The Soundness of Function Approximation (under local improvement) or Finding the Gradient in the Space of all Policies

Sham Kakade John Langford

1 Motivation

Gradient algorithms guarantee improvement but they do not guarantee that the policy we find will have a high payoff. We seek to find an algorithm that can not only improve but also find a policy with a large payoff. Further, we desire the algorithm to be able to adapt it's improvement to the amount of data it receives.

2 Introduction

Function approximation has gotten a bad wrap due to the fact that theoretical guarantees are weak and policy degradation has been empirically observed. This has lead to the recent popularity of gradient methods, where direct optimization of a parameterized policy is possible. However, the question of how to parameterize a policy which can return a high payoff is obscured in gradient methods (though improvement is guaranteed regardless of what parameterization is used). In value function methods, crudely, the answer is evident: the values must accurately approximate the true values to obtain a near optimal policy. We now will propose a scheme of function approximation which is sound, and allows us to directly address the question of how to find a near optimal policy.

2.1 What is wrong with function approximation

The standard theoretical guarantee of approximate value functions is for the l_∞ error $\varepsilon = \max_s |V(s) - \tilde{V}(s)|$ (where $\tilde{V}(s)$ is the approximate value function) then the greedy policy based on these values has expected discounted reward $\eta \geq \eta^* - 2\gamma\varepsilon/(1-\gamma)$ (where η^* is the optimal possible). This is fairly weak and trivial guarantee, since even for large gamma, a greedy update could generate a significantly worse policy even with an *accurate* approximation. This is essentially due to the fact that a greedy policy could be significantly different from the current policy and the error can be compounded $1/(1-\gamma)$ times. Let us enumerate the most damning criticisms of function approximation methods, which we must address (see Baxter's intro. for most of these criticisms):

1. Pretty much all existing algorithms seek to minimize an error defined under a different norm (usually l_2) which does not guarantee good performance improvement.

2. Obviously if ε is small enough then things are fine. However, this 'small enough' value, under the l_∞ norm, is essentially saying that we have effectively done a table lookup policy.

Further, just because we might be able to find a good policy under our approximation scheme does not imply that there don't exist parameters which could give a good greedy policy. The use of the l_2 norm is not theoretically justified and seems to be used with the hope that it might be possible to make significant improvement with only a crude approximation (even though the l_∞ norm will generally be very large). Using the l_∞ is also a very costly demand. Although non-parametric models could allow arbitrarily good FA, this is very costly since after a policy improvement step we have to repeat. It's not that we don't desire a good approximation but the degree of fit necessary could be costly and infeasible (since we are just going to change the policy anyways).

2.2 What we desire

A meaningful theoretical guarantee must tie using a practical metric, such as the l_2, to performance improvement. We now suggest such a method. The key insight in obtaining such a guarantee is that the policy update steps must be local (ie must only slightly change the current policy). The lack of strong results for FA seem to stem from the fact that the policy update uses the argmax operator (and thus could make drastic changes to the policy). Our theoretical result will be more reasonable.

The first step we must take is to choose a local update rule to analyze, since the greedy update rule underlies the lack of performance guarantees.

3 FA

Let us now consider the case where have some approximate advantages $\tilde{A}$ ($\tilde{A}$ denotes a vector in state-action space), or if we have found some approximate state-action value $\tilde{Q}$ then the approximate advantage is $\tilde{A}(s,a) = \tilde{Q}(s,a) - <\tilde{Q}(s,a')>_{\pi(s,a')}$. Naively, we would like $\tilde{A}$ to reflect the true advantages, A, for the policy to improve significantly. Let's now consider the local update rule:

$$\pi_s(a) \leftarrow \pi_s(a)\exp(\alpha\tilde{A}(s,a)) \tag{1}$$

Intuitively, we increase the chance of doing actions with high advantages the fastest. Note that for an infinite α we end up with a greedy policy, *ie* one that would result from a greedy policy improvement step.

(Side note: This is exactly the simplest rule we were considering before for policy improvement. Also, under the natural gradient, this is exactly what the gradient update would give for small α, where $\tilde{A}$ is the advantage fit from the compatible policy. In essence, a gradient method's performance is tied to how good its compatible $\tilde{A}$ is. Gradient methods do not address how to choose a parameterized policy, which induces a parameterized $\tilde{A}$, that can give a good approximation of A.)

Function approximation directly addresses the question of how to find a good state-action value function $\tilde{Q}$ or advantage $\tilde{A}$. However, the theoretical guarantees for function approximation are weak. Clearly, a line search over α will guarantee a non-decrease in performance, and an infinite α will clearly return a policy that is equivalent to what greedy policy improvement would return. However, the question we now seek to address is how to fit $\tilde{A}$ such that we get significant improvement.

