Statistical Linearization for Value Function Approximation in Reinforcement Learning

Matthieu Geist
Supélec, IMS Research Group
Campus of Metz, France
`matthieu.geist@supelec.fr`

Abstract

Reinforcement learning (RL) is a machine learning answer to the optimal control problem. It consists in learning an optimal control policy through interactions with the system to be controlled, the quality of this policy being quantified by the so-called value function. An important RL subtopic is to approximate this function when the system is too large for an exact representation. This paper presents statistical-linearization-based approaches to estimate such functions. Compared to more classical approaches, this allows considering nonlinear parameterizations as well as the Bellman optimality operator, which induces some differentiability problems. Moreover, the statistical point of view adopted here allows considering colored observation noise models instead of the classical white one; in RL, this can provide useful.

1 Introduction

Optimal control of stochastic dynamic systems is a trend of research with a long history. Several points of view can be adopted according to the information available on the system such as a model of the physics ruling the system (automation) or a stochastic model of its dynamic (dynamic programming). The machine learning response to this recurrent problem is the Reinforcement Learning (RL) paradigm [1, 2, 3].

The system to be controlled is usually modeled as a Markov Decision Process (MDP) $\{S, A, P, R, \gamma\}$ [4]. An MDP is made up of a set S of states (the different configurations of the system), a set A of actions (which cause a change of the system's state), a set $P : s, a \in S \times A \rightarrow p(.|s, a) \in \mathcal{P}(S)$ of Markovian transition probabilities (the probability to transit from one state to another under a given action), a reward function $R : s, a, s' \in S \times A \times S \rightarrow r = R(s, a, s') \in \mathbb{R}$ associating a scalar to each transition and a discounting factor γ which decreases long-term rewards' influence. How the agent acts with the system is modeled by a so-called policy, $\pi : s \in S \rightarrow \pi(.|s) \in \mathcal{P}(A)$, which associates to each state a probability distribution over actions. The quality of such a policy is quantified by a so-called value function, $V^\pi(s) = E[\sum_{i=0}^\infty \gamma^i r_i | s_0 = s, \pi]$, which associates to each state the expected cumulative discounted reward from starting in the considered state and then following the given policy. An optimal policy is one of those which maximize the associated value function for each state: $\pi^* \in \operatorname{argmax}_\pi V^\pi$.

If the model (that is transition probabilities and the reward function) is known and if state and action spaces are small enough, the optimal policy can be computed using dynamic programming. A first scheme, called value iteration, consists in computing directly the optimal value function (using the nonlinear Bellman optimality equation -see later- and an iterative scheme based on the fact that the value function is the unique fixed-point of the associated Bellman operator). The optimal policy is greedy respectively to the optimal value function. A second scheme, called policy iteration, consists in evaluating an initial policy and then improving it, the new policy being greedy respectively to the computed value function. Evaluation and improvement are iterated until convergence.

RL aims at estimating the optimal policy without knowing the model and from interactions with the system. Value functions can no longer be computed, they have to be estimated. RL heavily relies on dynamic programming, in the sense that most of approaches are some sort of generalizations of value or policy iteration. A first problem is that computing a greedy policy (required for both schemes) from a value function requires the model to be known. The state-action value (or Q-) function alleviates this problem by providing an additional degree of freedom on the first action to be chosen: $Q^\pi(s,a) = E[\sum_{i=0}^{\infty} \gamma^i r_i | s_0 = s, a_0 = a, \pi]$. A greedy policy can thus be obtained by maximizing the Q-function over actions. The state-action value function generalizes the value function in the sense that $V^\pi(s) = E_{a|s,\pi}[Q^\pi(s,a)]$. Therefore, the rest of this paper focuses uniquely on the Q-function.

There are two main approaches to estimate an optimal policy. The first one, based on value iteration, consists in estimating directly the optimal state-action value function which is then used to derive an estimate of the optimal policy. The second one, based on policy iteration, consists in mixing the estimation of the Q-function of the current policy (policy evaluation) with policy improvement in a generalized policy iteration scheme (generalized in the sense that evaluation and improvement processes interact, independently of the granularity and other details). This scheme presents many variations. Generally, the Q-function is not perfectly estimated when the improvement step occurs. Each change in the policy implies a change in the associated Q-function; therefore, the estimation process can be non-stationary. The policy can be derived from the estimated state-action value function (for example, using a Boltzmann distribution or an ϵ-greedy policy): this implies an underlying dilemma between exploration and exploitation. The policy can also have its own representation, which leads to actor-critic architectures.

