15. From policy search to policy gradients

In the previous lectures we attempted to reduce the complexity of planning by assuming that value functions over the large state-action spaces can be compactly represented with a few parameters. While value-functions are an indispensable component of poly-time MDP planners (see Lectures 3 and 4), it is far from clear whether they should also be given priority when working with larger MDPs.

Indeed, perhaps it is more natural to consider sets of policies with a compact description. Formally, in this problem setting the planner will be given a black-box simulation access to a (say, γ-discounted) MDP $M = (\mathcal{S}, \mathcal{A}, P, r)$ as before, but the interface also provides access to a parameterized family of policies over $(\mathcal{S}, \mathcal{A})$, $\pi = (\pi_\theta)_{\theta \in \mathbb{R}^d}$, where for any fixed parameter $\theta \in \mathbb{R}^d$, π_θ is a memoryless stochastic policy: $\pi_\theta : \mathcal{S} \to \mathcal{M}_1(\mathcal{A})$.

For example, π_θ could be such that for some feature-map $\phi : \mathcal{S} \times \mathcal{A} \to \mathcal{R}^d$,

$$\pi_\theta(a|s) = \frac{\exp(\theta^\top \varphi(s,a))}{\sum_{a'} \exp(\theta^\top \varphi(s,a'))}\,, \qquad (s,a) \in \mathcal{S} \times \mathcal{A}\,. \tag{1}$$

In this case "access" to π_θ means access to φ, which can be either global (i.e., the planner is given the "whole" of φ and can run any preprocessing on it), or local (i.e., $\varphi(s', a)$ is returned by the simulator for the "next states" $s' \in \mathcal{S}$ and for all actions a). Of course, the exponential function can be replaced with other functions, or, one can just use a neural network to output "scores", which are turned into probabilities in some way. Dispensing with stochastic policies, a narrower class is the class of policies that are greedy with respect to action-value functions that belong to some parametric class.

One special case that is worthy of attention due to its simplicity is the case when $\mathcal{S}$ is partitioned into m (disjoint) subsets $\mathcal{S}_1, \ldots, \mathcal{S}_m$ and for $i \in [m]$, we have A basis functions defined as follows:

$$\phi_{i,a'}(s,a) = \mathbb{I}(s \in \mathcal{S}_i, a = a')\,, \qquad s \in \mathcal{S}, a, a' \in \mathcal{A}, i \in [m]\,. \tag{2}$$

Here, to minimize clutter, we allow the basis functions to be indexed by pairs and identified $\mathcal{A}$ with $1, \ldots, \mathrm{A}$, as usual. Then, the policies are given by $\theta = (\theta_1, \ldots, \theta_m)$, the collection of m probability vectors $\theta_1, \ldots, \theta_m \in \mathcal{M}_1(\mathcal{A})$:

$$\pi_\theta(a|s) = \sum_{i=1}^{m} \sum_{a'} \phi_{i,a'} \theta_{i,a'} \,. \tag{3}$$

Note that because of the special choice of ϕ, $\pi_\theta(a|s) = \theta_{i,a}$ for the unique index $i \in [m]$ such that $s \in \mathcal{S}_i$. This is known as state-aggregretion: States belonging to the same group give rise to the same probability distribution over the actions. We say that the featuremap $\varphi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^d$ is of the **state-aggregation type** if it takes the form (2) with an appropriate reindexing of the basis functions.

Fix now a state-aggregation type featuremap. We can consider both the **direct parameterization** of policies given in (3), or the "Boltzmann" parameterization given in (1). As it is easy to see the set of possible policies that can be expressed with the two parameterizations are nearly identical. Letting Π_{direct} be the set of policies that can be expressed using φ and the direct parameterization and letting $\Pi_{\text{Boltzmann}}$ be the set of policies that can be expressed using φ but with the Boltzmann parameterization, first note that $\Pi_{\text{direct}}, \Pi_{\text{Boltzmann}} \subset \mathcal{M}_1(\mathcal{A})^{\mathcal{S}} \subset ([0,1]^{\mathrm{A}})^{\mathrm{S}}$, and if we take the closure, $\mathrm{clo}(\Pi_{\text{Boltzmann}})$ of $\Pi_{\text{Boltzmann}}$ then we can notice that

$$\mathrm{clo}(\Pi_{\text{Boltzmann}}) = \Pi_{\text{direct}} \,.$$

In particular, the Boltzmann policies cannot express point-mass distributions with finite parameters, but letting the parameter vectors grow without bound, any policy that can be expressed with the direct parameterization can also be expressed by the Boltzmann parameterization. There are many other possible parameterizations, as also mentioned earlier. The important point to notice is that while the parameterization is necessary so that the algorithms can work with a compressed representation, different representations may describe an identical set of policies.

