Following Newton Direction in Policy Gradient with Parameter Exploration

Giorgio Manganini, Matteo Pirotta, Marcello Restelli, Luca Bascetta
Dipartimento di Elettronica, Informazione e Bioingegneria
Politecnico di Milano, Piazza Leonardo da Vinci 32, 20133 Milano, Italy
E-mail: {matteo.pirotta, giorgio.manganini, marcello.restelli, luca.bascetta}@polimi.it

***Abstract*—This paper investigates the use of second–order methods to solve Markov Decision Processes (MDPs). Despite the popularity of second–order methods in optimization literature, so far little attention has been paid to the extension of such techniques to face sequential decision problems. Here we provide a model–free Reinforcement Learning method that estimates the Newton direction by sampling directly in the parameter space. In order to compute the Newton direction we provide the formulation of the Hessian of the expected return, a technique for variance reduction in the sample–based estimation and a finite sample analysis in the case of Normal distribution. Beside discussing the theoretical properties, we empirically evaluate the method on an instructional linear–quadratic regulator and on a complex dynamical quadrotor system.**

I. Introduction

Policy search is a Reinforcement Learning (RL) approach that focuses on the search for the optimal policy of a Markov Decision Process (MDP) in a limited policy space. It has gained popularity as an approach for complex, real applications since it can deal with high–dimensional state and action spaces, while keeping the search limited to a task–appropriate predefined parametrized policy class. In particular, policy search has become the standard RL approach in robotics [1].

Policy–gradient approaches perform gradient ascent in the parametrized policy space in order to derive the optimal policy directly and not through the estimation of the optimal value function. Besides being effective in many complex RL problems [2], they enjoy stability and robustness guarantees [3]. In order to produce an unbiased estimate of the gradient in model–free settings they require non–zero probability for every action in every visited policy, so that stochastic policies need to be considered. In robotic applications, where the underlying model is slightly stochastic or even deterministic– as the most of the industrial applications–stochastic policies are required to obtain a sufficient exploration of the domain, but collide with the need of interpratibility required by committees. Although several methods for variance reduction have been derived, the exploration in the action space may inject noise in the gradient estimate at every step. A second drawback is the use of stochastic policies since predictability of the policy behavior is a requisite in many robotic applications.

In order to overcome such limitations, several algorithms have been designed to incorporate an exploration strategy in the parameter space, by perturbing the policy parameters [4]–[6]. This is usually obtained through a parametric upper–level policy (or distribution) that selects the parameters of the (possibly deterministic) control policy. Among them, we focus on the Policy Gradients with Parameter–based Exploration (PGPE [6]) that exploits gradient ascent algorithms in order to find the optimal parametrization of the upper–level policy. The advantages are: I) less noisy estimate both from theoretical and empirical analysis [7]; II) possibility to exploit non–differentiable policies. Furthermore, PGPE has been shown to outperform standard RL gradient approaches in many complex scenarios [8], [9].

Unfortunately, the PGPE inherits the main drawback of gradient approaches, i.e., that the choice of the step size can affect both performance and convergence. Attempts to design more advanced gradient methods are limited by the fact that conventional techniques, such as line search and Krylov subspaces, are not suited for MDP settings, where the model function is unknown [10]. As far as we know, [11] is the only notable exception that has investigated an automatic technique for selecting the step size in policy gradient approaches.

A possible approach is to consider higher-order optimization methods. A popular search direction, maybe the most popular [12], is the Newton direction. Such direction is obtained from the second–order Taylor series approximation and, unlike steepest ascent direction, it is associated to a natural step size of 1. Moreover, Newton direction has a faster local convergence rate than the plain gradient, usually quadratic [12]. The main drawback is that the estimation or even computation of the second derivatives is usually hard and error prone.

After the analysis performed in [13], the RL literature has posed little attention to higher–order methods. In his work, Kakade showed that the estimate of the second derivatives of the policy expected return requires the knowledge of the MDP model. Recently, [14] have shown that, exploiting the trajectory–based definition of the Policy Gradient Theorem, it is possible to get rid of the knowledge of the model. In this paper, we will show that the hierarchical structure, or better the direct exploration in the parameter space, allows a simple derivation and estimation of the trajectory–based Hessian matrix. We also study the variance of the Hessian estimate in order to provide a finite sample analysis in the case of Gaussian distribution.

