16. Policy gradients

In this last lecture on planning, we look at policy search through the lens of applying gradient ascent. We start by proving the so-called policy gradient theorem which is then shown to give rise to an efficient way of constructing noisy, but unbiased gradient estimates in the presence of a simulator. We discuss at a high level the ideas underlying gradient ascent and stochastic gradient ascent methods (as opposed to more common case in machine learning where the goal is to minimize a loss, or objective function, we are maximizing rewards, hence ascending on the objective rather than descending). We then find out about the limitations of policy gradient even in the presence of "perfect representation" (unrestricted policy classes, tabular case) and perfect gradient information, which motivates the introduction of a variant known as "natural policy gradients" (NPG). We then uncover a close relationship between this method and Politex. The lecture concludes with comparing results for NPG and Politex.

The policy gradient theorem

Fix an MDP $M = (\mathcal{S}, \mathcal{A}, P, r)$ and a discount factor $0 \leq \gamma < 1$. Continuing from the last lecture for $\theta \in \mathbb{R}^d$ let π_θ be a stochastic policy: $\pi_\theta : \mathcal{S} \to \mathcal{M}_1(\mathcal{A})$. Further, fix a distribution $\mu \in \mathcal{M}_1(\mathcal{S})$ over the states and for a policy $\pi : \mathcal{S} \to \mathcal{M}_1(\mathcal{A})$ let

$$J(\pi) = \mu v^\pi$$

denote the expected value of using policy π in M from an initial state randomly chosen from μ. The policy gradient theorem gives sufficient conditions under which the map $\theta \mapsto J(\pi_\theta)$ is differentiable at some parameter $\theta = \theta_0$ and gives a "simple" expression for the gradient as a function of $\frac{d}{dx} M_{\pi_x} q^{\pi_{\theta_0}}$. Just to demistify this, for finite (or discrete) action spaces, for a memoryless policy π and function $q : \mathbb{R}^{\mathcal{S} \times \mathcal{A}} \to \mathbb{R}$, $M_\pi q$ is a function mapping states to reals defined via

$$(M_\pi q)(s) = \sum_a \pi(a|s) q(s,a)\,.$$

Hence, the derivative, $\frac{d}{dx} M_{\pi_x} q$ is actually quite simple. It is a function mapping states to d dimensional vectors which satisfies

$$\frac{d}{dx}(M_{\pi_x} q)(s) = \sum_a \frac{d}{dx} \pi_x(a|s) q(s,a)\,.$$

The theorem we give though is not limited to this case and also applies to when the action space is infinite and even when the policy is deterministic. For the theorem statement, recall that for a policy π we used $\tilde{\nu}_\mu^\pi$ to denote its discounted state occupancy measure. Also, for a function f, we use f' to denote its derivative.

Theorem (Policy Gradient Theorem): Fix an MDP $(\mathcal{S}, \mathcal{A}, P, r)$. For $x \in \mathbb{R}^d$, define the maps $f_\pi : x \mapsto \tilde{\nu}_\mu^\pi M_{\pi_x} q^\pi$ and $g_\pi : x \mapsto \tilde{\nu}_\mu^{\pi_x} v^\pi$. Fix $\theta_0 \in \mathbb{R}^d$. Assume that at least **one** of the following two conditions is met:

1. $\theta \mapsto f'_{\pi_\theta}(\theta_0)$ exists and is continuous in a neighborhood of θ_0 and $g'_{\pi_{\theta_0}}(\theta_0)$ exists;
2. $\theta \mapsto g'_{\pi_\theta}(\theta_0)$ exists and is continuous in a neighborhood of θ_0 and $f'_{\pi_{\theta_0}}(\theta_0)$ exists;

Then, $x \mapsto J(\pi_x)$ is differentiable at $x = \theta_0$ and

$$\frac{d}{dx} J(\pi_x)|_{x=\theta_0} = \frac{d}{dx} \tilde{\nu}_\mu^{\pi_{\theta_0}} M_{\pi_x} q^{\pi_{\theta_0}}|_{x=\theta_0} = \tilde{\nu}_\mu^{\pi_{\theta_0}} \frac{d}{dx} M_{\pi_x} q^{\pi_{\theta_0}}|_{x=\theta_0}\,, \tag{1}$$

where the last equality holds if $\mathcal{S}$ is finite.

For the second expression, we treat $\frac{d}{dx} M_{\pi_x} q^{\pi_{\theta_0}}$ as an $\mathrm{S} \times d$ matrix. Note that this fits well with our convention of treating functions as "column vectors" (hence $M_{\pi_x}) q^{\pi_{\theta_0}}$ is a vector of dimension S) and with the standard convention that a "vector derivative" creates "row vectors".

Above, the second expression where we moved the derivative with respect to the parameter inside the expression will only be valid in infinite state spaces when some additional regularity assumption is met. One such assumption is that $s \mapsto \| \frac{d}{dx} (M_{\pi_x} q^{\pi_{\theta_0}})(s)|_{x=\theta_0} \|$ is $\tilde{\nu}_\mu^{\pi_{\theta_0}}$-integrable.

In words, the theorem shows that the derivative of the performance of a policy can be obtained by integrating a simple derivative that involves the action-value function of the policy.

