6

Optimising actions for control objectives

Concept. To design and build software which enables the optimisation of automated control objectives over stochastic phenomena of any kind. The theory in this chapter will overlap significantly with that of Reinforcement Learning (RL), however, in contrast to more standard RL approaches, we shall be relying on all of the work from previous parts of this book to help agents characterise, measure and learn from their environment. In particular, the online learning of stochastic simulation state and parameters will be crucial to this model-based approach. The software which implements our generalised control optimisation algorithm will be implemented as an extension to the learnadex. For the mathematically-inclined, this chapter will cover how we formalise model-based automated control optimisation within the frameworks that we have already introduced in this book. For the programmers, the public Git repository for the code described in this chapter can be found here: https://github.com/umbralcalc/learnadex.

6.1 States, actions and attributing rewards

In the previous parts of this book we laid out the concept for a generalised framework to simulate and learn stochastic phenomena continually as data is received. Given that we have also introduced a framework for the automated control of these phenomena, we have all the ingredients we need to create optimal decision-making algorithms. The key question to answer then, is: *optimal with respect to what objective?*

The objective of an automated control algorithm could take many forms depending on the specific context. Since there is no loss in generality in doing so, it seems natural to follow the naming convention used by Markov Decision Processes (MDP) [1, 2] by referring to the objective outcome of an action at a particular point in time as having a 'reward' value r. Since the relationship between reward, actions and states may be stochastic, it makes sense to relate the reward outcome r given a state history X and action history A at timestep $\mathtt{t}+1$ through the probability distribution

$P_{\mathsf{t}+1}(r|X,A)$. Hence, generally, this reward signal is non-Markovian — as is the case in many real-world problems [3].

We can use the reward probability distribution to derive a joint distribution over both state history X' and reward r at timestep $\mathsf{t}+1$ like so

$$P_{(\mathsf{t}+1)\mathsf{t}}(r,x'|X,z,\theta) = P_{\mathsf{t}+1}(r|X',A)\Pi_{(\mathsf{t}+1)\mathsf{t}}(A|X,\theta)P_{(\mathsf{t}+1)\mathsf{t}}(x'|X,z,A)\,. \tag{6.1}$$

In this expression, let's recall that we are using the policy distribution $\Pi_{(\mathsf{t}+1)\mathsf{t}}(A|X,\theta)$ for agent interactions and the fundamental state update conditional probability for the underlying stochastic process $P_{(\mathsf{t}+1)\mathsf{t}}(x'|X,z,A)$.

Note that in most use cases, the state of real-world phenomena cannot be measured perfectly. So to enable any agent trained on simulated phenomena to potentially act in the real world, we will need to include a measurement process as part of the information retrieval step. This is the part where we can leverage our work in a previous chapter which develops an online learning system for stochastic process models. But we're jumping ahead with this thinking and will return to this point later on.

Using Eq. (6.1), we can now define a 'state value function' V_{t} at timestep t which is the expected γ-discounted future reward given the current state history X and the other parameters like this[1]

$$\begin{aligned} V_{\mathsf{t}}(X,z,\theta) &= \mathrm{E}_{\mathsf{t}}(\mathsf{Discounted\ Return}|X,z,\theta) \\ &= \sum_{\mathsf{t}'=\mathsf{t}}^{\infty}\int_{\omega_{\mathsf{t}'+1}}\mathrm{d}^n x'\int_{\rho_{\mathsf{t}'+1}}\mathrm{d}r\, r\,\gamma^{\mathsf{t}'-\mathsf{t}}\prod_{\mathsf{t}''=\mathsf{t}}^{\mathsf{t}'}P_{(\mathsf{t}''+1)\mathsf{t}''}(r,x'|X,z,\theta)\,, \end{aligned} \tag{6.2}$$

where $0<\gamma<1$. The idea behind this discount factor γ is to decrease the contribution of rewards to the optimisation objective (often called the 'expected discounted return' in RL) more and more as the prediction increases into the future. Note also that the state value function is inherently recursively defined, such that

$$V_{\mathsf{t}}(X,z,\theta) = \int_{\omega_{\mathsf{t}+1}}\mathrm{d}^n x\int_{\rho_{\mathsf{t}+1}}\mathrm{d}r\,P_{(\mathsf{t}+1)\mathsf{t}}(r,x'|X,z,\theta)\Big\{r+\gamma V_{\mathsf{t}+1}(X',z,\theta)\Big\}\,, \tag{6.3}$$

and the optimal θ can hence be derived from

$$\theta^*_{\mathsf{t}}(X,z) = \underset{\theta}{\mathrm{argmax}}\big[V_{\mathsf{t}}(X,z,\theta)\big]\,. \tag{6.4}$$