The most related RL approach in the literature is the Natural gradient [2], [13]. The connection between the Natural gradient and second–order methods resides in the fact that,

978-1-4799-1959-8/15/$31.00

when the cost function is quadratic, the Fisher information matrix is equal to the Hessian matrix, and thus Newton's and the natural gradient methods are identical [15].

Approaches similar to the PGPE have been derived in the field of Evolutionary Strategies. The similarities (also pointed out in in [16]) reside in the fact that the policy improvement step is treated as a black–box optimization problem, see [17]. The closest algorithm to the proposed one is the Natural Evolution Strategy (NES, [18]), which is a black–box optimization methods that exploits Natural Gradient information. It is worth noticing that Natural PGPE [19] and NES share the same algorithmic structure.

In this paper we compare our approach with its predecessor PGPE (well known and empirically compared in RL literature also against ES and natural algorithms with perturbations in the parameter space) and also with Natural PGPE, which has similar second–order properties as the Newton update, just to remain in the same algorithmic framework.

In Section III, we provide the Newton direction associated to the PGPE hierarchical structure and we show that the Hessian formulation is invariant to the introduction of a constant baseline. Such term is used to derive a low variance Hessian estimate (Section IV) directly from experience. The amount of variance reduction is also estimated. We further give a sample complexity analysis when the upper–level policy is a Gaussian distribution. Finally, the usefulness of the Newton method is demonstrated through experiments (Section V).

II. Preliminaries

A discrete–time continuous Markov decision process (MDP) is defined as a 6-tuple $\langle X, U, \mathcal{P}, \mathcal{R}, \gamma, D\rangle$, where X is the continuous state space, U is the continuous action space, $\mathcal{P}$ is a Markovian transition model where $\mathcal{P}(x'|x,u)$ defines the transition density between state x and x' under action u, $\mathcal{R}: X \times U \to \mathbb{R}$ is the reward function, such that $\mathcal{R}(x,u)$ is the expected immediate reward for the state-action pair (x,u), $\gamma \in [0,1)$ is the discount factor for future rewards, and D is the distribution of the initial state. The control policy is characterized by a density distribution $\pi(\cdot|x)$ that specifies for each state x the density distribution over the action space U.

We consider infinite horizon problems where the future rewards are exponentially discounted with γ. For each state x, we define the utility of following a stationary policy π as:

$$V^{\pi}(x) = \mathbb{E}_{\substack{u_t \sim \pi \\ x_t \sim \mathcal{P}}} \left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(x_t, u_t) \middle| x_0 = x \right].$$

Policies can be ranked by their expected discounted reward starting from the state distribution D:

$$J_D^{\pi} = \int_X D(x) V^{\pi}(x) \mathrm{d}x = \int_{\mathbb{T}} p(\tau|\pi)\, \mathcal{R}(\tau) \mathrm{d}\tau,$$

where $\tau \in \mathbb{T}$ is a trajectory of length T (possibly infinite) drawn from density distribution $p(\tau|\pi)$ with expected γ–discounted return: $\mathcal{R}(\tau) = \sum_{t=1}^{T} \gamma^{t-1} \mathcal{R}(x_t, u_t)$. The trajectory probability is simply described by the following equation:

$$p(\tau|\pi) = D(x_0) \prod_{k=1}^{T} \mathcal{P}(x_k|x_{k-1}, u_{k-1}) \pi(u_{k-1}|x_{k-1}).$$

Solving an MDP means finding a policy π^* that maximizes the expected long-term reward: $\pi^* \in arg\max_{\pi \in \Pi} J_D^{\pi}$. For any MDP there exists at least one deterministic optimal policy that simultaneously maximizes $V^{\pi}(x)$, $\forall x \in X$ [20].

A. Policy Gradient

We consider the problem of finding a policy that maximizes the expected discounted reward over a class of parametrized policies $\Pi_{\boldsymbol{\theta}} = \{\pi_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \mathbb{R}^d\}$, where $\pi_{\boldsymbol{\theta}}$ is a compact representation of $\pi(u|x, \boldsymbol{\theta})$.

