Softened Approximate Policy Iteration for Markov Games

Julien Pérolat[(a)] JULIEN.PEROLAT@ED.UNIV-LILLE1.FR
Bilal Piot[(a)] BILAL.PIOT@UNIV-LILLE1.FR
Matthieu Geist[(d)] MATTHIEU.GEIST@CENTRALESUPELEC.FR
Bruno Scherrer[(b)] BRUNO.SCHERRER@INRIA.FR
Olivier Pietquin[1] [(a,c)] OLIVIER.PIETQUIN@UNIV-LILLE1.FR

[(a)]Univ. Lille, CNRS, Centrale Lille, Inria UMR 9189 - CRIStAL, F-59000 Lille, France
[(b)]Inria, Villers-lès-Nancy, F-54600, France
[(c)]Institut Universitaire de France (IUF), France
[(d)] UMI 2958, GeorgiaTech-CNRS, CentraleSupélec, Université Paris-Saclay, Metz, France,

Abstract

This paper reports theoretical and empirical investigations on the use of quasi-Newton methods to minimize the Optimal Bellman Residual (OBR) of zero-sum two-player Markov Games. First, it reveals that state-of-the-art algorithms can be derived by the direct application of Newton's method to different norms of the OBR. More precisely, when applied to the norm of the OBR, Newton's method results in the Bellman Residual Minimization Policy Iteration (BRMPI) and, when applied to the norm of the Projected OBR (POBR), it results into the standard Least Squares Policy Iteration (LSPI) algorithm. Consequently, new algorithms are proposed, making use of quasi-Newton methods to minimize the OBR and the POBR so as to take benefit of enhanced empirical performances at low cost. Indeed, using a quasi-Newton method approach introduces slight modifications in term of coding of LSPI and BRMPI but improves significantly both the stability and the performance of those algorithms. These phenomena are illustrated on an experiment conducted on artificially constructed games called Garnets.

1. Introduction

A two-player zero-sum Markov Game (MG) is a model for adversarial interaction between two agents. Players are considered as agents stepping from state to state and collecting rewards in consequence of their mutual actions. In the zero-sum setting, the reward of one player is the loss of the other. Chess, Checkers, multi-agent systems can be modeled as an MG. Notice that MGs are natural extensions of Markov Decision Processes (MDP) and thus any MDP can be modeled as an MG (results thus also hold for MDPs). In this paper, we are interested in finding a minmax strategy of interaction for both players. Finding minmax strategies in small MGs is possible with exact Dynamic Programming (DP) algorithms. However, when the state space is too large or when the game is only known through logs of interactions, one needs to use Approximate DP (ADP). This paper focuses on ADP techniques with linear function approximation.

MGs were first introduced by Shapley (1953) (although MGs are referred to as Stochastic Games) who provided the first algorithm that solves zero-sum two-player MGs. This algorithm is analogous to Value Iteration (VI), very popular in the MDP literature. Following this, two extensions of Howard's Policy Iteration (PI) (Puterman, 1994) to zero-sum two-player MGs were developed. First, the Hoffman and Karp (1966) algorithm that requires solving an MDP as a subroutine at each iteration and is proven to converge. Second, the Pollatschek and Avi-Itzhak (1969) algorithm performs well in practice but is not proven to converge in general. Actually, Van Der Wal (1978) published a simple counterexample showing that this algorithm is unstable. To our knowledge, finding necessary and sufficient conditions for its convergence is still an open problem. However, in their enlightening paper, Filar & Tolwinski (1991) introduce a slight modification of the Pollatschek & Avi-Itzhak algorithm aimed at fixing its convergence issues. In fact, the Pollatschek & Avi-Itzhak algorithm is a Newton's method on the $\mathcal{L}_2$-norm of the Optimal Bellman Residual (OBR). The modification of Filar & Tolwinski amounts to use a quasi-Newton method on the $\mathcal{L}_2$-norm of the OBR in-

Proceedings of the 33rd International Conference on Machine Learning, New York, NY, USA, 2016. JMLR: W&CP volume 48. Copyright 2016 by the author(s).

[1]Now with Google DeepMind

stead of a Newton's Method. However, their proof of convergence is based on the assumption that the $\mathcal{L}_2$-norm of the OBR is smooth, which is untrue in general (as we show at the end of Section 2). Indeed, the derivative of the $\mathcal{L}_2$-norm of the OBR might be discontinuous and thus quasi-Newton method is not anymore guaranteed to converge to a local minimum (counterexamples exist (Lewis & Overton, 2013)). But, in practice (Lewis & Overton, 2013), using quasi-Newton methods often leads to pretty good solutions even in the non-smooth case. The major contribution of this paper is thus to explore the use of quasi-Newton methods to improve the stability of ADP methods with linear function approximations.

Adapting the Hoffman & Karp algorithm, belonging to the PI-like algorithms, with function approximation is cumbersome but feasible (Perolat et al., 2015). However, Pollatschek & Avi-Itzhak-like algorithms for MGs with function approximation have been the topic of very little attention except from Lagoudakis & Parr (2002). In that paper, the authors adapt the Least Squares Policy Iteration (LSPI) algorithm to zero-sum two-player MGs. When the dynamics is known and the feature space is the identity matrix, this algorithm is close to the Pollatschek & Avi-Itzhak one. But, in the exact case, Van Der Wal's example applies, and the algorithm suffers at least the same drawbacks as the Pollatschek & Avi-Itzhak one even if one uses stable value function approximations as suggested in (Lagoudakis & Parr, 2002) (see Section 3). Our strategy to adapt PI-like algorithms to linear value functions is somehow different from Lagoudakis & Parr. Indeed, in the exact case, the objective is to minimize the $\mathcal{L}_2$-norm of the OBR. With linear function approximation, two objectives used in the MDP literature (Lagoudakis & Parr, 2003), can canonically be minimized: the norm of the OBR and the norm of the Projected OPR (POBR) (the projection is made on a feature space). We thus propose to study the use of quasi-Newton method on those residuals and on empirical estimates as Filar & Tolwinski did for the exact case.

As a first set of contributions, we shall show that a Newton's method applied to the POBR leads to LSPI (both for games and for MDPs), while a Newton's method applied to the OBR of an MDP is known as Bellman Residual Minimization Policy Iteration (BRMPI) (see Section 3.1). Furthermore, we shall show that Newton's method on some empirical estimates of the POBR leads to LSPI with batch data (Section 4.1). Another set of contributions is to generalize these ideas and propose batch algorithms for MGs (and MDPs) applying quasi-Newton method (thus using a learning rate) to empirical estimates of the OBR and POBR (Section 4.2 and 4.1). We shall then study the use of such a learning rate for zero-sum two-player MGs and for MDPs as a subclass of zero-sum two-player MGs (Sections 5 and 6). The benefit of using quasi-Newton method instead of Newton's method will be demonstrated on synthetic problems, namely Garnets, in Section 6.