Of the two conditions of the theorem, the first condition is the one that is generally easier to verify. In particular, the condition on the continuous differentiability of $x \mapsto f_{\pi_x}$ at

$x = \theta_0$ is usually easy to verify. To show the differentiability of $x \mapsto g_{\pi_x}$ at $x = \theta_0$ just recall that if the partial derivatives of a function exist and are continuous the function is differentiable. Then recall that $\tilde{\nu}_\mu^{\pi_x} v = \sum_{t=0}^\infty \gamma^t \nu P_{\pi_x}^t v$ and hence its differentiability with respect to (say) x_1 follows if $x \mapsto \nu M_{\pi_x} P v$ is continuously differentiable at $x = \theta_0$. In effect, for finite state-action spaces, differentiability at θ_0 follows (and the conditions of the theorem are satisfied) as long as for any (s,a) state-action pair, the maps $x \mapsto \pi_x(a|s)$ have continuous partial derivatives at $x = \theta_0$.

Proof: The proof is based on a short calculation that starts with writing the value difference identity for $v^{\pi_x} - v^{\pi_{\theta_0}}$, multiplied from the right by μ, taking derivatives and then letting $x = \theta_0$.

The details are as follows: Recall from Calculus 101 the following result: Assume that $f = f(u,v)$ satisfies at least one of the two conditions:

1. $z \mapsto \frac{\partial}{\partial v} f(z,x)$ exists and is continuous in a neighborhood of $z = x$ and $\frac{\partial}{\partial u} f(u,x)|_{u=x}$ exists;
2. $z \mapsto \frac{\partial}{\partial u} f(x,z)$ exists and is continuous in a neighborhood of $z = x$ and $\frac{\partial}{\partial v} f(x,v)|_{v=x}$ exists.

Then $z \mapsto f(z,z)$ is differentiable at $z = x$ and

$$\frac{d}{dx} f(x,x) = \frac{\partial}{\partial u} f(x,x) + \frac{\partial}{\partial v} f(x,x)\,. \tag{2}$$

Let π', π be two memoryless policies. By the value difference identity,

$$\begin{aligned} v^{\pi'} - v^\pi &= (I - \gamma P_{\pi'})^{-1} [T_{\pi'} v^\pi - v^\pi] \\ &= (I - \gamma P_{\pi'})^{-1} [M_{\pi'} q^\pi - v^\pi]\,, \end{aligned}$$

where the last equality just used that that $T_{\pi'} v^\pi = M_{\pi'}(r + \gamma P v^\pi) = M_{\pi'} q^\pi$. Now let $\pi' = \pi_x$ and $\pi = \pi_{\theta_0}$ and multiply the value difference identity from the left by μ to get

$$\mu(v^{\pi_x} - v^{\pi_{\theta_0}}) = \tilde{\nu}_\mu^{\pi_x} [M_{\pi_x} q^{\pi_{\theta_0}} - v^{\pi_{\theta_0}}]\,. \tag{3}$$

Now, focusing on the first term on the right-hand-side, let

$$f(u,v) = \tilde{\nu}_\mu^{\pi_u} M_{\pi_v} q^{\pi_{\theta_0}}\,. \tag{4}$$

Provided that f is sufficiently regular in a neighborhood of (x,x) (to be discussed later), (2) gives that

$$\frac{d}{dx} f(x,x) = \frac{d}{du} \tilde{\nu}_\mu^{\pi_u} M_{\pi_x} q^{\pi_{\theta_0}}|_{u=x} + \frac{d}{dv} \tilde{\nu}_\mu^{\pi_x} M_{\pi_v} q^{\pi_{\theta_0}}|_{v=x}$$

Taking the derivative of both sides of (3) with respect to x and using the above display we get

$$\frac{d}{dx}J(x) = \frac{d}{dx}\mu(v^{\pi_x} - v^{\pi_{\theta_0}}) = \frac{d}{du}\tilde{\nu}_\mu^{\pi_u} M_{\pi_x} q^{\pi_{\theta_0}}|_{u=x} + \frac{d}{dv}\tilde{\nu}_\mu^{\pi_x} M_{\pi_v} q^{\pi_{\theta_0}}|_{v=x} + \frac{d}{dx}\tilde{\nu}_\mu^{\pi_x} v^{\pi_{\theta_0}}\,.$$

Now let $x = \theta_0$. Then, $M_{\pi_x} q^{\pi_{\theta_0}} = M_{\pi_{\theta_0}} q^{\pi_{\theta_0}} = v^{\pi_{\theta_0}}$. Hence, the first and the third term of the above display cancel each other and we get

$$\frac{d}{dx}J(\pi_x)|_{x=\theta_0} = \frac{d}{dv}\tilde{\nu}_\mu^{\pi_{\theta_0}} M_{\pi_v} q^{\pi_{\theta_0}}|_{v=\theta_0}\,.$$

Finally, the conditions to apply (2) to our f in (4) are met by our assumption on f_π and g_π, finishing the proof. ■

When the action-space is discrete and π_θ are stochastic policies, we can further manipulate the expression we obtained. In particular, in this case

$$(M_{\pi_x} q^{\pi_{\theta_0}})(s) = \sum_a \pi_x(a|s) q^{\pi_{\theta_0}}(s,a)$$

and thus, for finite $\mathcal{A}$,

$$\frac{d}{dx}(M_{\pi_x} q^{\pi_{\theta_0}})(s) = \sum_a \frac{d}{dx}\pi_x(a|s) q^{\pi_{\theta_0}}(s,a)\,. \tag{5}$$