By deriving the optimal policy in terms of the parameters $\theta^*_{\mathsf{t}}(X,z)$, the optimal state value function and policy distribution can therefore be derived from

$$V^*_{\mathsf{t}}(X,z) = V_{\mathsf{t}}[X,z,\theta^*_{\mathsf{t}}(X,z)] \tag{6.5}$$

$$\Pi^*_{(\mathsf{t}+1)\mathsf{t}}(A|X,z) = \Pi_{(\mathsf{t}+1)\mathsf{t}}[A|X,\theta^*_{\mathsf{t}}(X,z)]\,. \tag{6.6}$$

[1] The discount factor in continuous time could also be explicitly dependent on the stepsize such that we would replace the discount factor in Eq. (6.2) with

$$\gamma^{\mathsf{t}'-\mathsf{t}} \longrightarrow \frac{1}{\gamma[\delta t(\mathsf{t}+1)]}\prod_{\mathsf{t}''=\mathsf{t}}^{\mathsf{t}'}\gamma[\delta t(\mathsf{t}''+1)]\,.$$

Note that the type of decision process optimisation which we have introduced above differs from standard RL methodology. In the more conventional 'model-free' RL approaches, the state-action value function

$$Q_{\mathsf{t}}(X, A, z) = \mathrm{E}_{\mathsf{t}}(\mathsf{Discounted\ Return}|X, A, z)\,, \tag{6.7}$$

would be used to evaluate the optimal policy instead of the state value function $V_{\mathsf{t}}(X, z, \theta)$ that we are using above. We are able to use the latter here because the simulation model gives us explicit knowledge of the $P_{(\mathsf{t}+1)\mathsf{t}}(x'|X, z, A)$ distribution which is utilised by Eq. (6.1). When this model is not known, the state-action value function $Q_{\mathsf{t}}(X, A, z)$ must be learned explicitly through sample estimation from the measured state and experienced outcomes of actions taken by the agent.

When an agent takes an action to measure the state of the system (or when it is given measurements without needing to take action) there will typically be some uncertainty in how the history of measured real-world data Y maps to the latent states of the system X and its parameters z at time $\mathsf{t}+1$. It is natural, then, to represent this uncertainty with a posterior probability distribution $\mathcal{P}_{\mathsf{t}+1}(X, z|Y)$ as we did in the previous chapters of this book.

Follow-up this bit with the model-based approach that we're going to take in this book.

- Introduce broad concept of dynamic programming — partitioning a optimal global control into smaller optimal control segments/iterations.
- Talk about the utility of the model-based online learning approach in the case of partially observed systems [4].
- Look into the overlaps with this approach and Thompson sampling for exploration — discuss here.
- Looking at a stochastic policy iteration algorithm here combined with Monte Carlo rollouts.
- The value learning can be facilitated in software using a predictive model which is able to roll forecast rewards forward in time in a Monte Carlo fashion up to a window from a certain point given an input prior distribution of policies.
- This input prior distribution of policies can itself be optimised by maximising expected discounted utility in a Bayesian design framework. Draw parallels.

Bibliography

[1] D. P. Bertsekas et al., *Dynamic programming and optimal control 3rd edition, volume ii*, *Belmont, MA: Athena Scientific* **1** (2011) .

[2] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[3] M. Gaon and R. Brafman, *Reinforcement learning with non-markovian rewards*, in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, pp. 3980–3987, 2020.

[4] K. J. Åström, *Optimal control of markov processes with incomplete state information i*, *Journal of mathematical analysis and applications* **10** (1965) 174–205.