A common subproblem of these approaches is to estimate the (state-action) value function from a set of transitions. Basically, there are three ways to do it (bootstrapping, residual approach and projected fixed-point method [5]), each one corresponding to a particular optimization problem to be solved. If the parameterization is linear and if the Bellman evaluation operator is considered, everything is linear and these optimization problems can be solved easily. If not, corresponding optimization problems can be nonlinear (parameterization) and even non-differentiable (Bellman optimality operator, see Sec. 2.1). We advocate that they can be linearized using a derivative-free statistical linearization approach and then solved analytically. This way, classical RL algorithms can be easily extended to a nonlinear setting. This paper synthesizes recent works in combining statistical linearization with value function approximation [6, 7, 8].

2 Background

2.1 Value Function Approximation

Thanks to the Markov property of the transition probabilities, the state-action value function Q^π of a given policy π satisfies the linear Bellman evaluation equation: $Q^\pi(s,a) = E_{s',a'|s,a,\pi}[R(s,a,s') + \gamma Q^\pi(s',a')]$. A Bellman evaluation operator related to the Q-function, $T^\pi : Q \in \mathbb{R}^{S\times A} \to T^\pi Q \in \mathbb{R}^{S\times A}$, can be defined: $T^\pi Q(s,a) = E_{s',a'|s,\pi}[R(s,a,s') + \gamma Q(s',a')]$. This operator is a contraction and Q^π is its unique fixed-point: $Q^\pi = T^\pi Q^\pi$. Using it is not practical, as it requires knowing the model. Therefore, a sampled Bellman evaluation is introduced. For a transition $(s_i, a_i, s_{i+1}, a_{i+1})$, a_{i+1} being sampled according to π, and the associated reward r_i, it is defined as (defining the $\hat{P}^\pi$ operator on the fly):

$$\hat{T}^\pi Q(s_i,a_i) = r_i + \gamma Q(s_{i+1},a_{i+1}) = r_i + \gamma \hat{P}^\pi Q(s_i,a_i). \quad (1)$$

The optimal state-action value function Q^* satisfies the nonlinear Bellman optimality equation: $Q^*(s,a) = E_{s'|s,a}[R(s,a,s') + \gamma \max_{a'\in A} Q^*(s',a')]$.. The associated Bellman optimality operator, $T^* : Q \in \mathbb{R}^{S\times A} \to T^* Q \in \mathbb{R}^{S\times A}$, is defined as: $T^*Q(s,a) =$

$E_{s'|s,a}[R(s,a,s') + \gamma \max_{a' \in A} Q(s',a')]$. A more practical sampled Bellman optimality operator is also defined. Assume that a transition (s_i, a_i, s_{i+1}) and associated reward r_i are observed, it is given by (defining the $\hat{P}^*$ operator on the fly):

$$\hat{T}^* Q(s_i, a_i) = r_i + \gamma \max_{a \in A} Q(s_{i+1}, a) = r_i + \gamma \hat{P}^* Q(s_i, a_i). \tag{2}$$

It allows estimating directly the optimal policy from trajectories generated using a suboptimal policy, that is off-policy learning.

An important RL subtopic is to estimate the (state-action) value function of a given policy or directly the Q-function of the optimal policy from samples, that is observed trajectories of actual interactions. This paper focuses on parametric approximation: the estimated state-action value function is of the form $\hat{Q}_\theta$, where θ is the parameter vector; this estimate belongs to an hypothesis space $\mathcal{H} = \{\hat{Q}_\theta | \theta \in \mathbb{R}^p\}$ which specifies the architecture of the approximation. For example, if the state space is sufficiently small an exact tabular representation can be chosen for the value function. The estimate is thus of the form $\hat{Q}_\theta(s,a) = e_{s,a}^T \theta$ with $e_{s,a}$ being a unitary vector which is equal to one in the component corresponding to the state-action couple (s,a) and zero elsewhere. More complex hypothesis spaces can be envisioned, such as those generated by MLPs (multilayered perceptrons). Estimating a function from samples is a common topic of supervised learning. However, estimating a (state-action) value function is a more difficult problem because values are never directly observed, just rewards which define them. Actually, value function approximation consists in estimating a fixed-point of the T operator (either T^π or T^*) from sampled trajectories (that is using the sampled $\hat{T}$ operator and a finite number of state-action couples): $\hat{Q}_\theta \approx T\hat{Q}_\theta$.