2. Background

A two-player zero-sum MG is usually described as a tuple $< S, (A^i(s))_{s\in S, i=1,2}, r(s,a^1,a^2), p(.|s,a^1,a^2), \gamma >$ where S is a finite state space, $A^i(s)$ is a finite action space used by player $i \in \{1,2\}$ in state s. The reward $r(s,a^1,a^2)$ is the local benefit both players will collect by doing the joint action (a^1,a^2) in state s. The transition kernel $p(.|s,a^1,a^2)$ captures the dynamics of the game from s when both players play the joint action (a^1,a^2). Finally, $\gamma \in [0,1[$ is the discount factor.

We are interested in finding the best strategy for each player. Player 1 will play strategy $\mu(a^1|s)$ (respectively $\nu(a^2|s)$ for player 2) which is a mapping from the state space S to the set of distributions over the action space $A^1(s)$ (respectively $A^2(s)$). Let us write $r_{\mu,\nu}(s) = E_{a^1\sim\mu(.|s),a^2\sim\nu(.|s)}[r(s,a^1,a^2)]$ the expected average reward players will collect when the joint strategy (μ,ν) is played in state s. Let us write $P_{\mu,\nu}(s'|s) = E_{a^1\sim\mu(.|s),a^2\sim\nu(.|s)}[p(s'|s,a^1,a^2)]$ the corresponding kernel. Both players will try to optimize the hereinafter defined value function:

$$v_{\mu,\nu}(s) = \sum_{k=0}^{\infty} \gamma^k E[r_{\mu,\nu}(s_k)|s_0 = s, s_{k+1} \sim P_{\mu,\nu}(s_{k+1}|s_k)].$$

The value function $v_{\mu,\nu} \in \mathbb{R}^S$ is the γ-discounted sum of rewards when the joint strategy (μ,ν) is played. Player 1's goal is to maximize the above-mentioned value function while player 2's goal is to minimize it. We can introduce the following five Bellman operators:

$$\mathcal{T}_{\mu,\nu}v\ (s) = r_{\mu,\nu}(s) + \sum_{s'\in S} P_{\mu,\nu}(s'|s)v(s'),$$
$$\mathcal{T}_{\mu}v = \min_{\nu} \mathcal{T}_{\mu,\nu}v, \quad \tilde{\mathcal{T}}_{\nu}v = \max_{\mu} \mathcal{T}_{\mu,\nu}v,$$
$$\mathcal{T}^*v = \max_{\mu}\min_{\nu} \mathcal{T}_{\mu,\nu}v, \quad \tilde{\mathcal{T}}^*v = \min_{\nu}\max_{\mu} \mathcal{T}_{\mu,\nu}v.$$

All these operators are γ-contractions in $\mathcal{L}_\infty$ norm (Puterman, 1994). One should notice that the fixed point of operator $\mathcal{T}_{\mu,\nu}$ is the value function $v_{\mu,\nu}$. The fixed point of $\mathcal{T}_\mu$ is $v_\mu = \min_\nu v_{\mu,\nu}$ which is the value of a best response of player 2 against strategy μ. The dual of this value is the fixed point of operator $\tilde{\mathcal{T}}_\nu$ and is the value $\tilde{v}_\nu$. Operators $\mathcal{T}^*$ and $\tilde{\mathcal{T}}^*$ are equal according to the minimax theorem (Von Neumann, 1947). The fixed point of $\mathcal{T}^*$, written v^*, is called the optimal value of the game. The reader should notice that, when for all s in S $card(A^2(s)) = 1$, the MG is reduced to an MDP.

Let us first describe Newton's and quasi-Newton methods. Newton's method is an optimization technique aiming at minimizing a function $f : \mathcal{R}^n \to \mathcal{R}$. Starting from $x_0 \in \mathbb{R}^n$, it computes the sequence $(x_n)_{n\in\mathbb{N}}$ such that:

$$x_{n+1} = x_n - [Hf(x_n)]^{-1}\nabla f(x_n),$$

where $Hf(x_n)$ and $\nabla f(x_n)$ are respectively the Hessian matrix and the gradient in x_n. This method might be unstable (Nocedal & Wright, 2006) and one way to soften it is to introduce a learning rate α_k checking the Wolf conditions and to compute the sequence:

$$x_{n+1} = x_n - \alpha_k [Hf(x_n)]^{-1} \nabla f(x_n).$$

This is a quasi-Newton method. When the function f is not differentiable (in the classical Fréchet sense), one can use more general definitions of differential (or gradient) such as the Clarke differential (Clarke, 1990) for instance. In Section 3, we discuss our choice of differential for our specific objectives which are not differentiable everywhere due to the use of the minmax operator.

Here, we describe exact algorithms solving zero-sum two-player MGs. The first algorithm has been proposed by Shapley (1953). Shapley's algorithm is iterative and starts with an initial value $v_0 = 0$. It computes the sequence of values $v_{k+1} = \mathcal{T}^* v_k$. The sequence of v_k converges at a geometrical rate to the optimal value of the game v^*. As mentioned in the introduction, there exist two methods extending the Howard's PI algorithm. The first one was published by Hoffman & Karp (1966); Van Der Wal (1978). It is also an iterative one that computes a sequence of values and strategies such that $\mathcal{T}_{\mu_{k+1}} v_k = \mathcal{T}^* v_k$ and of value $v_{k+1} = \mathcal{T}_{\mu_{k+1}} v_{k+1}$. Computing the value v_{k+1} requires to find the fixed point of $\mathcal{T}_{\mu_{k+1}}$ and thus solving an MDP as a subroutine which may be computationally intensive.

In contrast, the algorithm by Pollatschek & Avi-Itzhak (1969) does not compute the best response at each iteration. Instead, it computes the value of a joint strategy. It thus starts with an initial value $v_0 = 0$ and then computes the following sequence of strategies and values. At each iteration, the algorithm computes (μ_{k+1}, ν_{k+1}) such that $\mathcal{T}_{\mu_{k+1}} v_k = \mathcal{T}_{\mu_{k+1},\nu_{k+1}} v_k = \mathcal{T}^* v_k = \tilde{\mathcal{T}}_{\nu_{k+1}} v_k$ and then it finds the fixed point $v_{k+1} = \mathcal{T}_{\mu_{k+1},\nu_{k+1}} v_{k+1} = v_{\mu_{k+1},\nu_{k+1}}$. The complexity of computing the fixed point of $\mathcal{T}_{\mu_{k+1},\nu_{k+1}}$ is just inverting a matrix of size $card(S)$ instead of solving an MDP. However, this scheme doesn't work in general (Van Der Wal, 1978).