Policy search

A reasonable goal then is to ask for a planner that competes with the best policy within the parameterized family, or the ε-best policy policy for some positive ε. Since there may not be a parameter θ such that $v^{\pi_\theta} \geq v^{\pi_{\theta'}} - \varepsilon \mathbf{1}$ for any $\theta' \in \mathbb{R}^d$, we simplify the problem by requiring that the policy computed is nearly best when started from some initial distribution $\mu \in \mathcal{M}_1(\mathcal{S})$.

Defining $J : \mathrm{ML} \to \mathbb{R}$ as

$$J(\pi) = \mu v^\pi (= \sum_{s \in \mathcal{S}} \mu(s) v^\pi(s)),$$

the **policy search problem** is to find a parameter $\theta \in \mathbb{R}^d$ such that

$$J(\pi_\theta) = \max_{\theta'} J(\pi_{\theta'}) \, .$$

The approximation version of the problem asks for finding $\theta' \in \mathbb{R}^d$ such that

$$J(\pi_\theta) = \max_{\theta'} J(\pi_{\theta'}) - \varepsilon \, .$$

The formal problem definition then is as follows: a planning algorithm is given the MDP M and a policy parameterization $(\pi_\theta)_\theta$ and we are asking for an algorithm that returns the solution to the policy search problem in time polynomial in the number of actions A and the number of parameters d that describes the policy. An even simpler problem is when the MDP has finitely many states, and the algorithm needs to run in polynomial time in S, A and d. In this case, it is clearly advantageous for the algorithm if it is given the exact description of the MDP (as described in Lecture 3) Sadly, even this mild version of policy search is intractable.

Theorem (Policy search hardness): Unless $\mathrm{P} = \mathrm{NP}$, there is no polynomial time algorithm for the finite policy search problem even when the policy space is restricted to the constant policies and the MDPs are restricted to be deterministic with binary rewards.

The constant policies are those that assign the same probability distribution to each state. This is a special case of state aggregation when all the states are aggregated into a single class. As the policy does not depend on the state, the problem is also known as the **blind policy search problem**. Note that the result holds regardless of the representation used to express the set of constant policies.

Proof: Let $\mathcal{S} = \mathcal{A} = [n]$. The dynamics is deterministic: The next state is a if action $a \in \mathcal{A}$ is taken in state n. A policy is simple a probability distribution $\pi \in \mathcal{M}_1([n])$ over the action space, which we shall view as a column vector taking values in $[0, 1]^n$. The

transition matrix of π is $P_\pi(s,s') = \pi(s')$, or, in matrix form, $P_\pi = \mathbf{1}\pi^\top$. Clearly, $P_\pi^2 = \mathbf{1}\pi^\top \mathbf{1}\pi^\top = P_\pi$ (i.e., P_π is idempotent). Thus, $P_\pi^t = \mathbf{1}\pi^\top$ for any $t > 0$ and hence

$$J(\pi) = \mu(r_\pi + \sum_{t \geq 1} \gamma^t P_\pi^t r_\pi) = \mu \left(I + \frac{\gamma}{1-\gamma} \mathbf{1}\pi^\top \right) r_\pi \,.$$

Defining $R_{s,a} = r_a(s)$ so that $R \in [0,1]^{n \times n}$, we have $r_\pi = R\pi$. Plugging this in into the previous displayed equation and using that $\mu \mathbf{1} = 1$, we get

$$J(\pi) = \mu R \pi + \frac{\gamma}{1-\gamma} \pi^\top R \pi \,.$$

Thus we see that the policy search problem is equivalent to maximizing the quadratic expression in the previous display over the probability simplex. Since there is no restriction on R, one may at this point conjecture that this will be hard to do. That this is indeed the case can be shown by a reduction to the **maximum independent set problem**, which asks for checking whether the independence number of a graph is above a threshold and which is known to be NP-hard even for 3-regular graphs (i.e., graphs where every vertex has exactly three neighbours).