First, let us define the inner product for vectors in state-action space to be:

$$< x, x' >_\nu = \sum_{sa} \nu(s,a) x(s,a) x'(s,a)$$

and we will define $\nu(s,a) = \rho(s)\pi_s(a)$ (*ie* the observed state-action distribution). This leads to following result:

Theorem* Let us consider the update rule specified in equation 1 for some $\tilde{A}$ and some small α. Let A be the true advantages. Then the performance improvement we get is:

$$\Delta\tilde{\eta} = \alpha < A, \tilde{A} >_\nu + O(\alpha^2) \,. \tag{2}$$

Thus, to obtain improvement all we need is the much weaker result that $\tilde{A}$ must have a positive inner product with A (as opposed to strong and intractable conditions with the l_∞ norm).

Proof First, let us consider the table-lookup Gibbs distribution, $\tilde{\Pi}_G = \{\pi_\theta : \pi(s,a) \propto \exp(\theta_{sa})\}$. The exact gradient of the average reward (for any parameterized policy) is:

$$\nabla\eta^\pi = \sum_{s,a} \rho^\pi(s) \nabla\pi(s,a) Q^\pi(s,a). \tag{3}$$

Using the notation $\partial_{sa} \equiv \partial/\partial\theta_{sa}$, it is straightforward to show for $\tilde{\Pi}_G$, since $\partial_{sa}\pi_{s'a'} = \delta_{ss'}\pi(s,a)(\delta_{aa'} - \pi(s,a'))$:

$$\begin{aligned}
\partial_{sa}\eta^\pi &= \sum_a \rho(s)\pi(s,a)(\delta_{aa'} - \pi(s,a'))Q(s,a) && (4)\\
&= \rho^\pi(s)\pi(s,a)(Q^\pi(s,a) - < \tilde{Q}(s,a') >_{\pi(s,a')}) && (5)\\
&= \rho^\pi(s)\pi(s,a)A^\pi(s,a) && (6)
\end{aligned}$$

. For some change $\Delta\theta_{sa}$ to the parameters, the update rule is:

$$\pi_s(a, \theta_{t+1}) \leftarrow \pi_s(a; \theta_t) \exp(\Delta\theta_{sa})$$

, the resulting change to the performance is $\Delta\eta = \sum_{sa} \partial_{sa}\eta\Delta\theta_{sa} = \sum_{sa} \rho(s)\pi_s(a)A(s,a)\Delta\theta_{sa}$ (for small $\Delta\theta$).

The update rule for the table-lookup Gibbs is exactly the same as that used in equation 1, so we evaluate the change to the performance

$$\Delta\eta = \sum_{sa} \rho(s)\pi_s(a)A(s,a)(\alpha\tilde{A}_{sa}) =< A, \tilde{A} >_\nu$$

. **q.e.d.**

4 Issues

In essence, this method can be seen as trying to find the gradient over the space of *all* policies.

Neglecting the scale of $\tilde{A}$ it is clear that the largest local improvement is when $\tilde{A}$ is aligned with A. This is precisely what the l_2 norm is trying to fit (the l_2 error is $< A-\tilde{A}, A-\tilde{A} >_\nu$). However, this scale is the real problem!!! Or put another way, what determines the best direction? It is the one which can be followed the furthest in a line search. We would like to find the $\tilde{A}$ that allows the largest improvement after a line search is done. However, trying to analyse this involves looking at the curvature. Ack! I have been trying to do this with little success. Essentially it is the $O(\alpha^2)$ term in equation 2 that determines what will be the best fit! As it stands, we want an infinite $\tilde{A}$ must be prevented.

Another way to look at the whole argument (perhaps simpler) is let us just start with the set of parameterized policy $\{\pi_\theta : \pi(s,a) \propto \exp(\theta_{sa})\}$. The gradient is $\partial_{sa}\eta = \rho(s)\pi(s,a)A(s,a)$. If we change the parameters by $\Delta\theta_{sa}$, then the resulting performance change will be $\Delta\eta =< A, \Delta\theta >_\nu$ (for small $\Delta\theta$). Now, if we constrain $< \Delta\theta, \Delta\theta >_\nu$ to be constant (*ie* ignore the magnitude of this vector), then the l_2 norm is fine to use. However, this essentially boils down to the question of what the best direction is (out of all of those that improve).

5 Regardless

The last argument essentially shows that we have a method to find the gradient over the space of all policies (it's actually the natural gradient that we are finding due to update rule of equation 1). As with all gradient methods, there is an issue as to how to actually find the precise direction to travel in (which I don't have good answer). With non-parametric methods, this method can try to find this gradient regardless of how much data is observed (with more data giving a more accurate gradient). This is an interesting way of looking at things, *ie* that we are trying to generalize from the observed data to find the gradient over the space of all policies. Thus we shouldn't need to worry about local minima issues if asymptotically we collect more data between gradient updates. I have not though about how to combine these ideas with conjugate ideas (which could address this issue of which direction to do a line search in).

Also, up till now I have assumed that the performance measure we are interested in is the average reward. However, the stationary distribution seems to be the wrong distribution and could get us trapped. A more uniform distribution is necessary, but the results here still hold under a different distribution.