While this can be used as the basis of evaluating (or approximating) gradient, it may be worthwhile to point out an alternate form which is available when $\pi_x(a|s) > 0$. In this case, using the chain rule we get

$$\frac{d}{dx}\log \pi_x(a|s) = \frac{\frac{d}{dx}\pi_x(a|s)}{\pi_x(a|s)}\,.$$

Using this in (5) we get

$$\frac{d}{dx}(M_{\pi_x} q^{\pi_{\theta_0}})(s) = \sum_a \pi_x(a|s)\left(\frac{d}{dx}\log \pi_x(a|s)\right) q^{\pi_{\theta_0}}(s,a)\,, \tag{6}$$

which has the pleasant property that it takes the form of an expected value over the actions of the **score function** of the policy map correlated with the action-value function.

Before moving on it is worth pointing out that an equivalent expression is obtained if $q^{\pi_{\theta_0}}(s,a)$ above is shifted by an arbitrary constant which may depend on θ_0 or s but not a. Indeed, since $\sum_a \pi_x(a|s) b(s,\theta_0) = b(s,\theta_0)$, differentiating both sides with respect to x gives $\sum_a \frac{d}{dx}\pi_x(a|a) b(s,\theta_0) = 0$. Hence, we also have

$$\frac{d}{dx}(M_{\pi_x} q^{\pi_{\theta_0}})(s) = \sum_a \pi_x(a|s) \left(\frac{d}{dx} \log \pi_x(a|s) \right) (q^{\pi_{\theta_0}}(s,a) - b(s,\theta_0)) \,. \tag{7}$$

This may have significance when using simulation to evaluate derivatives: One may attempt to use an appropriate "bias" term to reduce the variance of the estimate of the gradient. Before discussing simulation any further, it may be also worthwhile to discuss what happens when the action-space is infinite.

For countable infinite action spaces, the only difference is that (5) may not always hold. An easy sufficient condition for this to hold is that $\sum_a \| \frac{d}{dx} \pi_x(a|s) \| \, |q^{\pi_{\theta_0}}(s,a)|$ is summable, or equivalently, $\| \frac{d}{dx} \log \pi_x(a|s) \| \, |q^{\pi_{\theta_0}}(s,a)|$ is $\pi_x(\cdot|s)$-summable/integrable.

For uncountably infinite action spaces, this argument works with the minimal necessary changes. In the most general case, $\pi_\theta(\cdot|s)$ is a probability measure over $\mathcal{A}$ and its derivative is a vector-valued measure. The formulae derived above (e.g., (7)) remain valid if we replace the sum with an integral when $\pi_\theta(\cdot|s)$ is given in the form of a density with respect to some fixed measure λ over $\mathcal{A}$:

$$\frac{d}{dx}(M_{\pi_x} q^{\pi_{\theta_0}})(s) = \int_{\mathcal{A}} \pi_x(a|s) \left(\frac{d}{dx} \log \pi_x(a|s) \right) (q^{\pi_{\theta_0}}(s,a) - b(s,\theta_0)) \lambda(da) \,. \tag{8}$$

In fact, this is a strictly more general form: (7) is a special case of (8) when λ is set to the counting measure over $\mathcal{A}$.

In the special case when $\pi_\theta(\cdot|s) = \delta_{f_\theta(s)}(\cdot)$ (a Dirac at $f_\theta(s)$), in words, when we have a deterministic policy map and f is differentiable with respect to θ, it is better to start from the formula given in the theorem.

Indeed, in this case,

$$(M_{\pi_x} q^{\pi_{\theta_0}})(s) = q^{\pi_{\theta_0}}(s, f_\theta(s))$$

and hence

$$\frac{d}{dx}(M_{\pi_x} q^{\pi_{\theta_0}})(s) = \frac{d}{dx} q^{\pi_{\theta_0}}(s, f_x(s))$$

and thus, if either $\mathcal{S}$ is finite or an appropriate regularity condition holds,

$$\frac{d}{dx} J(\pi_x)|_{x=\theta_0} = \tilde{\nu}_\mu^{\pi_{\theta_0}} \frac{d}{dx} q^{\pi_{\theta_0}}(\cdot, f_x(\cdot))|_{x=\theta_0} \,.$$

If $a \mapsto q^{\pi_{\theta_0}}(s,a)$ is differentiable and $x \mapsto f_x(s)$ is also differentiable at $x = \theta_0$ for every s then

$$\frac{d}{dx} J(\pi_x)|_{x=\theta_0} = \tilde{\nu}_\mu^{\pi_{\theta_0}} \frac{\partial}{\partial a} q^{\pi_{\theta_0}}(\cdot, f_{\theta_0}(\cdot)) \frac{d}{dx} f_x(\cdot)|_{x=\theta_0} \,,$$

which is known as the "deterministic policy gradient formula".