Here, the independence number of a graph is defined as follows: We are given a simple graph $G = (V, E)$ (i.e., there are no self-loops, no double edges, and the graph is undirected). An independent set in G is a neighbour-free subset of vertices. The independence number of G is defined as

$$\alpha(G) = \max\{|V'| \,:\, V' \subset \text{ independent in } G\} \,.$$

Quadratic optimization has close ties to the maximum independent set problem:

Lemma (Motzkin-Strauss '65): Let $G \in \{0,1\}^n$ be the vertex-vertex adjacency matrix of simple graph (i.e., $G_{ij} = 1$ if and only if (i,j) is an edge of the graph). Then, for $I \in \{0,1\}^{n \times n}$ the $n \times n$ identity matrix,

$$\frac{1}{\alpha(G)} = \min_{y \in \mathcal{M}_1([n])} y^\top (G + I) y \,.$$

We now show that if there is an algorithm that solves policy search in polynomial time then it can also be used to solve the maximum independent set problem for simple, 3-regular graphs. For this pick a 3-regular graph G with n vertices. Define the MDP as above with n states and actions and the rewards chosen to that $R = E - (I + G)$ where G is the vertex-vertex adjacency matrix of the graph and E is the all-ones matrix: $E = \mathbf{1}\mathbf{1}^\top$. We add E so that the rewards are in the $[0, 1]$ interval and in fact are binary as required. Choose μ as the uniform distribution over the states. Note that $\mathbf{1}^\top (I + G) = 4\mathbf{1}^\top$ because the graph is 3-regular. Then, for $\pi \in \mathcal{M}_1(\mathcal{A})$,

$$\begin{aligned} J(\pi) &= \frac{1}{1-\gamma} - \mu(E + I + G)\pi - \frac{\gamma}{1-\gamma}\pi^\top (E + I + G)\pi \\ &= \frac{1}{1-\gamma} - \frac{1}{n}\mathbf{1}^\top (I + G)\pi - \frac{\gamma}{1-\gamma}\pi^\top (I + G)\pi \\ &= \frac{1}{1-\gamma} - \frac{4}{n} - \frac{\gamma}{1-\gamma}\pi^\top (I + G)\pi\,. \end{aligned}$$

Hence, $\max_{\pi \in \mathcal{M}_1([n])} J(\pi) = \frac{1}{1-\gamma} - \frac{4}{n} - \frac{\gamma}{1-\gamma}\frac{1}{\alpha(G)} \geq \frac{1}{1-\gamma} - \frac{4}{n} - \frac{\gamma}{1-\gamma}\frac{1}{m}$ holds if and only if $\alpha(G) \geq m$. Thus, the decision problem of deciding that $J(\pi) \geq a$ is at least as hard as the maximum independent set problem. As noted, this is an NP-hard problem, hence the result follows. ■

Potential remedy: Local search

Based on the theorem just proved it is not very likely that we can find computationally efficient planners to compete with the best policy in a restricted policy class, even if the class looks quite benign. This motivates aiming at some more modest goal, one possibility of which is to aim for computing stationary points of the map $J : \pi \mapsto \mu v^\pi$. Let $\Pi = \pi_\theta \,:\, \theta \in \mathbb{R}^d \in [0,1]^{\mathcal{S}\times\mathcal{A}}$ be the set of policies that can represented; we view these now as "large vectors". Then, in this approach we aim to identify $\pi^* \in \Pi$ (and its parameters) so that for any $\pi' \in \Pi$ and small enough $\delta > 0$ so that $\pi^* + \delta(\pi' - \pi^*) \in \Pi$, $J(\pi^* + \delta(\pi' - \pi^*)) \leq J(\pi^*)$. For δ small, $J(\pi^* + \delta(\pi' - \pi^*)) \approx J(\pi^*) + \delta\langle J'(\pi^*), \pi' - \pi^*\rangle$. Plugging this in into the previous inequality, reordering and dividing by $\delta > 0$ gives

$$\langle J'(\pi^*), \pi' - \pi^*\rangle \leq 0\,, \qquad \pi' \in \Pi\,. \tag{4}$$

Here, $J'(\pi)$ denotes the derivative of J. What remains to be seen is whether (1) relaxing the goal to computing π^* helps with the computation (and when) and (2) whether we can

get some guarantees for how well π^* satisfying (4) will do compared to $J^* = \max_{\pi \in \Pi} J(\pi)$, that is obtaining some **approximation guarantees**. For the latter we seek for some function ε of the MDP M and Π (or ϕ, when Π is based on some featuremap) so that

$$J(\pi^*) \geq J^* - \varepsilon(M, \Pi)$$

As to the computational approaches, we will consider a simple approach based on (approximately) following the gradient of $\theta \mapsto J(\pi_\theta$.