Van Der Wal (1978) shows that Shapley's algorithm is slower in practice than the two others. It thus motivates the use of PI schemes. Between the two extensions of PI to games, the Pollatschek & Avi-Itzhak algorithm is the less computationally intensive. In Filar & Tolwinski (1991), this algorithm is slightly modified to introduce a learning rate. The update is $v_{k+1} = \alpha v_{\mu_{k+1},\nu_{k+1}} + (1-\alpha) v_k$ where α is chosen according to Armijo's Rule. This comes from the fact that the Pollatschek & Avi-Itzhak algorithm is a Newton's method on the $\mathcal{L}_2$-norm of the Bellman residual $||v - \mathcal{T}^* v||_2^2$. The adaptation of the Pollatschek & Avi-Itzhak algorithm introduced by Filar & Tolwinski (1991) uses a quasi-Newton method instead of a Newton's method on the objective $\mathcal{J}(v) = ||v - \mathcal{T}^* v||_2^2$. To argue for the convergence of their algorithm towards the optimal value of the game, Filar & Tolwinski show first that a local minimum of the objective is also a global minimum and then they assure that their quasi-Newton method converges to a local minimum. However, they use a version of the Zoutendijk theorem (Nocedal & Wright, 2006) to prove the convergence to a local minimum. This theorem requires the gradient of the objective function ($\mathcal{L}_2$-norm of the Bellman residual) to exists (in the Fréchet sense) and to be a Lipschitz function. This assumption does not hold in the case of MDPs or MGs because of the max and minmax operators respectively. The question whether there is convergence to a local minimum or not when the gradient is not Lipschitz is still an open problem in optimization. However, empirical evidence suggests quasi-Newton method always converge to a Clarke stationary point even when the gradient is not Lipschitz (Lewis & Overton, 2009; 2013). This means that, despite Filar & Tolwinski's proof does not stand, there is good evidence that using quasi-Newton methods will empirically perform well. Again, there are no theoretical guarantees that quasi-Newton methods converge to a local optimum in the case of non-convex and non-smooth functions (Lewis & Overton, 2013) (which is the case of the Bellman residual). But, as written by Lemaréchal it can be "good practice to use a quasi-Newton method in nonsmooth optimization" in his opinion it "is essentially due to the fact that inaccurate line-searches are made". He also indicates that there are "no theoretical possibility to prove convergence to the right point (in fact counterexamples exist)" (see Lewis & Overton (2009)).

To summarize, the literature on exact algorithms on MGs, there exist two families of algorithms extending Howard's PI algorithm. The first family computes at each iteration a best response to the strategy of the maximizer (it includes the Hoffman & Karp algorithm). The other family only requires to evaluate a joint policy at each iteration. In this article, we will focus on the second family combined with linear function approximation.

3. Newton's Method on the OBR with Linear Function Approximation

For the remaining of the paper, we will use state-actions Q-functions instead of value functions. One can define, as in the previous section, Bellman operators on the Q-functions and the Q-function, $Q_{\mu,\nu} \in \mathbb{R}^{S \times A^2}$, for the strategy (μ, ν):

$$Q_{\mu,\nu}(s,a,b) = r(s,a,b) + \sum_{s' \in S} p(s'|s,a,b) v_{\mu,\nu}(s').$$

$$\mathcal{T}_{\mu,\nu} Q = r(s,a,b) + \sum_{s' \in S} P(s'|s,a,b) \underset{\substack{a' \sim \mu(.|s') \\ b' \sim \nu(.|s')}}{E} Q(s',a',b'),$$

$$\mathcal{T}_\mu Q = \min_\nu \mathcal{T}_{\mu,\nu} Q, \quad \tilde{\mathcal{T}}_\nu Q = \max_\mu \mathcal{T}_{\mu,\nu} Q,$$
$$\mathcal{T}^* Q = \max_\mu \min_\nu \mathcal{T}_{\mu,\nu} Q, \quad \tilde{\mathcal{T}}^* Q = \min_\nu \max_\mu \mathcal{T}_{\mu,\nu} Q.$$

The Q-function Q^* is the fixed point of operator $\mathcal{T}^*$ and the Q-function Q_μ is the fixed point of operator $\mathcal{T}_\mu$.

We will note $P_{\mu,\nu}$ the state-actions kernel defined as $P_{\mu,\nu}[(s', b^1, b^2), (s, a^1, a^2)] = \mu(b^1|s')\nu(b^2|s')p(s'|s, a^1, a^2)$. With linear function approximation, Q-functions are represented as a linear combination of d linear independent features $\Phi = [\phi_1, \phi_2, ..., \phi_d]$, where $\phi_i \in \mathbb{R}^{S\times A^2}$. Thus, for each $\omega \in \mathbb{R}^d$, we can define a Q-function $Q_\omega = \sum_{i=1}^d \omega_i \phi_i$. We also define the linear span corresponding to Φ as $\text{Span}(\Phi) = \{Q_\omega\}_{\omega\in\mathbb{R}^d}$. Moreover, it is usual to see Φ as a matrix where $\Phi[(s, a, b), i] = \phi_i(s, a, b)$. Thus, we have $Q_\omega = \Phi\omega$. In addition, we recall that for any element Q of $\mathbb{R}^{S\times A^2}$, its orthogonal projection $\Pi_{\rho,\Phi} Q$ under a non-null measure ρ over $S \times A^2$ is defined as:

$$\Pi_{\rho,\Phi} Q = \underset{u\in\text{Span}(\Phi)}{\text{argmin}} \|Q-u\|_{2,\rho} = \Phi(\Phi^\top \Delta_\rho \Phi)^{-1}\Phi^\top \Delta_\rho Q,$$

where Δ_ρ is a diagonal matrix of size $|S| \times |A|^2$ with $\Delta_\rho[(s, a, b), (s, a, b)] = \rho(s, a, b)$. Two canonical objectives to minimize can be defined in order to search for a Q-function within $\text{Span}(\Phi)$: the OBR $\mathcal{J}_{OBR}(\omega)$ and the POBR $\mathcal{J}_{POBR}(\omega)$:

$$\mathcal{J}_{OBR}(\omega) = \frac{1}{2}||\Phi\omega - \mathcal{T}^*\Phi\omega||^2_{2,\rho},$$
$$\mathcal{J}_{POBR}(\omega) = \frac{1}{2}||\Phi\omega - \Pi_{\rho,\Phi}\mathcal{T}^*\Phi\omega||^2_{2,\rho},$$
$$= \frac{1}{2}||\Phi^\top(\Phi^\top\Delta_\rho\Phi)^{-1}\Phi^\top\Delta_\rho(\Phi\omega - \mathcal{T}^*\Phi\omega)||^2_{2,\rho}.$$