Gradient methods

The idea of gradient methods is to make small steps in the parameter space in the direction of the gradient of an objective function that is to be maximized. In the context of policy search, this works as follows: If $x_i \in \mathbb{R}^d$ denotes the parameter vector in round i,

$$x_{i+1} = x_i + \alpha_i \nabla_x J(\pi_x)|_{x=x_i}\,,$$

where for f differentiable, $\nabla_x f = (\frac{d}{dx} f)^\top$ is the "gradient" (transpose of derivative). Above, α_i is a positive tuning parameter, called the "stepsize" of the update. The idea is that the "gradient" points in the direction where the function is expected to grow. Indeed, since by definition,

$$f(x') = f(x) + f'(x)(x'-x) + o(\|x'-x\|)$$

if $x' = x + \delta (f'(x))^\top$,

$$f(x + \delta (f'(x))^\top) = f(x) + \delta \|f'(x)\|_2^2 + o(|\delta|)\,,$$

or

$$\frac{f(x + \delta (f'(x))^\top) - f(x)}{\delta} = \|f'(x)\|_2^2 + o(1)\,,$$

For **any** δ sufficiently small so that the $o(1)$ term (in absolute value) is below $\|f'(x)\|_2^2$, we see that the right-hand side is positive, hence so is the left-hand side, as claimed. This simple observation is the basis of a huge number of algorithmic variants. In the lack of extra structure the best we can hope from a gradient method is that it will end up in the vicinity of a stationary point. In the presence of extra structure (.e.g, concave function to be maximized), convergence to a global maximum can be guaranteed.

In all cases the key to the success of gradient methods is the appropriate choice of the stepsizes; these choices are based on a refinement of the above simple argument that shows that moving towards the direction of the gradient helps. There are also ways of "speeding up" convergence; these "acceleration methods" use a refined iteration (two iterates updated simultaneously) and can greatly speed up convergence. As there are many excellent texts that describe various aspects of gradient methods which cover these ideas, we will not delve into them any further, but I will rather give some pointers to this literature in the endnotes.

The elephant in the room here is that the gradient of J is not readily available. The next best thing then is to attempt to build an estimate G of $\nabla_x J(\pi_x)$. In the planning setting, the question is whether one can get reasonable estimates of this gradient using a simulator.

Gradient estimation

Generally speaking there are two types of errors when construction an estimate of the gradient: The one that is purely random, and the one that is not. Defining $g(x) = \mathbb{E}[G]$, $b(x) = \nabla_x J(\pi_x) - g(x)$ measures the "bias" of the gradient estimate, while $G - g(x)$ is the noise. Gradient methods with decreasing (or small) stepsizes naturally "average out" the noise. The version of gradient methods that are able to do this are called **stochastic gradient** methods. Naturally, these methods are slower when the noise is larger and in general cannot converge faster than how fast the noise averages out. In particular, in persistent noise (i.e., noise with nonvanishing variance), the best rate available for stochastic gradient methods is $O(1/\sqrt{t})$. While this can be slower than what can be achieved without noise, if the iteration cost is polynomial in the relevant quantities, the total cost of achieving an $\varepsilon > 0$ stationary point can be bounded by a polynomial in these quantities and $1/\varepsilon^2$.

When the gradient estimates are biased, the bias will in general put a limit on how close a gradient method can get to a stationary point. While generally a zero bias is preferred to a nonzero bias, a nonzero bias which is positively aligned with the gradient ($\langle b(x), \nabla_x J(\pi_x)\rangle \geq 0$) does not hurt (again, for small stepsizes). When there is no way to guarantee that the bias is positively aligned with the gradient, one may get back into control by making sure that the magnitude of the bias is small relative to the magnitude of the gradient.

The next question is of course, how to estimate the gradient. For this many approaches have been proposed in the literature. When a simulator is available, as in our case, a straightforward approach is to start from the policy gradient theorem. Indeed, under mild regularity conditions (e.g., if there are finitely many states) (1) together with (8) gives

$$\frac{d}{dx} J(\pi_x) = \int_{\mathcal{S}} \tilde{\nu}_\mu^{\pi_x}(ds) \int_{\mathcal{A}} \pi_x(a|s) \left(\frac{d}{dx} \log \pi_x(a|s)\right) (q^{\pi_x}(s,a) - b(s,x)) \lambda(da)\,. \tag{9}$$

Now note that $(1-\gamma)\tilde{\nu}_\mu^{\pi_x}$ is a probability measure over $\mathcal{S}$. Let $S_0, A_0, S_1, A_1, \dots$ be an infinite sequence of state-action pairs obtained by simulating policy π_x starting from $S_0 \sim \mu$. In particular, $A_t \sim \pi_x(\cdot|S_t)$ and $S_{t+1} \sim P_{A_t}(S_t)$ for any $t \geq 0$. In addition, define T_1, T_2 to be independent of each other and from the trajectory $S_0, A_0, S_1, A_1, \dots$ and have a geometric distribution with parameter $1-\gamma$. Then,

$$G = \frac{1}{1-\gamma}\frac{d}{dx}\log \pi_x(A_{T_1}|S_{T_1})\left(\sum_{t=0}^{T_2-1} r_{A_{T_1+t}}(S_{T_1+t}) - b(S_{T_1}, x)\right)$$

is an unbiased estimate of $\frac{d}{dx}J(\pi_x)$:

$$\mathbb{E}[G] = \frac{d}{dx}J(\pi_x)\,.$$

The argument to show this has partially be given earlier in Lecture 8. One can also show that G has a finite covariance matrix, as well as that the expected effort to obtain G is $O(\frac{1}{1-\gamma})$.