2.2 Statistical Linearization

Let $f : \mathbf{x} \in \mathbb{R}^p \to \mathbf{y} = f(\mathbf{x}) \in \mathbb{R}^q$. Assume that we want to linearize f around some point $\mathbf{x}_0$. A common approach is to use a Taylor expansion: $f(\mathbf{x}) \approx f(\mathbf{x}_0) + \nabla f(\mathbf{x}_0)(\mathbf{x} - \mathbf{x}_0)$. However, it requires f to admit a gradient. Statistical linearization [9] consists in sampling a set of m points around $\mathbf{x}_0$ (how to sample these points being the user's choice, done in subsequent practical algorithms) and using them to compute a linear model minimizing the associated square error.

Let $(\mathbf{x}^{(j)}, \mathbf{y}^{(j)} = f(\mathbf{x}^{(j)}))_{1 \leq j \leq m}$ be a set of m sampled points and their images through f. We introduce some notations:

$$\begin{cases} \bar{\mathbf{x}} = \frac{1}{m}\sum_{j=1}^m \mathbf{x}^{(j)}, \quad \bar{\mathbf{y}} = \frac{1}{m}\sum_{j=1}^m \mathbf{y}^{(j)}, \\ P_{\mathbf{xx}} = \frac{1}{m}\sum_{j=1}^m \left(\mathbf{x}^{(j)} - \bar{\mathbf{x}}\right)\left(\mathbf{x}^{(j)} - \bar{\mathbf{x}}\right)^T, \quad P_{\mathbf{yy}} = \frac{1}{m}\sum_{j=1}^m \left(\mathbf{y}^{(j)} - \bar{\mathbf{y}}\right)\left(\mathbf{y}^{(j)} - \bar{\mathbf{y}}\right)^T \\ P_{\mathbf{xy}} = \frac{1}{m}\sum_{j=1}^m \left(\mathbf{x}^{(j)} - \bar{\mathbf{x}}\right)\left(\mathbf{y}^{(j)} - \bar{\mathbf{y}}\right)^T = P_{\mathbf{yx}}^T \end{cases} \tag{3}$$

As these points are sampled around $\mathbf{x}_0$, $\bar{\mathbf{x}} = \mathbf{x}_0$. Statistical linearization performs a linear regression of the form $\mathbf{y} = A^T\mathbf{x} + b + e$ where e is a centered noise and A and b minimize the sum of square errors: $\min_{A,b} \sum_{j=1}^m e_j^T e_j$ with $e_j = \mathbf{y}^{(j)} - A^T\mathbf{x}^{(j)} - b$. The solution to this optimization problem is given by: $A = P_{\mathbf{xx}}^{-1} P_{\mathbf{xy}}$ and $b = \bar{\mathbf{y}} - A^T\bar{\mathbf{x}}$. The error variance is therefore given by: $P_{ee} = \sum_{j=1}^m e_j e_j^T = P_{\mathbf{yy}} - A^T P_{\mathbf{xx}} A$. Notice that the noise term e quantifies the linearization error.