Notes

Access models

The reader may be wondering about what is the appropriate "access model" when π_θ is not restricted to the form given in (1). There are many possibilities. One is to develop planners for specific parametric forms. A more general approach is to let the planner access $\pi_\theta(\cdot|s)$ and $\frac{\partial}{\partial \theta}\pi_\theta(\cdot|s)$ for any s it has encountered and any value of $\theta \in \mathbb{R}^d$ it chooses. This is akin to the **first-order black-box oracle** model familiar from optimization theory.

From function approximation to POMDPs

The hardness result for policy search is taken from a paper of Vlassis, Littman and Barber, who actually were interested in the computational complexity of planning in partially observable Markov Decision Problems (POMDPs). It is in fact an important observation that with function approximation, planning in MDPs becomes a special case planning in POMDPs: In particular, if policies are restricted to depend on the states through a feature-map $\phi : \mathcal{S} \to \mathbb{R}^d$ (any two states with identical features will get the same action distribution assigned to them), then planning to achieve high reward with this restricted class is almost the same as planning to achieve high reward in a partially observable MDP where the observation function is ϕ. Planners for the former problem could still have some advantage though if they can also access the states: In particular, an online planner which is given a feature-map to help its search but is also given access to the states is in fact not restricted to return actions whose distribution follows a policy from the feature-restricted class of policies. In machine learning, in the analogue problem of competing with a best predictor within a class but using predictors that do not respect the restrictions put on the competitors are called **improper** and it is known that improper learning is often more powerful than proper learning. However, when it comes to learning online or in a batch fashion then feature-restricted learning and learning in POMDPs

become exact analogs. Finally, we note in passing that Vlassis et al. (2012) also add an argument that show that it is not likely that policy search is in NP.

Open problem: Hardness of approximate policy search

Provided that from an approximate solution to the Motzkin-Straus problem one can efficiently extract an approximate solution to the maximum independent set problem, it follows that the approximate version of policy search is also NP-hard. In particular, it is not hard to see with the same construction that if one has an efficient method find a policy with $J(\pi) \geq \max_\pi J_\pi - \varepsilon$ then this gives an efficient method to find an independent set of size $c\alpha(G)$ for the said 3-regular graphs where

$$c = \frac{1}{1 + \frac{1-\gamma}{\gamma}\varepsilon\alpha(G)} \geq \frac{1}{1 + \frac{1-\gamma}{\gamma}\varepsilon n} \geq 94/95\,,$$

where the last inequality follows if $\varepsilon \leq 0.5$, $\gamma \geq 0.5$ and $H := \frac{1}{1-\gamma} \geq \frac{n}{95/94-1} = 94n$ holds. Now, it is known that, unless P=NP, there is no polynomial time approximation algorithm for the maximal independent set problem with approximation factor $c = 94/95$ or better. Hence, we get that, unless P=NP, there is no polynomial time approximation algorithm for the policy search problem for any fixed $0 \leq \epsilon \leq 0.5$ provided the planning horizon is scaled with n so that $H = \mathrm{const} n$. (This is somewhat unsatisfactory given that the range of the optimal values is $1/(1-\gamma)$: It would be more natural to scale ϵ with $1/(1-\gamma)$, i.e., consider relative errors as in complexity theory.) Also, it remains an open problem to get a hardness result for a "constant" γ (independent of n).

The above is still dependent on whether an approximate solution to the maximum independent set problem can be extracted from an approximate solution to the Motzkin-Straus optimization problem.