Vanilla policy gradients (PG) with some special policy classes

Given the hardness result presented in the previous lecture, there is no hope that gradient methods or any other method will find the global optima of the objective function in policy search in a policy-class agnostic manner. To guarantee computational efficiency, one then

1. either needs to give up on convergence to a global optima, or
2. give up on generality, i.e., give up on that the method should work for any policy class and/or policy parameterization.

Gradient ascent to find a good policy ("vanilla policy gradients") is one possible approach to take even if it faces these restrictions. In fact, gradient ascent in some cases will find a globally optimal policy.

In particular, it has been long known that with small enough stepsizes gradient ascent converges at a reasonable speed to a global optimum provided that two conditions hold:

1. The objective function f is smooth (its derivative is Lipschitz continuous);
2. The objective function is gradient dominated, i.e., with some constants $c > 0$, $p \geq 1$, f satisfies $\sup_x f(x) - f(x') \leq c\|f'(x')\|_2^p$ for any $x' \in \mathbb{R}^d$.

An example when both of these conditions are met is the **direct policy parameterization**, which does not allow any compression and is thus not helpful per se, but can serve as a test-case to see how far policy gradient (PG) methods can be pushed.

In this case, the parameter vector θ is SA dimensional. By allowing "two-dimensional index", $\pi_\theta(a|s) = \theta_{s,a}$, that is, the parameters encode the action selection probabilities in a direct manner. In this case, since the components of θ represent probabilities, they need

to be nonnegative and the appropriate components needs to sum to one. Hence, $\theta \in \Theta$ for an appropriate set $\Theta \subset [0,1]^{\mathrm{SA}}$. Accordingly, one needs to change gradient ascent.

This is done as follows: When a proposed update moves the parameter vector outside of Θ, the proposed updated parameter vector is "back-projected" to Θ. For the projection there are a number of reasonable options, such as choosing the point within Θ which is closest to the proposed point in the standard Euclidean distance. With this modification, gradient ascent can be shown to converge at a reasonable speed in this case. This parallels the methods that were developed for the tabular case (policy iteration, value iteration). In fact, the algorithm can be seen as a "smoother", incremental version of policy iteration, which gradually adjusts the probabilities assigned to the individual actions. Using π_i to denote the ith policy, from the policy gradient theorem one gets

$$\tilde{\pi}_{i+1}(a|s) = \pi_i(a|s) + \alpha_i \tilde{\nu}_\mu^{\pi_i}(s) q^{\pi_i}(s,a)\,,$$

and

$$\pi_{i+1}(\cdot|s) = \arg\min_{p \in \mathcal{M}_1(\mathcal{A})} \|p - \pi_{i+1}(\cdot|s)\|_2, \qquad s \in \mathcal{S}\,.$$

Thus, the probability of an action in a state is increased in proportion to the value of that state.

That the action-value of action a at state s is multiplied with the discounted occupancy at s induced by using policy π_i started from μ is a bit of a surprise. In particular, if a state is inaccessible under policy π_i, the corresponding probabilities will not be updated. In fact, because this, the above iteration may get stuck at a suboptimal policy. The reader is invited to construct an example when this happens. To prevent this, it turns out to be sufficient if there is a constant $C > 0$ such that it holds that

$$\tilde{\nu}_\mu^{\pi^*}(s) \geq C\mu(s)\,, \qquad \text{for all } s \in \mathcal{S}\,, \tag{10}$$

where π^* is an optimal policy. Since μ appears on both sides and π^* is unknown, this condition does not look to helpful. However, if one chooses μ to be positive everywhere, the condition is clearly met. In any case, when (10) holds, gradient dominance and smoothness can be both verified, which in turn implies that the above update will converge at a geometric speed, the geometric speed involves an instance dependent constant which has no polynomial bound in terms of $H_\gamma = 1/(1-\gamma)$ and the size of the state-action space. Needless to say this is quite unattractive.

Policy gradient methods can be sensitive to how policies are parameterized. For illustration, consider still the "tabular case", just now change the way the memoryless

policies are represented. One possibility is to use the Boltzmann, also known as the **softmax** representation. In this case $\theta \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ and

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

A straightforward calculation gives

$$\frac{\partial}{\partial\theta_{s,a}}\log\pi_\theta(a'|s') = \mathbb{I}(s=s',a=a') - \pi_\theta(a|s)\mathbb{I}(s=s')$$

and hence

$$\begin{aligned}\frac{\partial}{\partial\theta_{(s,a)}}J(\pi_\theta) &= \sum_{s'}\tilde{\nu}_\mu^{\pi_\theta}(s')\sum_{a'}\pi_\theta(a'|s)\frac{\partial}{\partial\theta_{s,a}}\log\pi_\theta(a'|s')q^{\pi_\theta}(s',a') \\ &= \nu_\mu^{\pi_\theta}(s,a)\left(q^{\pi_\theta}(s,a) - v^{\pi_\theta}(s)\right),\end{aligned}$$

where recall that ν_μ^π is the discounted state-occupancy measure over the state-action pairs of policy π when the initial state distribution is μ. The difference in the bracket on the right-hand side is known as the **advantage** of action a and, accordingly, the function

$$\mathfrak{a}^\pi = q^\pi - v^\pi\,,$$

which is a function mapping state-action pairs to reals, is called the **advantage function** underlying policy π. To justify the terminology, note that policy iteration can be seen as choosing in each state the action that maximizes the "advantage". Thus, we expect that we get a better policy if the "probability mass" in the action distribution is shifted towards actions with a larger advantage. Note though that advantages (as defined above) can also be negative and in fact if π is optimal, all actions have nonnegative advantages only.