Notice that explicit minimization of the OBR through gradient descent is not new (Piot et al., 2014; Baird et al., 1995), as for the POBR with approximate stochastic gradient descent (Maei et al., 2010). However, few works, at our knowledge, focus on the use of Newton's or quasi-Newton techniques to minimize those objectives. The Newton's method requires computing a gradient and a Hessian matrix. This means that one needs to compute the derivative of $\mathcal{T}^*\Phi\omega$ with respect to ω. As shown by (Correa & Seeger, 1985), when for a fixed ω, there exists a unique pair of strategies (μ_ω, ν_ω) such that $\mathcal{T}^*\Phi\omega = \mathcal{T}_{\mu_\omega,\nu_\omega}\Phi\omega = \mathcal{T}_{\mu_\omega}\Phi\omega = \tilde{\mathcal{T}}_{\nu_\omega}\Phi\omega$, the gradient of $\mathcal{T}^*\Phi$ on ω, is simply the gradient of the linearized operator $\mathcal{T}_{\mu_\omega,\nu_\omega}\Phi = r + \gamma P_{\mu_\omega,\nu_\omega}\Phi$. From now on, we will note $P_{\mu_\omega,\nu_\omega} = P_\omega$ and $\mathcal{T}_{\mu_\omega,\nu_\omega} = \mathcal{T}_\omega$. When there exist U_ω and V_ω such that, for all μ in U_ω and for all ν in V_ω, $\mathcal{T}^*\Phi\omega = \mathcal{T}_{\mu,\nu}\Phi\omega = \mathcal{T}_\mu\Phi\omega = \tilde{\mathcal{T}}_\nu\Phi\omega$, then the directional derivative of $\mathcal{T}^*\Phi\omega$ (written $\partial_d\mathcal{T}^*\Phi\omega$) is $\min_{\nu\in V_\omega}\max_{\mu\in U_\omega}\partial_d\mathcal{T}_{\mu,\nu}\Phi$ (Correa & Seeger, 1985). In practice, when the minmax is not unique, a pair of strategies (μ_ω, ν_ω) is chosen in $U_\omega \times V_\omega$ and $\mathcal{T}^*\Phi$ is linearized with $\mathcal{T}_\omega\Phi$ to compute the gradient: $\nabla\mathcal{T}^*\Phi\omega = \nabla\mathcal{T}_\omega\Phi\omega = \gamma P_\omega\Phi$.

For example, in the Pollatschek & Avi-Itzhak algorithm, a couple (μ_ω, ν_ω) is chosen such that $\mathcal{T}^*\Phi\omega = \mathcal{T}_\omega\Phi\omega = \mathcal{T}_{\mu_\omega}\Phi\omega = \tilde{\mathcal{T}}_{\nu_\omega}\Phi\omega$ which is the pair of strategies for which the chosen gradient is $\gamma P_\omega\Phi$. In LSPI for games, the gradient is not the gradient of the linearized operator $\mathcal{T}_\omega$. Rather, they choose $\mathcal{T}_{\mu,\nu}$ such that $\mathcal{T}^*\Phi\omega = \mathcal{T}_{\mu,\nu}\Phi\omega = \mathcal{T}_\mu\Phi\omega$ and with ν a deterministic strategy. Thus, the LSPI algorithm as it has been described by Lagoudakis & Parr (2002) does not follow the gradient $\gamma P_\omega\Phi$. From now, we will call the Newton-LSPI algorithm the one with the Newton's gradient $\gamma P_\omega\Phi$. The reader should notice that this difference only appears when it comes to MGs and that Newton-LSPI is exactly LSPI when applied to MDPs since ν is not considered in MDPs.

3.1. Newton's Method on the POBR

As mentioned in the previous section, the Newton's method on the POBR reduces to the LSPI algorithm but with a slightly different choice of the gradient. Here, we will consider the POBR with the linearized gradient ($\nabla\mathcal{T}^*\Phi\omega = \nabla\mathcal{T}_\omega\Phi\omega = \gamma P_\omega\Phi$) described previously. Let's write $A_\omega = \Phi^\top\Delta_\rho(I - \gamma P_{\omega_k})\Phi$ and $b = \Phi^\top\Delta_\rho r$

The POBR gradient is (for details see appendix B):

$$\nabla J_{POBR}(\omega) = A_\omega^\top(\Phi^\top\Delta_\rho\Phi)^{-1}(A_\omega\omega - b).$$

The Hessian matrix is (for details see appendix B):

$$HJ_{POBR}(\omega) = A_\omega^\top(\Phi^\top\Delta_\rho\Phi)^{-1}A_\omega.$$

If the matrix A_ω is invertible and if ρ is an on-null measure on $S \times A^2$, the Newton's method direction is:

$$[HJ_{POBR}(\omega)]^{-1}\nabla J_{POBR}(\omega) = (A_\omega)^{-1}(A_\omega\omega - b).$$

The update of ω_k of the Newton's method is thus:

$$\omega_{k+1} = \omega_k - (A_{\omega_k})^{-1}(A_{\omega_k}\omega_k - b) = (A_{\omega_k})^{-1}b,$$

which is the update of the Newton-LSPI algorithm Lagoudakis & Parr (2003). The Newton-LSPI algorithm optimizes $J_{POBR}(\omega)$ as a cost function, which might never be zero (see Remark A.1 below). Moreover, to our knowledge, controlling the POBR does not allow controlling the quality of the final policy. The main benefit of the Newton-LSPI algorithm is that the matrix A_{ω_k} is easy to estimate from batch data whether the state space is continuous or not. This makes Newton-LSPI a very practical algorithm (Lagoudakis & Parr, 2002).

It is well known that, for a fixed policy, the minimum of the projected Bellman residual $\frac{1}{2}||\Phi^\top(\Phi^\top\Delta_\rho\Phi)^{-1}\Phi^\top\Delta_\rho(\Phi\omega - \mathcal{T}_\pi\Phi\omega)||^2_{2,\rho}$ is equal to 0 (Koller & Parr, 2000) for all but finitely many number of γ. Nonetheless, we can show that it is not the case for the POBR. In other words, the minimum of $\frac{1}{2}||\Phi^\top(\Phi^\top\Delta_\rho\Phi)^{-1}\Phi^\top\Delta_\rho(\Phi\omega - \mathcal{T}^*\Phi\omega)||^2_{2,\rho}$ might be positive for a continuous interval of γ (see appendix A.1).

3.2. Newton's Method on the OBR

Here we will note $C_\omega = \Phi^\top(I-\gamma P_\omega)^\top\Delta_\rho(I-\gamma P_\omega)\Phi$ and $e_\omega = \Phi^\top(I-\gamma P_\omega)^\top\Delta_\rho r$. Applying the Newton's method to the residual $\mathcal{J}_{OBR}(\omega)$ gives the following algorithm:

$$\omega_0 = 0,$$
$$\omega_{k+1} = \omega_k + (H\mathcal{J}_{OBR}(\omega_k))^{-1}\nabla\mathcal{J}_{OBR}(\omega_k).$$

The gradient is (for details see appendix B):

$$\nabla\mathcal{J}_{OBR}(\omega) = C_\omega\omega - e_\omega.$$

The Hessian matrix is (for details see appendix B):

$$H\mathcal{J}_{OBR}(\omega) = C_\omega.$$

The update of ω_k of the Newton's method is:

$$\begin{aligned}\omega_{k+1} &= \omega_k - (C_{\omega_k})^{-1}(C_\omega\omega_k - e_{\omega_k}),\\ &= (C_{\omega_k})^{-1}e_{\omega_k}.\end{aligned}$$

Lagoudakis & Parr (2003) propose this algorithm as an alternative method to learn an optimal policy in MDPs and call it Bellman Residual Minimization Policy Iteration (BRMPI). Extension to games is easy as shown previously.