The policy parameters can be updated following the direction of the gradient of the expected discounted reward: $\boldsymbol{\theta}' = \boldsymbol{\theta} + \alpha \nabla_{\boldsymbol{\theta}} J_D(\boldsymbol{\theta})$, where [2]

$$\nabla_{\boldsymbol{\theta}} J_D(\boldsymbol{\theta}) = \int_{\mathbb{T}} p(\tau|\boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} \log p(\tau|\boldsymbol{\theta}) \mathcal{R}(\tau) \mathrm{d}\tau.$$

Given that $\nabla_{\boldsymbol{\theta}} \log p(\tau|\boldsymbol{\theta}) = \sum_{t=1}^{T} \nabla_{\boldsymbol{\theta}} \log \pi(u_t|x_t, \boldsymbol{\theta})$, the above equation shows that the differentiability of the expected return is related to the differentiability of the policy model. In general, integrating over all possible trajectories is practically unfeasible, and the gradient is estimated, e.g., by means of Monte Carlo simulations, as in the REINFORCE algorithm [10]. Standard policy–gradient approaches require actions to be stochastically selected according to parameter vector $\boldsymbol{\theta}$, resulting in high–variance gradient estimates [2]. Recently, a deterministic version of the policy gradient theorem was provided in [21]. Although the name suggests the absence of stochastic policies, this is not the case. In fact, in order to explore the entire state and action space, they exploit stochastic policies. The result is an off–policy learning algorithm that behaves according to a stochastic policy, but learns a deterministic target policy.

B. PGPE

In order to overcome the limits of policy gradient methods, the Policy Gradient with Parameter–Based Exploration (PGPE) was proposed in [6]. Unlike standard policy gradient, PGPE is able to directly exploit deterministic policies both for exploration and learning since stochasticity is encapsulated in a higher control level. This property is particularly relevant in critical robotic and control applications where the interpretability of the target policy is a desiderata in order to avoid unpredictable behaviors during the operational phase. Another advantage of PGPE is that no assumption on the differentiability of the policy w.r.t. to the parameters $\boldsymbol{\theta}$ has to be enforced. As a consequence, [7] has shown that PGPE is able to reduce the variance in the gradient estimate w.r.t. to REINFORCE method.

These advantages are obtained by replacing the exploration in the state–action space ($u_t = \pi(x_t) + \epsilon_t$) with an exploration in the policy space ($\boldsymbol{\theta}_t = \boldsymbol{\theta}_t + \epsilon_t$), where ϵ_t is an exploration

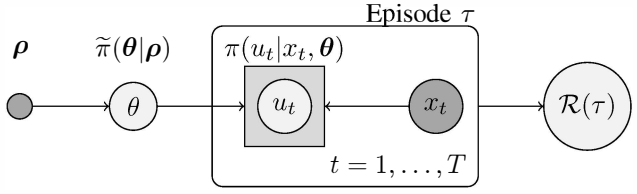


Fig. 1: PGPE hierarchical structure

noise. PGPE introduces the concept of upper–level distribution, i.e., a parametrized distribution $\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho})$, that selects the parameters of the control policy $\pi(u|x,\boldsymbol{\theta})$. This setting configures a hierarchical structure where the actual control policy is the lower–level [1]. Graphical representation of the hierarchy is given in Figure 1. In this way, when deterministic control policies are exploited, the variance of trajectories is given only by the intrinsic variance of the MDP.

Instead of directly finding the policy parameters $\boldsymbol{\theta}$, the goal of PGPE is to find the optimal hyperparameters $\boldsymbol{\rho}^* \in \arg\max_{\boldsymbol{\rho}\in P} \mathcal{J}(\boldsymbol{\rho})$, where $\mathcal{J}(\boldsymbol{\rho}) = \mathbb{E}_{\boldsymbol{\theta}\sim\widetilde{\pi}(\cdot|\boldsymbol{\rho})}[J_D(\boldsymbol{\theta})]$. According to gradient–based optimization the hyperparameters $\boldsymbol{\rho}$ are updated following the gradient direction $\boldsymbol{\rho}' = \boldsymbol{\rho} + \alpha\nabla_{\boldsymbol{\rho}}\mathcal{J}(\boldsymbol{\rho})$, where

$$\nabla_{\boldsymbol{\rho}}\mathcal{J}(\boldsymbol{\rho}) = \int_\Theta \widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho})\nabla_{\boldsymbol{\rho}}\log\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho})J_D(\boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta}.$$

The above equation shows that the requirement on the differentiability of the policy model is dropped in favor of the differentiability of the upper–level distribution $\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho})$.

