nonlinear on	Linear off **	Nonlinear off
Gradient MC *					
Semi-gradient TD *				No C!	No C!
Gradient TD *					
LSTD			N.A.		N.A.

Local or global convergence?

* with appropriate step-size schedule

** if features independent, single solution

Convergence with function approximation

Lectures 5&6

<table>
<tr><th></th><th colspan="2">Tabular On/Off</th><th colspan="2">Linear on **</th><th colspan="2">Nonlinear on</th><th colspan="2">Linear off **</th><th colspan="2">Nonlinear off</th></tr>
<tr><td>Gradient MC *</td><td rowspan="4">global</td><td></td><td rowspan="4">global</td><td></td><td rowspan="4">non-linear: local</td><td></td><td rowspan="4">global</td><td></td><td rowspan="4">non-linear: local</td><td></td></tr>
<tr><td>Semi-gradient TD *</td><td></td><td></td><td>No C!</td><td>No C!</td><td>No C!</td></tr>
<tr><td>Gradient TD *</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>LSTD</td><td></td><td></td><td>N.A.</td><td></td><td>N.A.</td></tr>
</table>

* with appropriate step-size schedule

** if features independent, single solution

Convergence with function approximation

Lectures 5&6

	Tabular On/Off	Linear on **	Nonlinear on	Linear off **	Nonlinear off
Gradient MC *					
Semi-gradient TD *				No C!	No C!
Gradient TD *					
LSTD			N.A.		N.A.

global

cal

non-linear: local

Convergence to minimum of which error?

* with appropriate step-size schedule

** if features independent, single solution

Convergence with function approximation

Lectures 5&6

	Tabular On/Off	Linear on **	Nonlinear on	Linear off **	Nonlinear off
Gradient MC *	MC:VE				
Semi-gradient TD *	TD: PBE		No C!	No C!	No C!
Gradient TD *	TD: PBE				
LSTD	TD: PBE		N.A.		N.A.

* with appropriate step-size schedule   ** if features independent, single solution

Convergence with function approximation

Lectures 5&6

	Tabular On/Off	Linear on **	Nonlinear on	Linear off **	Nonlinear off
	global	global	non-linear: local	global	non-linear: local
Gradient MC *			MC:VE		
Semi-gradient TD *			TD: PBE No C!	No C!	No C!
Gradient TD *			TD: PBE		
LSTD			TD: PBE N.A.		N.A.

* with appropriate step-size schedule ** if features independent, single solution

Semi-gradients?

Semi-gradient TD(0)

$$w \leftarrow w + \alpha \left(\underline{R + \hat{V}_w(s')} - \hat{V}_w(s) \right) \cdot \nabla_w \hat{V}_w(s)$$

not gradient of any thing.

looks like gradient
MS TD error.

converge to minimum TD error.

weird result, depency of value
on the past

we don't do this, typically.

TD fix point

MSPBE

gradient

GTD2

found by
LSTD.

Policy-based methods

Lectures 7, 8, 9, 10

Policy

Policy Gradient

Finite-difference

$\nabla J(\theta) = \frac{J(\theta-\epsilon) - J(\theta+\epsilon)}{2\epsilon}$

DPG $\nabla J_\beta = \nabla_\theta Q(s, \pi_\theta(s))$

$= \nabla_a Q(s,a) \cdot \nabla_\theta \pi_\theta(s)$

log-ratio

$\theta \leftarrow \theta + \alpha(\text{target} - \text{baseline}) \cdot \nabla_\theta \log \pi(a_t|s_t)$

NPG

↳ $\tilde{\nabla} J = F^{-1}(\theta) \cdot \nabla J(\theta)$

↳ 'p.g.'