3.3. Comparison of BRMPI and Newton-LSPI

In MGs, the two algorithms previously described derive from the Pollatschek & Avi-Itzhak algorithm and are thus not guaranteed to converge (Van Der Wal, 1978). They both follow the Newton's direction of either $\mathcal{J}_{OBR}$ or $\mathcal{J}_{POBR}$. On the one hand, the cost function $\mathcal{J}_{OBR}(\omega)$ controls the norm $||Q^* - Q_\mu||_2$ where μ is the maximizer's greedy strategy with respect to $\Phi\omega$ and Q_μ the fixed point of $\mathcal{T}_\mu$ (see (Piot et al., 2014) in the case of an MDP). In consequence, applying the Newton's method to the OBR gives a nearly optimal strategy. However, there are no easy ways to estimate the matrix C_ω and the vector e_ω from data, especially when the state space is continuous. One possible way to obtain such estimates is to use embeddings in RKHS (Grunewalder et al., 2012).

Theorem 1. *Let Q be a value function and let μ be a greedy strategy on Q (i.e. $\mathcal{T}^*Q = \mathcal{T}_\mu Q$). Then we have:*

$$||Q^* - Q_\mu||_{2,\rho} \leq \frac{2}{1-\gamma}\left(\frac{C_2(\rho,\mu,\nu) + C_2(\rho,\mu^*,\tilde{\nu})}{2}\right)^{\frac{1}{2}} ||\mathcal{T}^*Q - Q||_{2,\rho}$$

with μ, ν, μ^* and $\tilde{\nu}$ strategies such that $\mathcal{T}^*Q^* = \mathcal{T}_{\mu^*}Q^*$, $\mathcal{T}_{\mu^*,\tilde{\nu}}Q = \mathcal{T}_{\mu^*}Q$, $\mathcal{T}_{\mu,\overline{\nu}}Q_\mu = \mathcal{T}_\mu Q_\mu$ and $\mathcal{T}_{\mu,\nu}Q = \mathcal{T}_\mu Q$. And with $C_2(\rho,\mu,\nu) = ||\frac{\partial\rho^\top(1-\gamma)(I-\gamma P_{\mu,\nu})^{-1}}{\partial\rho^\top}||_{2,\rho}$ $\mathcal{L}_\in$-norm of the Radon-Nikodym derivative of the Kernel $(1-\gamma)(I-\gamma P_{\mu,\nu})^{-1}$.

Proof. Proof in appendix A. □

On the other hand, applying the Newton's method to $\mathcal{J}_{POBR}(\omega)$ is easy. Estimating A_ω and b_ω is computationally cheap. However, minimizing $\mathcal{J}_{POBR}(\omega)$ does not guarantee to find a good strategy.

Concerning the stability, Newton-LSPI is clearly the less stable one. The Hessian matrix $H\mathcal{J}_{POBR}(\omega)$ might not even be invertible since A_ω is not invertible for a finite number of values of γ (Koller & Parr, 2000). For BRMPI however the Hessian $H\mathcal{J}_{OBR}(\omega)$ is always invertible and the cosine between the Newton's update and the gradient is bounded away from zero.

Theorem 2. *For an MG with finite state space, there exists $\delta > 0$ such that:*

$$-\frac{\mathcal{J}_{OBR}(\omega)^\top(H\mathcal{J}_{OBR}(\omega))^{-1}\mathcal{J}_{OBR}(\omega)}{||\mathcal{J}_{OBR}(\omega)||_2||(H\mathcal{J}_{OBR}(\omega))^{-1}\mathcal{J}_{OBR}(\omega)||_2} \leq -\delta.$$

Proof. Proof in appendix A □

This means that, when on a point ω where the derivative is well defined, the Newton's update is guaranteed to decrease the OBR in a neighborhood of ω.

4. Batch Learning in Games

In the batch scenario, the dynamics and the reward function are known only through logs of interactions between the two players and the environment. These logs are a collection of tuple $\{(s_i, a_i^1, a_i^2, r_i, s_i')\}_{i\in\{1,\dots,N\}}$ where (s_i, a_i^1, a_i^2) is drawn from a distribution ρ where $r_i = r(s_i, a_i^1, a_i^2)$ and where $s_i' \sim p(.|s_i, a_i^1, a_i^2)$. Thus, at each iteration of Newton-LSPI or of BRMPI, the objective will be to estimate one matrix and one vector.

4.1. Newton-LSPI with Batch Data

Newton-LSPI with batch data proceeds as follows. The goal is to estimate the matrix A_ω and the vector b (the estimates will be noted $\hat{A}_\omega$ and $\hat{b}$). The procedure is described in Algorithm 1 where μ_ω and ν_ω are the minmax strategies with respect to the approximate Q-function $\Phi\omega$. The update of $\hat{A}_\omega$ is convenient since it allows computing $(\hat{A}_\omega)^{-1}$ very efficiently (Lagoudakis & Parr, 2003) with the Sherman–Morrison formula. Then doing Newton-LSPI with batch data corresponds simply in updating our parameter ω as follows:

$$\omega_{k+1} = (\hat{A}_{\omega_k})^{-1}\hat{b}.$$

Algorithm 1 LSPI-Matrix update

```
    Input: $D_N = ((x_j, a_j^1, a_j^2), r_j, x'_j)_{j=1,...,N}$ some samples, $\Phi$ a $d$ dimensional feature space and $\omega$.
    for $i \in \{1, ..., N\}$ do
        $\hat{A}_\omega += \phi(s_i, a_i^1, a_i^2)[\phi(s_i, a_i^1, a_i^2) - \gamma \sum_{b^1 \in A^1(s'_i)} \sum_{b^2 \in A^2(s'_i)} \mu_\omega(s'_i, b^1)\nu_\omega(s'_i, b^2)\phi(s'_i, b^1, b^2)]^\top$
        $\hat{b} \ += \phi(s_i, a_i^1, a_i^2) r_i$
    end for
output $\hat{A}_\omega$, $\hat{b}$
```

4.2. BRMPI with Batch Data

A natural extension of BRMPI to batch data when the state space is finite is to find estimates of C_ω and e_ω. To do so we first estimate $(I - \gamma P_\omega)$ (noted $\hat{B}_\omega$) and r (noted $\hat{r}$). The procedure is described in Algorithm-2.