The gradient ascent rule prescribes that

$$\theta_{i+1} = \theta_i + \alpha_i\nu_\mu^{\pi_{\theta_i}}\circ\mathfrak{a}^{\pi_{\theta_i}}\,,$$

where $\circ$ denotes componentwise product. While this is similar to the previous update, now the meaning of parameters is quite different. In fact, just because a parameter is increased does not necessarily mean that the probability of the corresponding action is increased: This will only happen if the increase of this parameter exceeds that of the other parameters "at the same state". By slightly abusing notation with defining $\pi_i = \pi_{\theta_i}$, we have

$$\pi_{i+1}(a|s) \propto \pi_i(a|s)\exp(\alpha_i\nu_\mu^{\pi_i}(s,a)\mathfrak{a}^{\pi_i}(s,a))\,. \tag{11}$$

Just like in the previous update rule, we also see the occupancy measure "weighting" the update. This is again not necessarily helpful and if anything, again, speaks to the arbitrariness of gradient methods. And while this does not entirely stop policy gradient to find an optimal policy, and again, one can even show that the speed is geometric, though, as before, the algorithm altogether fails to run in polynomial time in the relevant quantities. For this theorem which we give without proof recall that $H_\gamma = 1/(1-\gamma)$.

Theorem (PG is slow with Boltzmann policies): There exists universal constants $\gamma_0, c, C > 0$ such that for any $\gamma_0 < \gamma < 1$, if $\mathrm{S} > CH_\gamma^6$ then one can find a discounted MDP with S states and 3 actions, setting μ to be the uniform distribution and initializing the parameters so that π_0 is the uniform random policy, softmax PG with a constant stepsize of $\alpha > 0$ takes at least

$$\frac{c}{\alpha}\mathrm{S}^{2^{\Omega(H_\gamma)}}$$

iterations.

As one expects that without any compression, the chosen planner should behave reasonably, this rules out the "vanilla" version of policy gradient.

Natural policy gradient (NPG) methods

In fact, a quite unsatisfactory property of gradient ascent that the speed at which it converges can greatly depend on the parameterization used. Thus, for the same policy class, there are many possible "gradient directions", depending on the parameterization chosen. What is a gradient direction for one parameterization is not necessarily a gradient direction for another one. But what is common about these directions that an infinitesimal step along them is guaranteed increase the objective. One can in fact take a direction obtained with a parameterization and look at what direction it gives with another parameterizations. To get some order, consider transforming all these directions into the space that corresponds to the direct parameterization. It is not hard to see that all possible directions that are within 90 degrees of the gradient direction with this parameterization can be obtained by considering an appropriate parameterization.

More generally, regardless of parameterization, all directions within 90 degrees of the gradient direction are ascent directions. This motivates changing the stepsize α_i from a

scalar to a matrix A_i. Clearly, to keep the angle between the original gradient direction g and the transformed direction $A_i g$ below 90 degrees, $g^\top A_i g \geq 0$ has to hold. For A_i symmetric, this restricts the set of matrix "stepsizes" to the set of positive definite matrices (still, a large set).

There are many ways to choose a matrix stepsize. Newton's method is to choose it so that the direction is the "best" if the function is replaced by its local quadratic approximation. This provably helps to reduce the number of iterations when the objective function is "ill-conditioned", though all matrix stepsize methods incur additional cost per each iteration, which will often offset the gains.

Another idea, which comes from statistical problems where one often works with distributions is to find the direction of update which coincides with the direction one would obtain if one used the steepest descent direction directly in the space of distributions where distances are measured with respect to relative entropy. In some cases, this approach, which was coined the "natural gradient" approach, has been shown to give better results, though the evidence is purely empirical.

As it turns out, the matrix stepsize to be used with this approach is the (pseudo)inverse of the so-called Fischer information matrix. In our context, for every state, we have distributions over the actions. Fixing a state s, the Fischer information matrix becomes

$$F_x(s) = \frac{d}{dx}\log \pi_x(\cdot|s)\, \frac{d}{dx}\log \pi_x(\cdot|s)^\top .$$

To get the "information rate" over the states, one can sum these matrices up, weighted by the discounted state occupancy measure underlying μ and π_x to get

$$F(x) := \nu_\mu^{\pi_x} F_x .$$

The update rule then takes the form

$$x_{i+1} = x_i + \alpha_i F(x_i)^\dagger \nabla_x J(\pi_x) ,$$

where for a square matrix A, $A^\dagger$ denotes the pseudoinverse of A. Interestingly, the update direction can be obtained without calculating F and inverting it:

Proposition: We have

$$(1-\gamma) F(x)^\dagger \nabla_x J(\pi_x) = \arg\min_{w\in\mathbb{R}^d} \nu_\mu^{\pi_x} \left(w^\top \nabla_x \log \pi_x(\cdot|\cdot) - \mathfrak{a}^{\pi_x}\right)^2 ,$$

where $\mathfrak{a}^{\pi_x} = q^{\pi_x} - v^{\pi_x}$ and $\arg\min$ chooses the minimum $\|\cdot\|_2$-norm solution if multiple minimizers exist.