NPG gives direction of maximal improv of J per unit KL

TRPO

KL constraint

step size set to exactly satisfy the KL

REINFORCE

target G_0

G(PO)MDP

G_t

PGT+AC

$q_w(s_t, a_t)$

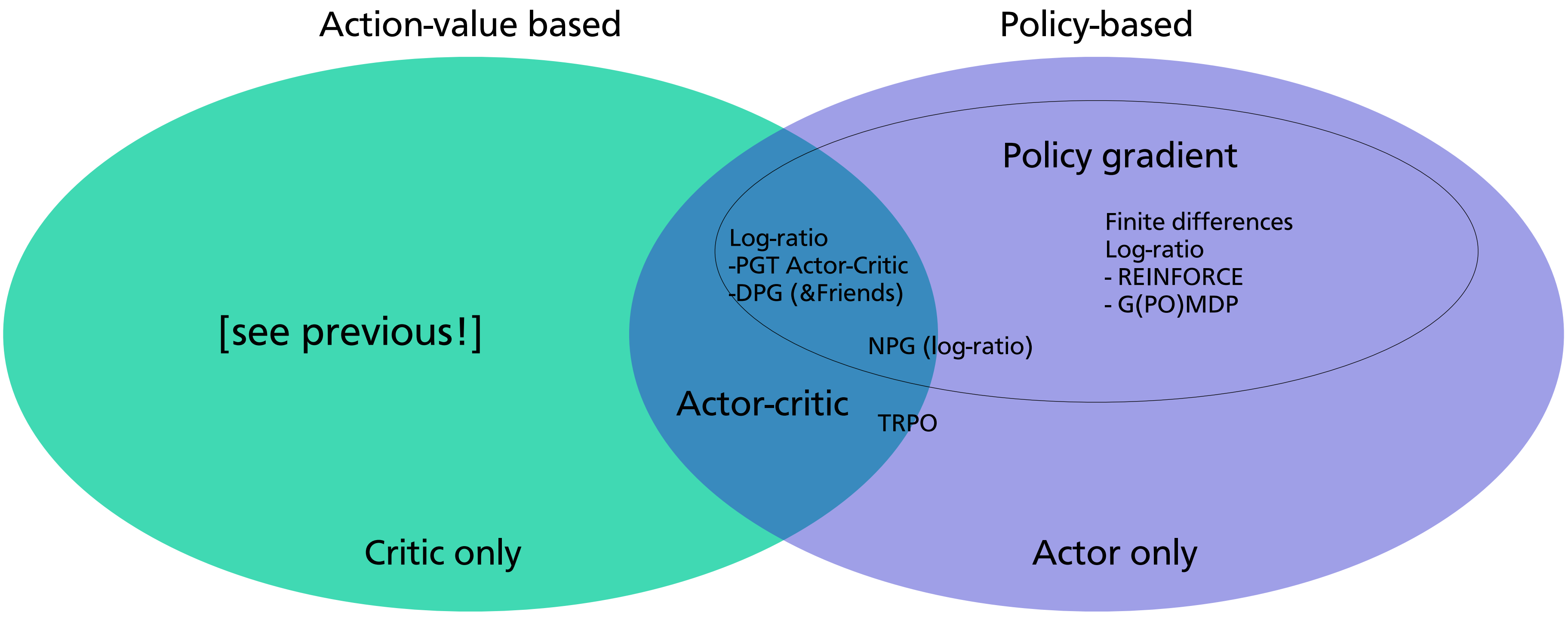
↳ Compatible f.a.

local optimum } unbiased.

$(GAE(\lambda, \gamma))$

Policies and action-values

Lectures 7, 8, 9, 10



Model-based learning

Lecture 11

Transition-model.

- Types of models (generative, trajectory, distributional)
- Dyna-Q
    - Prioritized sweeping, on-policy & uniform sampling
- Planning a full policy or from current state only (See lecture 11)
- Backpropagation through the model

State update functions in POMDPs

Lecture 12

Why do we need update functions for internal states?

What are the properties of int. states (compactness, markovian)

Exact methods

- **Full history**
  Not compact...
- **Belief state**
  Easy to interpret (Comp. heavy)
  Requires known model
- **Predictive state**
  Model learnable from data
  Most compact

Approximate methods

- **Recent observation(s)**
  Easy
  Lose long-term dependencies
- **End-to-end learning**
  Quite general
  RNN learning can be tricky, requires much data...

Other topics that are important

Maximization bias (lecture 4)

Exploration vs exploitation (throughout, also lecture 1)

Pure exploration & best arm identification (lecture 13)

Anything else I missed?

Other tips

Very important - know when/why to use each method/strategy (advantages, disadvantages&limitations)

Good luck with your preparation!

Intentionally left blank