Algorithm 2 BRMPI-Matrix update

```
    Input: $D_N = ((x_j, a_j^1, a_j^2), r_j, x'_j)_{j=1,...,N}$ some samples, $\Phi$ a $d$ dimensional feature space and $\omega$.
    for $i \in \{1, ..., N\}$ do
        $\hat{B}_\omega(s_i, a_i^1, a_i^2, s_i, a_i^1, a_i^2) \ += 1$
        for $b^1 \in A^1(s'_i)$ and $b^2 \in A^2(s'_i)$ do
            $\hat{B}_\omega(s_i, a_i^1, a_i^2, s'_i, b^1, b^2) += -\gamma \mu_\omega(s'_i, b^1)\nu_\omega(s'_i, b^2)$
        end for
        $\hat{r}(s_i, a_i^1, a_i^2) \leftarrow \hat{r}(s_i, a_i^1, a_i^2) + r_i$
    end for
output $\hat{B}_\omega$, $\hat{r}$
```

Then, estimates of C_ω and e_ω would be respectively $\hat{C}_\omega = \Phi^\top \hat{B}_\omega^\top \hat{B}_\omega \Phi$ and $\hat{e}_\omega = \Phi^\top \hat{B}_\omega^\top \hat{r}$. Those estimates are clearly biased estimates of C_ω and e_ω. Thus, the Newton's update is:

$$\omega_{k+1} = (\hat{C}_{\omega_k})^{-1} \hat{e}_{\omega_k}.$$

5. Quasi-Newton's Method on the OBR and on the POBR

In this section, a quasi-Newton's method is applied to the norm of the OBR and of the PORBR. This corresponds to the introduction of a learning rate in Newton-LSPI and BRMPI and their batch versions described in the hereinbefore section. In (Filar & Tolwinski, 1991) the learning rate is chosen to verify Amijo's rule for the exact case. We choose to use a learning rate according to the Wolf conditions for the case with linear function approximation. There exist theorems to guarantee convergence to a local minimum when the gradient is a Lipschitz function which is not the case here. However, there are empirical evidences that using inexact line search improves convergence to a local minimum of the objective function (Lewis & Overton, 2009).

In both Newton-LSPI and BRMPI, when the dynamics is known and in the batch scenario, the update takes the following shape ($\Xi_\omega = A_\omega$ and $\psi_\omega = b$ for LSPI):

$$\omega_{k+1} = (\Xi_{\omega_k})^{-1} \psi_{\omega_k}$$

and the objective is to minimize a $\mathcal{L}_2$ norm of the following shape ($\Psi_\omega = B_\omega \Phi$ and $\upsilon = r$ for BRMPI):

$$f(\omega) = ||\Psi_\omega \omega - \upsilon||_2.$$

Let's note the Newton's update:

$$g_{\omega_k} = (\Xi_{\omega_k})^{-1} \psi_{\omega_k} - \omega_k,$$

The quasi-Newton method will instead perform the following update:

$$\omega_{k+1} = (1 - \alpha_k)\omega_k + \alpha_k (\Xi_{\omega_k})^{-1} \psi_{\omega_k},$$

where the learning rate α is chosen to satisfy the Wolf conditions ($0 < c_1 < c_2 < 1$):

$$f(\omega + \alpha g_\omega) \leq f(\omega) + c_1 \alpha g_\omega^\top . \nabla f(\omega) \quad (1)$$

$$f(\omega + \alpha g_\omega) \geq f(\omega) + c_2 \alpha g_\omega^\top . \nabla f(\omega) \quad (2)$$

Constants c_1 is typically chosen around 10^{-4} and c_2 is usually chosen close to 0.9. Condition (1) ensures that the learning rate is not too large whereas condition (2) guarantees a large enough learning rate.

6. Experiments

To empirically illustrate the application of quasi-Newton methods to both MDPs and MGs, we ran experiments on a class of randomly generated MDPs and MGs namely Garnets (see Archibald et al. (1995) for reference on Garnets on MDPs). A Garnet is an abstract class of MDPs. It is described by a triplet (N_S, N_A, N_B). Parameters N_S and N_A are respectively the number of states and the number of actions per state. The parameter N_B, the branching factor, denotes the number of reachable states from any state-action pair. Thus, for all (s, a) we generate the following transition distribution $p(.|s, a)$ according to the following procedure: first one draws uniformly $N_B - 1$ points over $[0, 1]$. Let's note those numbers $(p_i)_{1 \leq i \leq N_B - 1}$ (in increasing order) and $p_0 = 0$ and $p_{N_B} = 1$. Then we draw a subset of size N_B of $\{1, ..., N_S\}$ written $\{s'_1, ..., s'_{N_B}\}$ in the case of type 1 Garnet. In the case of type 2 Garnet, $\{s'_1, ..., s'_{N_B}\}$ is drawn in a subset of $\{1, ..., N_S\}$ centered around s (meaning we will draw states s' such that $|s - s'| \leq \zeta$ in the experiment $\zeta = N_B$). Finally, we define the kernel as follows,

$\forall i \in \{1, ..., N_B\}$, $p(s'_i|s, a) = p_i - p_{i-1}$. The reward function will depend on the experiment.

For two-player zero-sum MGs the N_A parameter will describe the number of actions both players will be allowed to play at each state in the MG. The kernel $p(.|a^1, a^2, s)$ is generated according to the same procedure as for MDPs.

Results always show the performance of the strategy with respect to the iteration number. The performance of a strategy μ is quantified as the following ratio: $\frac{||Q^* - Q_\mu||_2}{||Q^*||_2}$. This is the normalized norm of the difference between the optimal Q-function of the MG and the Q-function of strategy μ considering the opponent plays his best response. Results are averaged over experiments and the envelope is proportional to the standard deviation.

The main purpose of the two following subsections is to emphasize the effects of quasi-Newton methods both on the stability and on the performance of the algorithms. In those experiments, we can notice the importance of tuning constants c_1 and c_2 from equations (1) and (2). Here, $c_1 = 10^{-4}$ and $c_2 = 0.9$. We performed the line search as follows: if condition (1) was not checked we decreased the learning rate geometrically $\alpha_k \leftarrow \eta \times \alpha_k$ and, if condition (2) was not checked, we increased it geometrically $\alpha_k \leftarrow \frac{1}{\eta} \times \alpha_k$ ($\eta = 0.9$). This was done until both conditions were checked.

6.1. Experiments on Markov Decision Processes

The PI algorithm for MDPs is known to converge because it builds a sequence of increasing values while searching in an underlying finite policy space (Puterman, 1994). This does not stand anymore when it comes to function approximation, since the argument of increasing values is not verified. As a result, the LSPI algorithm is not proven to converge in general for MDPs.