3 Statistical Linearization for Reinforcement Learning

One of the oldest approaches for value function approximation is bootstrapping [2]. The idea is to treat the corresponding estimation problem as a classical regression, assuming that the Q-values are observable (noted q_j for the state-action couple (s_j, a_j)). The cost function to be minimized is therefore $\|Q - \hat{Q}_\theta\|$ with Q being either Q^π for a given policy or Q^*. For a finite number of samples, the estimate is therefore $\theta_i = \operatorname{argmin}_{\omega \in \mathbb{R}^p} \sum_{j=1}^i (q_j - \hat{Q}_\omega(s_j, a_j))^2$. This optimization problem is solved in order to provide a recursive estimate, using either a stochastic gradient descent (TD, SARSA and Q-learning [2]) or a linear recursive least-squares (the Fixed-point Kalman Filter [10], assuming a linear parameterization). The parameter vector is thus updated according to a Widrow-Hoff update (K_i being an algorithm-dependent gain): $\theta_i = \theta_{i-1} + K_i(q_j - \hat{Q}_{\theta_{i-1}}(s_i, a_i))$. However, q_j is never observed. That is where the bootstrap principle applies: the unobserved value q_j is replaced by the estimate $\hat{T}\hat{Q}_{\theta_{i-1}}(s_i, a_i)$, with $\hat{T} = \hat{T}^{\pi}$ or $\hat{T} = \hat{T}^{*}$, depending if one wants to estimate the Q-function of the given policy π or the optimal one directly.

Thanks to the bootstrap principle, many online regression algorithms developed in the supervised learning field can be adapted to reinforcement learning. However, TD with function approximation cannot be guaranteed to be convergent with a nonlinear parameterization [11], even if one of the practical successes of RL is based on a neural network [12]. Therefore, there is a common belief in the RL community that it is dangerous to combine RL algorithms with nonlinear function approximators such as MLPs. However, the problem comes from the bootstrap principle, and there are other approaches for estimating the Q-function. Moreover, they can handle nonlinearities if combined with statistical linearization.

3.1 Kalman Temporal Differences (KTD)

Residual approaches aim at minimizing the distance between the Q-function and its image through a Bellman operator, $\|\hat{Q}_\theta - T\hat{Q}_\theta\|^2$. Practically, learning is done from samples and the sampled Bellman operator is used instead (the model being not known): $\theta_i = \operatorname{argmin}_{\omega \in \mathbb{R}^p} \sum_{j=1}^{i} (\hat{Q}_\omega(s_j, a_j) - \hat{T}\hat{Q}_\omega(s_j, a_j))^2$. If this optimization problem is solved using a stochastic gradient descent (assuming the sampled Bellman evaluation operator for differentiability), this provides the so-called residual algorithm [13]. If in addition a linear parameterization is assumed and this is solved using a (linear) recursive least-squares (RLS) approach, it is the parametric Gaussian Process Temporal Differences (GPTD) algorithm [14]. Notice that in both cases the Bellman optimality operator cannot be considered because it is generally non-linear and non-differentiable (because $\max_{a \in A} \hat{Q}_\omega(s, a)$ is generally non differentiable respectively to ω).

The following statistical observation model can be associated to the previous empirical cost-function:

$$r_i = \hat{Q}_\omega(s_i, a_i) - \gamma \hat{P}\hat{Q}_\omega(s_i, a_i) + n_i, \tag{4}$$

where n_i is a unitary white noise and $\hat{P}$ is either $\hat{P}^{\pi}$ or $\hat{P}^{*}$. The statistical linearization principle described in Sec. 2.2 can be applied to this observation model (linearization respectively to the parameter vector ω): $r_i = A_i^T \omega + b_i + u_i$, where $u_i = e_i + n_i$ is a white noise (of variance P_{u_i}) which takes into account the statistical linearization error e_i. This linearized observation model can be used to construct the alternative following optimization problem: $\theta_i = \operatorname{argmin}_{\omega \in \mathbb{R}^p} \sum_{j=1}^{i} \frac{1}{P_{u_i}} (r_i - A_i^T \omega - b_i)^2$. A linear RLS approach can be used to solve it, which provides a recursive estimate. It is also required to determine around what and with what spread one should linearize. In a recursive estimation process, the statistical linearization is performed around the last estimate and with a spread quantified by the variance matrix maintained by the RLS approach. Moreover, this statistical linearization is performed using the unscented transform [15]. This provide the so-called Kalman Temporal Differences (KTD) framework, which can be derived using a Kalman-based approach [6]. It can also be derived from the preceding statistically linearized RLS approach [5]. This link between unscented Kalman filtering [15] and statistically linearized RLS is studied in [16].