Proof: Just recall the formula that gives the solution to a least-squares problem. The details are left to the reader. ■

As an example of how things look like consider the case when π_x takes the form of a Boltzmann policy:

$$\pi_x(a|s) \propto \exp(x^\top \phi(s,a))\,,$$

where $\phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^d$ is a feature-map. Then, assuming that there are finitely many actions,

$$\nabla_x \log \pi_x(a|s) = \underbrace{\phi(s,a) - \sum_{a'} \pi_x(a'|s)\phi(s,a')}_{\psi_x(s,a)}\,.$$

Then, the natural policy gradient update takes the form

$$x_{i+1} = x_i + \alpha_i w_i\,,$$

where

$$w_i = \arg\min_{w \in \mathbb{R}^d} \nu_\mu^{\pi_x} \left(w^\top \psi_x - \mathfrak{a}^{\pi_{x_i}}\right)^2$$

In the tabular case ($d = \mathrm{SA}$, no compression),

$$w_i(s,a) = \mathfrak{a}^{\pi_{x_i}}(s,a)$$

and thus

$$\pi_{i+1}(a|s) \propto \pi_i(a|s) \exp(\alpha_i \mathfrak{a}^{\pi_i}(s,a)) = \pi_i(a|s) \exp(\alpha_i q^{\pi_i}(s,a))\,.$$

Note that this update rule eliminates the term $\nu_\mu^{\pi_i}(s,a)$ term that we have previously seen (cf. (11)).

NPG is known to enjoy a reasonable speed of convergence, which gives altogether polynomial planning time. This is promising. No similar results are available for the nontabular case.

Note that if we (arbitrarily) change the definition of w_i by replacing ψ_x above with ϕ and a^{π_x} with q^{π_x}, we get what has been called in the literature Q-NPG:

$$w_i = \arg\min_{w \in \mathbb{R}^d} \nu_\mu^{\pi_x} \left(w^\top \phi - q^{\pi_x} \right)^2 .$$

Note that the only difference between Q-NPG and Politex is that in Politex one uses

$$w_i = \arg\min_{w \in \mathbb{R}^d} \hat{\nu} \left(w^\top \phi - q^{\pi_x} \right)^2 ,$$

where $\hat{\nu}$ is the measure obtained from solving the G-optimal design problem.

The price of not using $\hat{\nu}$ but using $\nu_\mu^{\pi_x}$ in Q-NPG is that the approximation error in Q-NPG becomes

$$\frac{C\varepsilon}{(1-\gamma)^{1.5}}$$

where

$$C = \left\| \frac{d\tilde{\nu}_\mu^{\pi^*}}{d\mu} \right\|_\infty$$

gives a bound on how much the distribution μ differs from that of obtained when the optimal policy π^* is followed from μ. As was argued before, it is necessary that C is finite for policy gradient methods not to "get stuck" at local optima. However, C can be arbitrarily large even for finite state-action MDPs; an in fact it is the presence of C that makes the policy gradient with the direct parameterization a slow algorithm.

In contrast, the same quantity in Politex is

$$\frac{\sqrt{d}\varepsilon}{1-\gamma} .$$

Not only the uncontrolled constant C is removed, but the dependence on the planning horizon is also improved. Other than these differences, the results available for Q-NPG are similar to that of Politex and in fact the proof technique to obtain the results is also the same.

The proof of the Calculus 101 result

For completeness, here is the proof of (2). For the proof recall that for a function $g : \mathbb{R}^d \to \mathbb{R}$, $\frac{d}{dx} g(x_0)$ is the unique linear operator (row vector, in the Euclidean case) that satisfies

$$g(x) = g(x_0) + \frac{d}{dx} g(x_0)(x - x_0) + o(\|x - x_0\|) \text{ as } x \to x_0 .$$

Hence, it suffices to show that

$$f(x',x') = f(x,x) + \left(\frac{\partial}{\partial u} f(u,x)|_{u=x} + \frac{\partial}{\partial v} f(x,v)|_{v=x}\right)(x'-x) + o(\|x'-x\|)\,.$$

To minimize clutter we will write $\frac{\partial}{\partial u} f(x',x)$ for $\frac{\partial}{\partial u} f(u,x)|_{u=x'}$ (and similarly we write $\frac{\partial}{\partial v} f(x,x')$ for $\frac{\partial}{\partial v} f(x,v)|_{v=x'}$).

By definition we have

$$f(x',x') = f(x',x) + \frac{\partial}{\partial v} f(x',x)(x'-x) + o(\|x'-x\|)$$

and

$$f(x',x) = f(x,x) + \frac{\partial}{\partial u} f(x,x)(x'-x) + o(\|x'-x\|)\,.$$

Putting these together we get

$$\begin{aligned}
f(x',x') &= f(x,x) + \left(\frac{\partial}{\partial v} f(x',x) + \frac{\partial}{\partial u} f(x,x)\right)(x'-x) + o(\|x'-x\|) \\
&= f(x,x) + \left(\frac{\partial}{\partial v} f(x,x) + \frac{\partial}{\partial u} f(x,x)\right)(x'-x) \\
&\qquad + \left(\frac{\partial}{\partial v} f(x',x) - \frac{\partial}{\partial v} f(x,x)\right)(x'-x) + o(\|x'-x\|) \\
&= f(x,x) + \left(\frac{\partial}{\partial v} f(x,x) + \frac{\partial}{\partial u} f(x,x)\right)(x'-x) + o(\|x'-x\|)\,.
\end{aligned}$$

where the last equality follows if $\frac{\partial}{\partial v} f(x',x) - \frac{\partial}{\partial v} f(x,x) = o(1)$ as $x' \to x$, i.e., if $x' \mapsto \frac{\partial}{\partial v} f(x',x)$ is continuous at $x' = x$.

