Pei (https://blogs.cuit.columbia.edu/zp2130/)

Search

Search

+ Add post (https://blogs.cuit.columbia.edu/zp2130/wp-admin/post-new.php)

Menu

- Reinforcement Learning (https://blogs.cuit.columbia.edu/zp2130/)
- Posts (https://blogs.cuit.columbia.edu/zp2130/posts/)
- Resources (https://blogs.cuit.columbia.edu/zp2130/resources/)
- Cacti-based Framework (https://blogs.cuit.columbia.edu/zp2130/cacti/)
- Publications (https://blogs.cuit.columbia.edu/zp2130/publications/)

Email Address:

zp2130@caa.columbia.edu (mailto:zp2130@caa.columbia.edu)

p@caa.columbia.edu (mailto:p@caa.columbia.edu)

Reinforcement Learning is Direct Adaptive Optimal Control

Reinforcement Learning is Direct Adaptive Optimal Control (https://blogs.cuit.columbia.edu/zp2130/files/2019/03/w01-ReinforcementLearning.pdf)

Stanford_cs229-notes12_Andrew_Ng Reinforcement Learning and Control (http://blogs.cuit.columbia.edu/zp2130/files/2019/03/cs229-notes12.pdf)

How should **Reinforcement learning** be viewed from a **control** systems perspective?

Control problems can be divided into two classes:

1. **regulation** and **tracking** problems, in which the objective is to follow a **reference trajectory**.
2. **optimal** control problems, which the objective is to **extremize a functional** of the controlled system's behavior that not necessarily defined in terms of a reference trajectory.

Tracking vs Optimization

In some problems, the control objective is defined in terms of a reference level or **reference trajectory** that the controlled system's output should match or **track as closely as possible**. **Stability** is the **key issue** in these regulation and **tracking problems**.

In other problems, the control objective is to **extremize a functional** of the controlled system's behavior that is **not necessarily** defined in terms of a reference level or **trajectory**. The key issue in the latter problems is **constrained optimization**; here optimal-control methods based on the calculus of variations and dynamic programming have been extensively studied.

When a detailed and accurate model of the system to be controlled is not available, adaptive control methods can be applied. The overwhelming majority of **adaptive control** methods address regulation and **tracking** problems. However, **adaptive methods** for **optimal control** problems would be widely applicable if methods could be developed that were computationally feasible and that could be applied robustly to nonlinear systems.

For example, **trajectory planning** is a key and difficult problem in robot navigation tasks, as it is in other robot control tasks. To design a **robot** capable of **walking bipedally**, one may not be able to **specify a desired trajectory for the limbs** a priori, but one can specify the objective of moving forward, **maintaining equilibrium**, not damaging the robot, etc.

For both **tracking** and **optimal** control, it is usual to distinguish between **indirect** and **direct** adaptive control methods. An **indirect** method replies on a system identification procedure to form an explicit **model** of the controlled system and determines then the **control rule** from the **model**. **Direct** methods determine the **control rule without forming such a system model**.

Control problems	**Tracking problems**	**Optimal problems**
Trajectory	**Yes**	Not necessary
functional	No	**Extremize** a functional of the controlled system's behavior
Adaptive control methods explicit model	Indirect (system model)/Direct	Direct (no system model, Q-learning)/Indirect

Present **reinforcement learning** methods as a **direct** approach to **adaptive optimal control**.

Reinforcement Learning

Reinforcement learning is based on the common sense idea that if an action is followed by a satisfactory state of affairs, or by an improvement in the state of affairs (as determined in some clearly defined way), then the **tendency** to produce that action is strengthened, i.e., **reinforced**. This idea plays a fundamental role in theories of animal learning, in parameter-perturbation adaptive-control method, and in the theory of learning automata and bandit problems. Extending this idea to allow action selections to depend on state information introduces aspects of feedback control, pattern recognition and associative learning. Further, it is possible to extend the idea of being "followed by a satisfactory state of affairs" to include the long-term consequences of actions.

Conserved and Nonconserved Quantities

A conserved quantity cannot be created or destroyed and therefore has no sources or sinks; for conserved quantities such as atomic species i or internal energy U,

$$\frac{\partial c_i}{\partial t} = -\bigtriangledown \cdot \vec{J_i}$$
$$\frac{\partial u}{\partial t} = -\bigtriangledown \cdot \vec{J_u}$$
$$\text{where u: internal energy density.}$$

For nonconserved quantities such as entropy, S,

$$\frac{\partial s}{\partial t} = -\bigtriangledown \cdot \vec{J_s} + \dot{\sigma}$$

where

$\dot{\sigma}$: the rate of entropy production per unit volume.

Flux

A flux of i, $\vec{J_i}(\vec{r})$ describes the rate at which i flows through a unit area fixed with respect to a specified coordinate system.