3.2 KTD(λ)

In supervised learning, most often a white observation noise is assumed. However, it can be of interest to color it, notably for residual approaches. Due to the use of the sampled Bellman operator, residual approaches produce biased estimate [17]. This is due to the fact that the mean of a square is not the square of the mean: $E[\|\hat{T}\hat{Q}_\omega - \hat{Q}_\omega\|^2] = \|T\hat{Q}_\omega - \hat{Q}_\omega\|^2 + \operatorname{Var}(\hat{T}\hat{Q}_\omega)$. Thanks to the statistical point of view, the optimization problem solved by residual approaches can be linked to the observation model (4). It can be shown that the bias problem can be linked to the fact that the noise n_i is assumed white [14, 6]. Using a colored noise instead can reduce and even remove this bias [14, 6, 8]. The more general colored noise model is $n^\lambda = u_i - \gamma\lambda b_{i-1}^\lambda$ with u_i being a white noise and $b_i^\lambda = \gamma(1-\lambda)b_{i-1}^\lambda + u_i$ being an autoregressive noise (this noise, introduced in [8], generalizes the white noise case with $\lambda = 0$ and noises introduced in [14, 6] with $\lambda = 1$). With a colored observation noise instead of the white one, the optimization problem to be solved differs. However, a recursive estimate can be provided using ideas underlying Bayesian and Kalman filtering [8].

3.3 Statistically Linearized Least-Squares Temporal Differences

Another way to estimate a state-action value function is the projected fixed-point approach. The image of the Q-function through a Bellman operator does not necessarily lies into the hypothesis space $\mathcal{H}$. It can therefore be projected back onto it. Projected fixed-point approaches minimize the distance between the Q-function and the projection onto $\mathcal{H}$ of its image through T. The corresponding cost-function is $\|\hat{Q}_\theta - \Pi T\hat{Q}_\theta\|^2$, where Π is the projection operator: $\Pi f = \operatorname{argmin}_{\hat{f}\in\mathcal{H}} \|f - \hat{f}\|^2$. This defines two nested optimization problems, and the corresponding empirical optimization problem is $\theta_i = \operatorname{argmin}_{\omega\in\mathbb{R}^p} \sum_{j=1}^i (\hat{T}\hat{Q}_{\theta_i}(s_j,a_j) - \hat{Q}_\omega(s_j,a_j))^2$. Notice that θ_i appears in both sides of this equation: this translates the two nested optimization problems, and therefore it is not a pure least-squares problem. Recently, this optimization problem has been solved using a stochastic gradient descent [18] (which is far from being direct because of the particular considered optimization problem). The corresponding algorithm is named Greedy-GQ. Assuming a linear parameterization and the Bellman evaluation operator, it can be solved analytically using a RLS-like approach, this is the Least-Squares Temporal Differences (LSTD) algorithm [19].

As for residual approaches, an observation model can be derived from this optimization problem:

$$r_j = \hat{Q}_\omega(s_j,a_j) - \gamma\hat{P}\hat{Q}_{\theta_i}(s_j,a_j) + n_j \tag{5}$$

Here again, the statistical linearization principle can be applied. However, two statistical linearizations should be applied, one for the θ_i parameter vector and one for the ω parameter vector. This gives an observation model of the following form: $r_i = A_i^T\theta_i + B_i^T\omega + c_i + u_i$, with u_i being the white noise term comprising the linearization error. From this linearized observation model, an alternative linear optimization problem can be written and analytically solved: $\theta_i = \operatorname{argmin}_{\omega\in\mathbb{R}^p} \sum_{j=1}^i \frac{1}{P_{u_i}}(r_i - A_i^T\theta_i - B_i^T\omega - c_i)^2$. Once again, the linearization error is taken into account, which is an interesting feature: the higher the linearization error, the less the corresponding sample is taken into account. This provides the statistically linearized LSTD (slLSTD) algorithm [7]. The only other approach able to handle both a nonlinear parameterization and the Bellman optimality operator is the Greedy-GQ algorithm. Contrary to it, slLSTD has no convergence analysis for now, but it is empirically more efficient [7].