III. Second order PGPE

The PGPE scheme can be simply extended to higher order methods. In the standard policy–gradient framework, the computation of the Hessian matrix of the policy performance $J_D(\boldsymbol{\theta})$ requires the knowledge of the derivative of the Q–function that is difficult to estimate since it requires the knowledge of the transition model [13]. In a recent work, Furmston et al. [14] have provided a Hessian estimate that is able to overcome such limitation by exploiting the trajectory based definition. The main issue related to this approach is that it requires stochastic policies (since it is just an extension to the second order derivatives of the standard policy gradient) that lead to a potentially high variability and error prone estimation of the Hessian matrix. However, the trajectory–based formulation of the PGPE allows a simple derivation of the Hessian matrix that allows to exploit deterministic policies. Moreover, since we are optimizing a parameter vector that does not directly influence the transition or reward model, the knowledge and differentiability of these terms is not required.

Theorem III.1. *Given an MDP $\mathcal{M}$, an upper–level distribution $\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho})$ and a parametric control policy $\pi(u|x,\boldsymbol{\theta})$, the $(d \times d)$ Hessian matrix of the expected γ–discounted return $\mathcal{J}(\boldsymbol{\rho})$ is given by*

$$H_{\boldsymbol{\rho}}\mathcal{J}(\boldsymbol{\rho}) = \int_\Theta \widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho})\Big(\nabla_{\boldsymbol{\rho}}\log\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho})(\nabla_{\boldsymbol{\rho}}\log\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho}))^T + H_{\boldsymbol{\rho}}\log\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho})\Big)J_D(\boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta}.$$

Since the following integral is always zero[1]

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{\theta}}\left(\nabla_{\boldsymbol{\rho}}\log\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho})\nabla_{\boldsymbol{\rho}}\log\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho})^{\mathrm{T}} + H_{\boldsymbol{\rho}}\log\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho})\right)\\
&= \int_\Theta \nabla_{\boldsymbol{\rho}}\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho})\nabla_{\boldsymbol{\rho}}\log\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho})^{\mathrm{T}} + \widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho})H_{\boldsymbol{\rho}}\log\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho})\mathrm{d}\boldsymbol{\theta}\\
&= \int_\Theta \nabla_{\boldsymbol{\rho}}\left(\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho})\nabla_{\boldsymbol{\rho}}\log\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho})\right)\mathrm{d}\boldsymbol{\theta}\\
&= \int_\Theta \nabla_{\boldsymbol{\rho}}\left(\nabla_{\boldsymbol{\rho}}\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho})\right)\mathrm{d}\boldsymbol{\theta} = H_{\boldsymbol{\rho}}(1) = 0,
\end{aligned}$$

a baseline b that does not depend on the parameters $\boldsymbol{\theta}$ can be added to the gradient formula

$$H_{\boldsymbol{\rho}}\mathcal{J}(\boldsymbol{\rho}) = \int_\Theta \widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho})\Big(\nabla_{\boldsymbol{\rho}}\log\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho})(\nabla_{\boldsymbol{\rho}}\log\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho}))^{\mathrm{T}} + H_{\boldsymbol{\rho}}\log\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho})\Big)\left(J_D(\boldsymbol{\theta}) - b\right)\mathrm{d}\boldsymbol{\theta}. \quad (1)$$

Section IV will derive the optimal baseline b_H^* for the PGPE Hessian as the term that minimizes the variance of the Hessian estimate.