Gradient

$$\bigtriangledown f = \frac{\partial f}{\partial x}\vec{i} + \frac{\partial f}{\partial y}\vec{j} + \frac{\partial f}{\partial z}\vec{k}$$

Divergence

$$\bigtriangledown \cdot f = \frac{\partial f}{\partial x} + \frac{\partial f}{\partial y} + \frac{\partial f}{\partial z}$$

Studies of reinforcement-learning neural networks in nonlinear control problems have generally focused on one of two main types of algorithm: actor–critic learning or Q-learning (they are direct adaptive optimal control algorithms). An actor–critic learning system contains two distinct subsystems, one (value) to estimate the long-term utility for each state and another (policy) to learn to choose the optimal action in each state. A

Q-learning system maintains estimates of utilities for all state-action pairs and makes use of these estimates to select actions.

Q-Learning

The name “Q-learning” comes purely form Watkins’ notation.

Watkins-Dayan1992_Article_Q-learning (http://blogs.cuit.columbia.edu/zp2130/files/2019/03/Watkins-Dayan1992_Article_Q-learning.pdf)

The objective is to find a control rule that maximizes at each time step the expected discounted sum of future reward. That is , at any time step ***k***, the control rule should specify action ***a_k*** so as to maximize

$$E\left\{\sum_{j=0}^{\infty}\gamma^j r_{k+j}\right\}$$

where γ , ***0 ≤ γ < 1*** , is a discount factor.

Reinforcement learning methods such as Q-learning do not estimate a system model. The basic idea in Q-learning is to estimate a ***real-valued function***, ***Q*** , of states and actions, where Q(x,a) is the expected discounted sum of future reward for performing action a in state x and performing optimally thereafter. This function satisfies the following recursive relationship (or “***functional equation***“):

$$Q(x,a) = E\{r_k + \gamma max_b Q(x_{k+1}, b) \mid x_k = x, a_k = a\}$$

The Q-learning procedure maintains an ***estimate*** $\hat{Q}$ of the function *Q*. At each transitions from step *k* to *k+1*, the learning system can observe x_k , a_k, r_k, *and* x_{k+1}. Based on these observations, $\hat{Q}$ is updated at time step ***k+1*** as follows: $\hat{Q}(x,a)$ remains unchanged for all pairs (x, a) ≠ (x_k, a_k)

$$\hat{Q}(x_k,a_k) := \hat{Q}(x_k,a_k) + \beta_k\left[r_k + \gamma max_b\hat{Q}\left(x_{k+1},b\right) - \hat{Q}\left(x_k,a_k\right)\right] \quad (1)$$

where

β_k : gain sequence (learning rate) such that

$0 < \beta_k < 1, \sum_{k=1}^{\infty}\beta_k = \infty$ and $\sum_{k=1}^{\infty}\beta_k^2 < \infty$

Watkins has shown that $\hat{Q}$ converges to ***Q*** with probability one if all actions continue to be tried from all states.

Because it **does not** reply on an explicit model of the Markov Process, Q-learning is a **direct** adaptive.

The Q function, combines information about state transitions and estimates of future reward **without** replying on explicit estimates of state transition probabilities.

Representing Q Function

Like conventional DP methods, the Q-learning method given by (1) requires memory and overall computation proportional to the number of **state–action pairs**. In large problems, or in problems with continuous **state and action spaces** which must be quantized, these methods becomes **extremely complex** (**Bellman’s “curse of dimensionality”**). One approach to reducing the severity of this problem is to represent $\hat{Q}$ not as a look-up table, but as a parameterized structure such as a low-order polynomial, k-d tree, decision tree, or neural network. In general , the local update rule for $\hat{Q}$ given by (1) can be adapted for use with any method for adjusting parameters of function representations via *supervised learning* methods. One can define a general way of moving from a unit of experience (x_k , a_k, r_k, *and* x_{k+1} as in (1)) to a training example for $\hat{Q}$:

$$\hat{Q}(x_k,a_k)\ should\ be\ r_k + \gamma max_b\hat{Q}\left(x_{k+1},b\right)$$

This training example can then be input to any supervised learning method, such as parameter estimation procedure based on stochastic approximation. The choice of learning method will have a strong effect on generlizatoin, the speed of learning and the quality of the final result. This approach has been used successfully with supervised learning methods based on error backpropagation, CMACs, and nearest-neighbor methods.