4 Conclusion and perspectives

State-action value function approximation is an important subtopic of reinforcement learning. Basically, there are three ways to estimate a Q-function: bootstrapping, residual methods and projected fixed-point approaches [5]. To each one corresponds an optimization problem to be solved. If the parameterization is linear and if the (linear) Bellman evaluation operator is considered, there exists analytical solutions. If the parameterization is nonlinear, or worse if the Bellman optimality operator is considered (which implies differentiability problems), things become more complicated. Bootstrapping combined with nonlinear parameterizations can be unstable or even diverge [11]. Considering residual and projected fixed-point approaches, a related observation model can be derived. We advocate that applying statistical linearization to them (which is more or less direct) and minimizing the associated linearized cost-function (with the possible addition of a colored observation noise model) is an interesting way to handle the tricky underlying optimization problems. Thanks to its derivative-free aspect, it does not require to compute gradients, which proves really useful in RL (optimality operator). Statistical linearization leads to RLS-like approaches, so related algorithms are second order and tend to be efficient. Moreover, a variance matrix is usually maintained in the process; this provides uncertainty information which is useful for active learning or for the dilemma between exploration and exploitation [20]. These approaches show good empirical results but they lack some theoretical analysis. Nevertheless, we think that this statistical linearization idea can be applied to broader fields, not only to reinforcement learning.

References

[1] D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming (Optimization and Neural Computation Series, 3)*. Athena Scientific, 1996.

[2] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 3rd edition, March 1998.

[3] O. Sigaud and O. Buffet, editors. *Markov Decision Processes and Artificial Intelligence*. Wiley - ISTE, 2010.

[4] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley-Interscience, 1994.

[5] M. Geist and O. Pietquin. A Brief Survey of Parametric Value Function Approximation. Technical report, Supélec, september 2010.

[6] M. Geist and O. Pietquin. Kalman Temporal Differences. *Journal of Artificial Intelligence Research (JAIR)*, 39:489–532, 2010.

[7] M. Geist and O. Pietquin. Statistically Linearized Least-Squares Temporal Differences. In *Proceedings of the IEEE International Conference on Ultra Modern Telecommunications and Control Systems (ICUMT 2010)*, Moscow (Russia), 2010. IEEE.

[8] M. Geist and O. Pietquin. Eligibility Traces through Colored Noises. In *Proceedings of the IEEE International Conference on Ultra Modern Telecommunications and Control Systems (ICUMT 2010)*, Moscow (Russia), 2010. IEEE.

[9] T. W. Anderson. *An Introduction to Multivariate Statistical Analysis*. Wiley, 1984.

[10] D. Choi and B. Van Roy. A Generalized Kalman Filter for Fixed Point Approximation and Efficient Temporal-Difference Learning. *Discrete Event Dynamic Systems*, 16:207–239, 2006.

[11] J. N. Tsitsiklis and B. Van Roy. An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control*, 42, 1997.

[12] G. Tesauro. Temporal Difference Learning and TD-Gammon. *Communications of the ACM*, 38(3), March 1995.

[13] L. C. Baird. Residual Algorithms: Reinforcement Learning with Function Approximation. In *Proceedings of the International Conference on Machine Learning (ICML 95)*, pages 30–37, 1995.

[14] Y. Engel. *Algorithms and Representations for Reinforcement Learning*. PhD thesis, Hebrew University, April 2005.

[15] S. J. Julier and J. K. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422, 2004.

[16] M. Geist and O. Pietquin. Statistically Linearized Recursive Least Squares. In *Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing (MLSP 2010)*, Kittilä (Finland), 2010. IEEE.

[17] A. Antos, C. Szepesvári, and R. Munos. Learning near-optimal policies with Bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning*, 71(1):89–129, 2008.

[18] Hamid Reza Maei, Csaba Szepesvari, Shalabh Bhatnagar, and Richard S. Sutton. Toward Off-Policy Learning Control with Function Approximation. In *27th conference on Machine Learning (ICML 2010)*, 2010.

[19] S. J. Bradtke and A. G. Barto. Linear Least-Squares algorithms for temporal difference learning. *Machine Learning*, 22(1-3):33–57, 1996.

[20] Matthieu Geist and Olivier Pietquin. Managing Uncertainty within the KTD Framework. In *Workshop on Active Learning and Experimental Design*, Journal of Machine Learning Research Conference and Workshop Proceedings, 2010. to appear.