The best way to exploit the information about the local curvature of the objective function is to follow Newton direction. Consider the following update rule: $\boldsymbol{\rho}' = \boldsymbol{\rho} + \Delta\boldsymbol{\rho}$. The Newton step represents the amount of information that must be added to the actual point $\boldsymbol{\rho}$ so that the quadratic (second–order Taylor series) approximation of the objective function

$$\mathcal{J}(\boldsymbol{\rho}') \approx \mathcal{J}(\boldsymbol{\rho}) + \Delta\boldsymbol{\rho}^{\mathrm{T}}\nabla_{\boldsymbol{\rho}}\mathcal{J}(\boldsymbol{\rho}) + \frac{1}{2}\Delta\boldsymbol{\rho}^{\mathrm{T}}H_{\boldsymbol{\rho}}\mathcal{J}(\boldsymbol{\rho})\Delta\boldsymbol{\rho}$$

is maximized. If $H_{\boldsymbol{\rho}}\mathcal{J}(\boldsymbol{\rho})$ is not singular, then $\left(H_{\boldsymbol{\rho}}\mathcal{J}(\boldsymbol{\rho})\right)^{-1}$ exists and the Newton step is defined as:

$$\Delta\boldsymbol{\rho} = -\left(H_{\boldsymbol{\rho}}\mathcal{J}(\boldsymbol{\rho})\right)^{-1}\nabla_{\boldsymbol{\rho}}\mathcal{J}(\boldsymbol{\rho}).$$

We name this algorithm as Newton PGPE (N-PGPE).

The main advantage of the Newton direction is that it is associated to a step size equal to 1. Newton methods are globally convergent[2] if the Hessian matrix has a bounded condition number and is negative definite over all the iterations, and if the step lengths satisfy the Wolfe conditions [12]. Under these assumptions, they enjoy quadratic convergence rate near the stationary point leading to faster convergence rate than gradient ascent. Global convergence can be achieved also if each search direction satisfies the Zoutendijk condition [12], that is, the search directions are never too close to orthogonality with the gradient. Latter conditions are stronger, but easier to check in RL settings than Wolfe conditions.

Several numerical algorithms have been presented in the literature in order to mitigate such limitations. For a complete review we refer to [12].

[1] We made use of the equivalence $(x\nabla_y\log(x) = \nabla_y x)$ in the derivation.

[2] Global convergence refers to the property of the algorithm to converge to a stationary point from any point of the domain.

IV. VARIANCE REDUCTION

In this section we investigate the sample–based estimation of the terms involved in the N-PGPE. This section extends the variance analysis performed in [7] to second–order methods. As mentioned in the previous sections, both the gradient and the Hessian formula turned to be invariant under the addition of a constant baseline, yielding to the following modified estimates

$$\nabla_{\boldsymbol{\rho}}\widehat{\mathcal{J}}(\boldsymbol{\rho}, b_{\nabla}) = \frac{1}{N}\sum_{n=1}^{N} \nabla_{\boldsymbol{\rho}} \log \widetilde{\pi}\left(\boldsymbol{\theta}^{(n)}|\boldsymbol{\rho}\right)\left(\mathcal{R}\left(\tau^{(n)}\right) - b_{\nabla}\right)$$

$$H_{\boldsymbol{\rho}}\widehat{\mathcal{J}}(\boldsymbol{\rho}, b_{H}) = \frac{1}{N}\sum_{n=1}^{N} \Big(\nabla_{\boldsymbol{\rho}} \log \widetilde{\pi}(\boldsymbol{\theta}^{(n)}|\boldsymbol{\rho}) \nabla_{\boldsymbol{\rho}} \log \widetilde{\pi}(\boldsymbol{\theta}^{(n)}|\boldsymbol{\rho})^{\mathrm{T}} + H_{\boldsymbol{\rho}} \log \widetilde{\pi}(\boldsymbol{\theta}^{(n)}|\boldsymbol{\rho})\Big)\left(\mathcal{R}\left(\tau^{(n)}\right) - b_{H}\right), \tag{2}$$

where the pairs $\left(\boldsymbol{\theta}^{(n)}, \tau^{(n)}\right)$ are extracted independently. Note that the expected γ–discounted return $J_D(\boldsymbol{\theta}^{(n)})$ has been approximated by means of a a single trajectory via $\mathcal{R}(\tau^{(n)})$ [1]. As a consequence, the PGPE implementation requires to generate at every iteration a data set $\mathcal{D} = \left\{\boldsymbol{\theta}^{(n)}, \tau^{(n)}\right\}_{n=1}^{N}$.