Reference about Optimal Control

1. Q-learning and Pontryagin’s Minimum Principle (https://blogs.cuit.columbia.edu/zp2130/files/2019/03/Q-learning-and-Pontryagin’s-Minimum-Principle.pdf)
2. Lecture10_Pontryagin’s Minimum Principle (https://blogs.cuit.columbia.edu/zp2130/files/2019/03/Lecture10_Pontryagins-Minimum-Principle.pdf)
3. Reinforcement Learning and Optimal Control Methods for Uncertain Nonlinear Systems (https://blogs.cuit.columbia.edu/zp2130/files/2019/03/Reinforcement-Learning-and-Optimal-Control-Methods-for-Uncertain-Nonlinear-Systems-1.pdf)
4. RL_and_Optimal_Control_A_Selective_Overview_Slide (https://blogs.cuit.columbia.edu/zp2130/files/2019/03/RL_and_Optimal_Control_A_Selective_Overview_Slide.pdf)
5. Q-learning_for_Optimal_Control_of_Continuous-time_systems (https://blogs.cuit.columbia.edu/zp2130/files/2019/03/Q-learning_for_Optimal_Control_of_Continuous-time_systems.pdf)
6. Slides_New Developments in Integral Reinforcement Learning Continuous-time Optimal Control and Games (https://blogs.cuit.columbia.edu/zp2130/files/2019/03/Slides_New-Developments-in-Integral-Reinforcement-Learning-Continuous-time-Optimal-Control-and-Games.pdf)

7. On_Stochasitc_Optimal_Control_and_Reinforcement_Learning_by_Approximate_Inference (https://blogs.cuit.columbia.edu/zp2130/files/2019/03/On_Stochasitc_Optimal_Control_and_Reinforcement_Learning_by_Approximate_Inference.pdf)

edit (https://blogs.cuit.columbia.edu/zp2130/wp-admin/post.php?post=4886&action=edit)

Author: Z Pei (https://blogs.cuit.columbia.edu/zp2130/author/zp2130/) on March 1, 2019

Categories: AI (https://blogs.cuit.columbia.edu/zp2130/category/ai/), Control (https://blogs.cuit.columbia.edu/zp2130/category/control/), Optimal Control (https://blogs.cuit.columbia.edu/zp2130/tag/optimal-control/), Reinforcement Learning (https://blogs.cuit.columbia.edu/zp2130/category/reinforcement-learning/), RL (https://blogs.cuit.columbia.edu/zp2130/category/rl/)

Tags: AI (https://blogs.cuit.columbia.edu/zp2130/tag/ai/), Control (https://blogs.cuit.columbia.edu/zp2130/tag/control/), Optimal Control (https://blogs.cuit.columbia.edu/zp2130/tag/optimal-control/), Reinforcement Learning (https://blogs.cuit.columbia.edu/zp2130/tag/reinforcement-learning/), RL (https://blogs.cuit.columbia.edu/zp2130/tag/rl/)

Other posts

Decentralized Stabilization for a Class of Continuous-Time Nonlinear Interconnected Systems Using Online Learning Optimal Control Approach (https://blogs.cuit.columbia.edu/zp2130/decentralized_stabilization_for_a_class_of_continuous-time_nonlinear_interconnected_systems_using_online_learning_optimal_control_approach/) «» Neural-network-based decentralized control of continuous-time nonlinear interconnected systems with unknown dynamics (https://blogs.cuit.columbia.edu/zp2130/neural-network-based_decentralized_control_of_continuous-time_nonlinear_interconnected_systems_with_unknown_dynamics/)

Last posts

- Symbolic Netlist to Innovus-friendly Netlist (https://blogs.cuit.columbia.edu/zp2130/symbolic_netlist_to_innovus-friendly_netlist/)
- Finite-Sample Convergence Rates for Q-Learning and Indirect Algorithms (https://blogs.cuit.columbia.edu/zp2130/finite-sample_convergence_rates_for_q-learning_and_indirect_algorithms/)
- Solving H-horizon, Stationary Markov Decision Problems In Time Proportional To Log(H) (https://blogs.cuit.columbia.edu/zp2130/paul_tseng_1990/)
- Randomized Linear Programming Solves the Discounted Markov Decision Problem In Nearly-Linear (Sometimes Sublinear) Run Time (https://blogs.cuit.columbia.edu/zp2130/randomized_linear_programming_solves_the_discounted_markov_decision_problem_in_nearly-linear_sometimes_sublinear_run_time/)
- KL Divergence (https://blogs.cuit.columbia.edu/zp2130/kl_divergence/)
- The Asymptotic Convergence-Rate of Q-learning (https://blogs.cuit.columbia.edu/zp2130/the_asymptotic_convergence-rate_of_q-learning/)
- Hierarchical Apprenticeship Learning, with Application to Quadruped Locomotion (https://blogs.cuit.columbia.edu/zp2130/hierarchical_apprenticeship_learning_with_application_to_quadruped_locomotion/)
- Policy Gradient Methods (https://blogs.cuit.columbia.edu/zp2130/policy_gradient_methods/)
- Actor-Critic Algorithms for Hierarchical Markov Decision Processes (https://blogs.cuit.columbia.edu/zp2130/actor-critic_algorithms_for_hierarchical_markov_decision_processes/)
- Hierarchical Deep Reinforcement Learning: Integrating Temporal Abstraction and Intrinsic Motivation (https://blogs.cuit.columbia.edu/zp2130/hierarchical_deep_reinforcement_learning_integrating_temporal_abstraction_and_intrinsic_motivation/)