We ran experiments on Garnets to compare LSPI and BRMPI (Newton's methods) with their quasi-Newton counterparts (Softened LSPI described in Algorithm-3). Here, the reward function depends on states and actions, and only a ratio of the rewards were non-zero (this ratio is called the sparsity). Experiments of figure 1 was done on type 2 Garnets. The feature space Φ has $d = 0.2 \times N_S \times N_A$ features. Those d features are vectors randomly generated according to a Gaussian law.

First, one can notice that quasi-Newton version of LSPI and respectively BRMPI did not under-perform LSPI and respectively BRMPI. Figure 1 illustrates that quasi-Newton methods do not under perform their respective counterparts. We noticed that the BRMPI algorithm often oscillates from one iteration to another. The use of quasi-Newton method usually eliminates that instability.

In Figure 1 the Garnet is drawn with $N_B = 10$. Usually, MDPs with high branching factors are easier to optimize since a high branching factor has a tendency to smooth the optimal value function. Figure 1 shows the effect of the learning rate on the learning curve. Especially in the case of Newton-LSPI, introducing a learning rate improves both the stability and the performance of the algorithm. Also, we can notice that the BRMPI algorithm oscillates on MDPs.

Algorithm 3 quasi-Newton LSPI

```
  Input: $D_N = ((x_j, a^1_j, a^2_j), r_j, x'_j)_{j=1,...,N}$ some samples, $\Phi$ a $d$ dimensional feature space and $\omega_0 = 0$. Constants $c_1 (= 10^{-4})$,$c_2 (= 0.9)$, $\eta (= 0.9)$, $\alpha_0 = 1$, $\alpha_{min} (= 10^{-10})$ and $\alpha_{min} (= 1.0)$.
  let's suppose there is an algorithm $f$ such that $\hat{A}_\omega, \hat{b} = f(D_N, \omega, \Phi)$ (the procedure is described in section 4.1)
  for k=1,2,...,K do
    $\hat{A}_{\omega_k}, \hat{b} = f(D_N, \omega_k, \Phi)$
    $g_{\omega_k} = (\hat{A}_{\omega_k})^{-1}\hat{b} - \omega_k$
    $p_{\omega_k} = \hat{A}^\top_{\omega_k}(\hat{A}_{\omega_k}\omega_k - \hat{b})$
    $\alpha_k = \alpha_{k-1}$
    $l_{\omega_k} = ||\hat{A}_{\omega_k}\omega_k - \hat{b}||^2_2$
    $\hat{A}_{\omega_k + \alpha_k g_{\omega_k}}, \hat{b} = f(D_N, \omega_k + \alpha_k g_{\omega_k}, \Phi)$
    $l = ||\hat{A}_{\omega_k + \alpha_k g_{\omega_k}}(\omega_k + \alpha_k g_{\omega_k}) - \hat{b}||^2_2$
    if $l > l_{\omega_k} + c_1 \alpha_k g^\top_{\omega_k} p_{\omega_k}$ then
      while $l > l_{\omega_k} + c_1 \alpha_k g^\top_{\omega_k} p_{\omega_k}$ and $\alpha_k \geq \alpha_{min}$ do
        $\alpha_k \leftarrow \alpha_k \times \eta$
        $\hat{A}_{\omega_k + \alpha_k g_{\omega_k}}, \hat{b} = f(D_N, \omega_k + \alpha_k g_{\omega_k}, \Phi)$
        $l = ||\hat{A}_{\omega_k + \alpha_k g_{\omega_k}}(\omega_k + \alpha_k g_{\omega_k}) - \hat{b}||^2_2$
      end while
    end if
    if $l < l_{\omega_k} + c_1 \alpha_k g^\top_{\omega_k} p_{\omega_k}$ then
      while $l < l_{\omega_k} + c_1 \alpha_k g^\top_{\omega_k} p_{\omega_k}$ and $\alpha_k \leq \alpha_{max}$ do
        $\alpha_k \leftarrow \frac{\alpha_k}{\eta}$
        $\hat{A}_{\omega_k + \alpha_k g_{\omega_k}}, \hat{b} = f(D_N, \omega_k + \alpha_k g_{\omega_k}, \Phi)$
        $l = ||\hat{A}_{\omega_k + \alpha_k g_{\omega_k}}(\omega_k + \alpha_k g_{\omega_k}) - \hat{b}||^2_2$
      end while
    end if
    $\omega_{k+1} = \omega_k + \alpha_k g_{\omega_k}$
  end for
output $\Phi\omega_{K+1}$
```

6.2. Experiments on Markov Games

We ran experiments on MG Garnets to compare Newton's method and quasi-Newton method. In the hereafter experiment, the reward function only depends on the state. Only a ratio of rewards are non-zero and are drawn according to a normal law. Again, we compared Newton-LSPI and BRMPI with their quasi-Newton counterparts. The feature space is also randomly generated according to a Gaussian law. But, we needed more features compared to the MDP

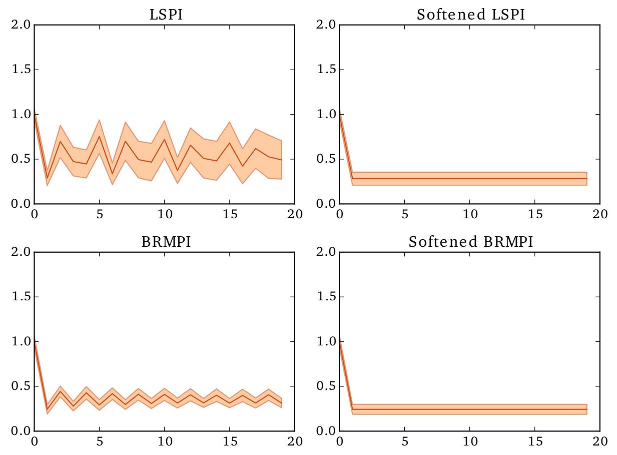

Figure 1. Performance (y-axis) of the strategy at step k (x-axis) for LSPI, BRMPI and the corresponding quasi-Newton method. Results are averaged over 50 Garnets $N_S = 100$, $N_A = 8$, $N_B = 10$. All Garnets have a sparsity of 0.5 and $\gamma = 0.99$. The number of batch samples used is $0.5 \times N_A \times N_S$.