Unfortunately we cannot obtain an unbiased estimate of the Newton direction $\Delta\boldsymbol{\rho} = -\left(H_{\boldsymbol{\rho}}\mathcal{J}(\boldsymbol{\rho})\right)^{-1}\nabla_{\boldsymbol{\rho}}\mathcal{J}(\boldsymbol{\rho})$ from a single data set $\mathcal{D}$. This difficult estimation problem—estimation of product of dependent terms—has arised several times in RL (e.g., [22]) and can be overcome using double sampling. Precisely, we need two independent data sets $\mathcal{D}_H$ and $\mathcal{D}_{\nabla}$ of the form $\left\{\boldsymbol{\theta}^{(n)}, \tau^{(n)}\right\}_{n=1}^{N}$ for Hessian and gradient estimates, respectively.

Concerning baselines, while the optimal baseline b^*_{∇} for the gradient estimate has been provided in [7], here we derive the optimal baseline for the Hessian estimate as the term that minimizes the variance

$$b^*_H = \arg\min_{b} \mathbf{Var}\left(H_{\boldsymbol{\rho}}\widehat{\mathcal{J}}(\boldsymbol{\rho}, b)\right),$$

where $\mathbf{Var}$ is the generalization of the variance to a random matrix.[3] The following sections report the analysis in the general and Gaussian upper–level distribution cases.

A. General case

The following theorem gives the optimal baseline for the Hessian estimate under no assumption on the upper–level distribution.

[3]Let $\mathbf{X}$ be a $(p \times q)$ matrix whose components $\mathbf{X}_{ij}$ are random variables. We define the variance of the matrix as the variance of its vectorization [7], i.e., the sum of the variances of the individual components

$$\mathbf{Var}(\mathbf{X}) = \sum_{i=1}^{p}\sum_{j=1}^{q}\mathbf{Var}(\mathbf{X}_{ij}) = \sum_{k=1}^{p\cdot q}\mathbf{Var}(\mathbf{y}_k),$$

where $\mathbf{y} = vec(\mathbf{X})$ is the pq–dimensional vectorization of $\mathbf{X}$ obtained by stacking its columns.

Theorem IV.1. *If $\boldsymbol{\theta}_1$, $\boldsymbol{\theta}_2$,...,$\boldsymbol{\theta}_N$ are i.i.d., the optimal baseline for the Hessian estimate $H_{\boldsymbol{\rho}}\widehat{\mathcal{J}}(\boldsymbol{\rho}, b_H)$ in (2) is*

$$b^*_H = \frac{\mathbb{E}_{\boldsymbol{\theta}\sim\widetilde{\pi}(\cdot|\boldsymbol{\rho})}\left[\widehat{J}_D(\boldsymbol{\theta})\|\overline{\mathbf{g}}(\boldsymbol{\theta})\|_2^2\right]}{\mathbb{E}_{\boldsymbol{\theta}\sim\widetilde{\pi}(\cdot|\boldsymbol{\rho})}\left[\|\overline{\mathbf{g}}(\boldsymbol{\theta})\|_2^2\right]},$$

where $\overline{\mathbf{g}}(\boldsymbol{\theta}) = vec(\mathbf{G}(\boldsymbol{\theta}))$ is the vectorization of $\mathbf{G}(\boldsymbol{\theta})$ and

$$\mathbf{G}(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\rho}} \log \widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho}) \nabla_{\boldsymbol{\rho}} \log \widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho})^{T} + H_{\boldsymbol{\rho}} \log \widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho}).$$

Given a generic baseline b, the variance reduction obtained through the optimal baseline b^*_H is

$$\mathbf{Var}(H_{\boldsymbol{\rho}}\widehat{\mathcal{J}}(\boldsymbol{\rho}, b)) - \mathbf{Var}(H_{\boldsymbol{\rho}}\widehat{\mathcal{J}}(\boldsymbol{\rho}, b^*_H)) = \frac{(b - b^*_H)^2}{N}\mathbb{E}_{\boldsymbol{\theta}\sim\widetilde{\pi}(\cdot|\boldsymbol{\rho})}\left[\overline{\mathbf{g}}(\boldsymbol{\theta})^T\overline{\mathbf{g}}(\boldsymbol{\theta})\right].$$

First, it is important to underline that, from a practical point of view, the optimal baseline can be computed in real time without requiring any additional storage of information, as happens for the gradient baseline b^*_{∇}. Second, considering a baseline different from the optimal one makes the variance to increase quadratically.