That the result also holds under the assumption that $x' \mapsto \frac{\partial}{\partial u} f(x,x')$ is continuous at $x' = x$ follows from a symmetric argument. ■

Summary

While policy gradient methods remain extremely popular and the idea of directly searching in the set of policies is attractive, at the moment it appears that they not only lack theoretical support, but the theoretical results suggest that it is hard to find any setting where policy gradient methods would be provably competitive with alternatives. At minimum, they need careful choices of policy parameterizations and even in that case the update rule may need to be changed to guarantee efficiency and effectiveness, as we have seen above. As an approach to algorithm design their main advantage is their generality and a strong support through various software libraries. Compared to vanilla "dynamic programming" methods they make generally smaller, more incremental changes to the policies, which seems useful. However, this is also achieved by methods

like Politex, which is derived using a "bound minimization" approach. While this may seem more ad hoc than following gradients, in fact, one may argue that following gradients is more ad hoc as it fails to guarantee good performance. However, perhaps the most important point here is that one should not care too much about how a method is derived, or what "interpretation" it may have (is Politex a gradient algorithm? does this matter?). What matters is the outcome: In this case how the methods perform. It is thus wise to learn about all possible ways of designing algorithms, especially since there is much room for improving the performance of current algorithms.

Notes

Philip Thomas (2014, see citation below) takes a careful look at the claims surrounding natural gradient descent. One claim that is often heard is that natural gradient descent will speed up convergence. This is usually back up by giving a demonstration (e.g., Kakade, 2002, or Amari, 1998). However, it is far from clear whether this speedup will necessarily happen. As it turns out, this is far from being true. In fact, natural policy gradient can cause divergence even where following the normal gradient is guaranteed to converge to a global optimum. An example of this is given in Section 6.5 of the paper of Thomas (2014).

References

- Amari, S. Natural gradient works efficiently in learning.Neural Computation, 10:251–276, 1998.
- Kakade, S. A natural policy gradient. In Advances in Neural Information Processing Systems, volume 14, pp.1531–1538, 2002.
- Bagnell, J. A. and Schneider, J. Covariant policy search. In Proceedings of the International Joint Conference on Artificial Intelligence, pp. 1019–1024, 2003.
- Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y. (1999). Policy gradient methods for reinforce-ment learning with function approximation. In Neural Information Processing Systems 12, pages 1057–1063.
- Silver, David, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. 2014. "Deterministic Policy Gradient Algorithms." In ICML. http://hal.inria.fr/hal-00938992/.
- Bhandari, Jalaj, and Daniel Russo. 2019. "Global Optimality Guarantees For Policy Gradient Methods," June. https://arxiv.org/abs/1906.01786v1.

- Agarwal, Alekh, Sham M. Kakade, Jason D. Lee, and Gaurav Mahajan. 2019. “On the Theory of Policy Gradient Methods: Optimality, Approximation, and Distribution Shift.” arXiv [cs.LG]. arXiv. http://arxiv.org/abs/1908.00261.
- Mei, Jincheng, Chenjun Xiao, Csaba Szepesvari, and Dale Schuurmans. 2020. “On the Global Convergence Rates of Softmax Policy Gradient Methods.” arXiv [cs.LG]. arXiv. http://arxiv.org/abs/2005.06392.
- Zhang, Junyu, Alec Koppel, Amrit Singh Bedi, Csaba Szepesvari, and Mengdi Wang. 2020. “Variational Policy Gradient Method for Reinforcement Learning with General Utilities.” arXiv [cs.LG]. arXiv. http://arxiv.org/abs/2007.02151.
- Bhandari, Jalaj, and Daniel Russo. 2020. “A Note on the Linear Convergence of Policy Gradient Methods.” arXiv [cs.LG]. arXiv. http://arxiv.org/abs/2007.11120.
- Chung, Wesley, Valentin Thomas, Marlos C. Machado, and Nicolas Le Roux. 2020. “Beyond Variance Reduction: Understanding the True Impact of Baselines on Policy Optimization.” arXiv [cs.LG]. arXiv. http://arxiv.org/abs/2008.13773.
- Li, Gen, Yuting Wei, Yuejie Chi, Yuantao Gu, and Yuxin Chen. 2021. “Softmax Policy Gradient Methods Can Take Exponential Time to Converge.” arXiv [cs.LG]. arXiv. http://arxiv.org/abs/2102.11270.
- Thomas, Philip S. “GeNGA: A Generalization of Natural Gradient Ascent with Positive and Negative Convergence Results.” ICML 2014. http://proceedings.mlr.press/v32/thomasb14.pdf.

The paper to read about natural gradient methods:

- Martens, James. 2014. “New Insights and Perspectives on the Natural Gradient Method,” December. https://arxiv.org/abs/1412.1193v9. Last update: September, 2020.