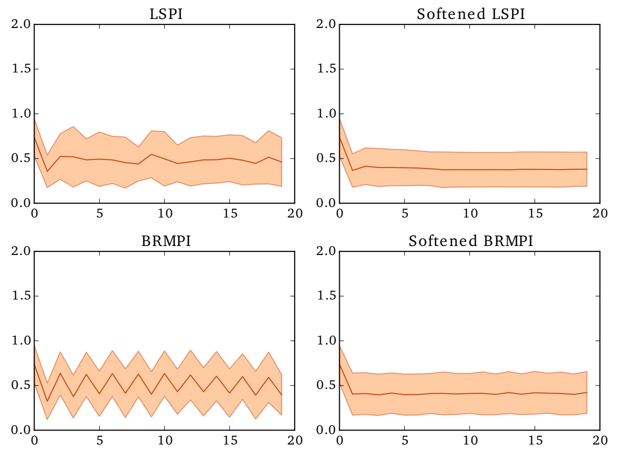

Figure 2. Performance (y-axis) of the strategy at step k (x-axis) for Newton-LSPI, BRMPI and the corresponding quasi-Newton method. Results are averaged over 50 Garnets $N_S = 50$, $N_A = 2$, $N_B = 1$. All Garnets have a sparsity of 0.3 and $\gamma = 0.9$. The number of batch samples used is $1.0 \times N_A \times N_A \times N_S$.

case to learn a strategy. In the following experiment, we used $d = 0.8 \times N_A \times N_A \times N_S$ features.

Compared to experiments on MDPs, one can notice that the variance of the performance is higher even if we use more features, more samples on MGs with less actions. We do not fully understand why it is significantly harder to learn a good strategy in an MG compared to an MDP. The reason might be that simultaneous games are simply more complex to optimize or that comparing with the value of a best response is too conservative. Anyway, the use of quasi-Newton method appears to be useful even with this conservative performance criterion.

Here again, one can notice that quasi-Newton methods did not under-perform their counterparts. Although, we could not notice huge gaps in the performance (Figure 2 and 3) as in experiments on MDPs, quasi-Newton methods did reduce the unstable behavior of the strategy (Figure 2).

7. Conclusion

To sum up, we first pointed out the fact that the proof of convergence of Filar & Tolwinski algorithm is based on assumptions that do not hold. Second, we found out that, as a consequence of the instability of the Pollatschek & Avi-Itzhak algorithm, LSPI for games does not converge whether the linear approximation is stable or not. Third, we showed that LSPI and BRMPI can be regarded as Newton's methods. Fourth, this naturally led to the use of quasi-Newton methods instead of Newton's method as they result in more stable solutions. And, finally, these slight modifications on algorithms for games dramatically improve the stability of one of the most popular model-free algorithm to solve MDPs on synthetic problems, namely LSPI.

This paper makes novel connections between approximate policy iteration schemes on both MDPs and MGs and optimization methods. It describes 3 algorithms as Newton's method on different Bellman residuals. This unified picture of 3 popular algorithms naturally led to the use of quasi-Newton method which is believed, in the optimization community, to be steady and to perform better in practice (Nocedal & Wright, 2006).

Yet, this paper rises three open questions: first, since LSPI and BRMPI with batch data are minimizing some empirical residual, a natural question is whether those estimators are good estimators of the residual or not. Few works exist on direct minimization of the OBR (Baird et al., 1995; Piot et al., 2014). Piot et al. proved that minimizing a specific estimate of the Optimal Bellman residual on MDPs is consistent in the Vapnik sense. A second question is whether it makes sense theoretically to minimize the POBR or not. Little is known on the quality of the solution given by the minimum of the POBR and research should be conducted in that sense. Finally, since the goal is to minimize some Bellman residual, a third question is whether there exist smarter methods than quasi-Newton ones to minimize the considered residual. There exists a trove of Newton-like methods that are known to perform well on non-smooth functions. One interesting perspective would be to study how the BFGS method (Nocedal & Wright, 2006; Lewis & Overton, 2013) would perform, as it does not require to compute and invert the Hessian matrix at each iteration.

Acknowledgments

This work has received partial funding from CPER Nord-Pas de Calais/FEDER DATA Advanced data science and technologies 2015-2020 , the European Commission H2020 framework program under the research Grant number 687831 (BabyRobot) and the European Union Seventh Framework Program (FP7/2007-2013) under grant agreement number 270780. Experiments presented in this paper were carried out using the Grid'5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see https://www.grid5000.fr).

References

Archibald, TW, McKinnon, KIM, and Thomas, LC. On the generation of markov decision processes. *Journal of the Operational Research Society*, pp. 354–361, 1995.

Baird, Leemon et al. Residual algorithms: Reinforcement learning with function approximation. In *Proc. of ICML*, 1995.

Clarke, Frank H. *Optimization and nonsmooth analysis*, volume 5. Siam, 1990.

Correa, Rafael and Seeger, Alberto. Directional derivative of a minimax function. *Nonlinear Analysis: Theory, Methods & Applications*, 9(1):13–22, 1985.

Filar, J. A. and Tolwinski, B. *On the Algorithm of Pollatschek and Avi-ltzhak*. Springer, 1991.

Grunewalder, Steffen, Lever, Guy, Baldassarre, Luca, Pontil, Massi, and Gretton, Arthur. Modelling transition dynamics in mdps with rkhs embeddings. *Proc. of ICML*, 2012.

Hoffman, A. J. and Karp, R. M. On nonterminating stochastic games. *Management Science*, 12(5):359–370, 1966.

Koller, Daphne and Parr, Ronald. Policy iteration for factored mdps. In *Proc. of UAI*, pp. 326–334, 2000.

Lagoudakis, M. G. and Parr, R. Least-squares policy iteration. *Journal of Machine Learning Research*, pp. 1107–1149, 2003.

Lagoudakis, Michail G and Parr, Ronald. Value function approximation in zero-sum markov games. In *Proc. of UAI*, 2002.

Lewis, Adrian S and Overton, Michael L. Nonsmooth optimization via bfgs. *Submitted to SIAM J. Optimiz*, 2009.

Lewis, Adrian S and Overton, Michael L. Nonsmooth optimization via quasi-newton methods. *Mathematical Programming*, 141(1-2):135–163, 2013.

Maei, Hamid R, Szepesvári, Csaba, Bhatnagar, Shalabh, and Sutton, Richard S. Toward off-policy learning control with function approximation. In *Proc. of ICML*, pp. 719–726, 2010.

Nocedal, Jorge and Wright, Stephen. *Numerical optimization*. Springer Science & Business Media, 2006.

Perolat, Julien, Scherrer, Bruno, Piot, Bilal, and Pietquin, Olivier. Approximate Dynamic Programming for Two-Player Zero-Sum Markov Games. In *Proc. of ICML*, 2015.

Piot, Bilal, Geist, Matthieu, and Pietquin, Olivier. Difference of convex functions programming for reinforcement learning. In *Proc. of NIPS*. 2014.

Pollatschek, MA and Avi-Itzhak, B. Algorithms for stochastic games with geometrical interpretation. *Management Science*, 15(7):399–415, 1969.

Puterman, M. L. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 1994.

Shapley, L. S. Stochastic Games. *Proceedings of the National Academy of Sciences of the United States of America*, 1953.

Van Der Wal, J. Discounted Markov Games: Generalized Policy Iteration Method. *Journal of Optimization Theory and Applications*, 25(1):125–138, 1978.

Von Neumann, J. Morgenstern, 0.(1944) theory of games and economic behavior. *Princeton: Princeton UP*, 1947.