Dealing with large action spaces

A common reason to consider policy search is because working with a restricted parametric family of policies holds the promise of decoupling the computational cost of learning and planning from the cardinality of the action-space. Indeed, with action-value functions, one usually needs an efficient way of computing greedy actions (with respect to some fixed action-value function). Computing $\arg\max_{a \in \mathcal{A}} q(s,a)$ in the lack of extra structure of the action-space and the function $q(s,\cdot)$ takes linear time in the size of $\mathcal{A}$, which is highly problematic unless $\mathcal{A}$ has a small cardinality. In many applications of

practical interest this is not the case: The action space can be "combinatorially sized", or even a subset of some (potentially multidimensional) continuous space.

If **sampling from $\pi_\theta(\cdot|s)$ can be done efficiently**, one may then potentially avoid the above expensive calculation. Thus, policy search is often proposed as a remedy to extend algorithms to work with large action spaces. Of course, this only applies if the sampling problem can indeed be efficiently implemented, which adds an extra restriction on the policy representation. Nevertheless, there are a number of options to achieve this: One can use for example an implicit representation (perhaps in conjunction with a direct one that uses probabilities/densities) for the policy.

For example, the policy may be "represented" as a map $f_\theta : \mathcal{S} \times \mathcal{R} \to \mathcal{A}$ so that sampling from $\pi_\theta(\cdot|s)$ is accomplished by drawing a sample $R \sim P$ from a fixed distribution over the set $\mathcal{R}$ and then returning $f(s, R) \in \mathcal{A}$. Clearly, this is efficient as long as f_θ can be efficiently evaluated at any of its inputs and the random value R can be efficiently produced. If f_θ is sufficiently flexible, one can in fact choose a very simple distribution for P, such as the standard normal distribution, or the uniform distribution.

Note that when $\mathcal{A}$ is continuous and the policies are deterministic is a special case: The key is still to be able to efficiently produce a sample from $\pi_\theta(\cdot|s)$, just in this case this means a deterministic computation.

The catch is that one may also still need the derivatives of $\pi_\theta(\cdot|s)$ with respect to the parameter θ and with an implicit representation as described above, it is unclear whether these derivatives can be efficiently obtained. As it turns out, this can be arranged if $f_\theta(\cdot|s)$ is made of composition of elementary (invertible, differentiable) transformations with this property (by the chain rule). This observation is the basis of various approaches to "neural" density estimation (e.g., Tabak and Vanden-Eijnden, 2010, Rezende, Mohamed, 2015, or Jaini et al. 2019).

References

- Vlassis, Nikos, Michael L. Littman, and David Barber. 2012. "On the Computational Complexity of Stochastic Controller Optimization in POMDPs." ACM Trans. Comput. Theory, 12, 4 (4): 1–8.
- Esteban G. Tabak. Eric Vanden-Eijnden. "Density estimation by dual ascent of the log-likelihood." Commun. Math. Sci. 8 (1) 217 - 233, March 2010.

- Rezende, Danilo Jimenez, and Shakir Mohamed. 2015. "Variational Inference with Normalizing Flows" link.
- Rezende, D. J., and S. Mohamed. 2014. "Stochastic Backpropagation and Approximate Inference in Deep Generative Models." ICML. link.
- Jaini, Priyank, Kira A. Selby, and Yaoliang Yu. 2019. "Sum-of-Squares Polynomial Flow." In Proceedings of the 36th International Conference on Machine Learning, edited by Kamalika Chaudhuri and Ruslan Salakhutdinov, 97:3009–18. Proceedings of Machine Learning Research. PMLR.
- Arora, Sanjeev, and Boaz Barak. 2009. Computational Complexity. A Modern Approach. Cambridge: Cambridge University Press.

The hardness of the maximum independent set problem is a classic result; see, e.g., Theorem 2.15 in the book of Arora and Barak (2009) above, though this proof does not show that the hardness also applies to the case of 3-regular graphs. Below is the paper that shows that approximating the maximum independent set size within a factor of $94/95 = 0.9894\ldots$ is NP-hard even for 3-regular graphs. The precise statement is in the main theorem statement on page 29 (this is the first, unnumbered and unnamed theorem on pdf page 3). In particular, the 2nd bullet point has this bound, specifically the hardness kicks in for approximation factors at least as large as $94/95$. I am very grateful for Zachary Friggstad who pointed me to this paper.

- Miroslav Chlebík, Janka Chlebíková: Inapproximability Results for Bounded Variants of Optimization Problems. FCT 2003: 27-38 DBLP page