B. Gaussian case

The upper–level distribution $\widetilde{\pi}(\boldsymbol{\theta}|\boldsymbol{\rho})$ is usually modeled as a Normal distribution $\mathcal{N}(\boldsymbol{\theta}|\mu, \Sigma)$ [1], [23], while the control policy $u = \pi(x|\boldsymbol{\theta})$ is typically describe by a deterministic rule. Here we consider only the mean $\mu \in \mathbb{R}^d$ to be the parameter to be tuned (i.e., $\boldsymbol{\rho} = \mu$) while the covariance $\Sigma = diag(\sigma_1^2, \ldots, \sigma_d^2)$ is a constant matrix. By keeping variance fixed the exploration is fixed to a certain level. However it would be straightforward to incorporate, at the cost of longer analytic derivations, the covariance matrix as a parameter, and several approaches have been presented in literature (e.g., diagonal covariance matrix with sigmoid representation or covariance factorization via Cholesky decomposition [24]). Moreover the complexity of the Hessian estimation would be $\mathcal{O}(d^4)$ in case of a full covariance matrix, but this is true for any second order method (e.g., natural gradient). Under these settings several results on the Hessian estimate can be derived. Below we consider the following assumption

Assumption IV.1. *The reward model is positively bounded for any state–action pair (x, u) by $\mathcal{R}(x, u) \in [R_{\min}, R_{\max}]$, with $0 < R_{\min} < R_{\max}$.*

The previous assumption can be easily enforced by scaling the reward function without introducing any bias in the MDP solution.

First, we analyze the variance of the Hessian estimate in N-PGPE.

Lemma IV.2. *Given a normal meta–distribution $\widetilde{\pi}(\boldsymbol{\theta}|\mu) = \mathcal{N}(\boldsymbol{\theta}|\mu, \Sigma)$ where μ is the parameter vector and $\Sigma = \mathrm{diag}(\sigma_1^2, \ldots, \sigma_d^2)$ is a constant diagonal matrix, the variance of the Hessian estimate is bounded by*

$$\mathbf{Var}(H_{\boldsymbol{\rho}}\widehat{\mathcal{J}}(\boldsymbol{\rho}, \mathbf{0})) \leq \frac{R_{\max}^2(1-\gamma^T)^2}{N(1-\gamma)^2}\sum_{i=1}^{d}\sum_{j=i}^{d}\frac{1}{\sigma_i^2\sigma_j^2}.$$

This means that the variance is proportional to the squared bound of the immediate reward, but decreases exponentially with the length T of the episode. Clearly it is inversely proportional to the number of samples N.

As the reader may have noticed, Lemma IV.2 is given for the baseline–free Hessian estimate. The following lemma exploits such result to derive an upper bound to the variance of the Hessian estimate with optimal baseline b_H^*.

Lemma IV.3. *Consider Assumption IV.1. Given a normal meta–distribution $\widetilde{\pi}(\boldsymbol{\theta}|\mu) = \mathcal{N}(\boldsymbol{\theta}|\mu, \Sigma)$ where μ is the parameter vector and $\Sigma = \mathrm{diag}(\sigma_1^2, \dots, \sigma_d^2)$ is a constant diagonal matrix, the variance of the Hessian estimate with optimal baseline b_H^* is bounded by*

$$\begin{aligned}&\mathbf{Var}(H_{\boldsymbol{\rho}}\widehat{\mathcal{J}}(\boldsymbol{\rho}, b_H^*)) \\ &\leq \frac{(1-\gamma^T)^2}{N(1-\gamma)^2}\left(R_{\max}^2 - R_{\min}^2\right)\sum_{i=1}^{d}\sum_{j=i}^{d}\frac{1}{\sigma_i^2\sigma_j^2}.\end{aligned}$$

The result in the previous Lemma combined with the Chebyshev's inequality allows to provide a high–probability upper bound to the Hessian approximation error using the N-PGPE estimator given in Equation (2).

Theorem IV.4. *Consider Assumption IV.1. Given a normal meta–distribution $\widetilde{\pi}(\boldsymbol{\theta}|\mu) = \mathcal{N}(\boldsymbol{\theta}|\mu, \Sigma)$ where μ is the parameter vector and $\Sigma = \mathrm{diag}(\sigma_1^2, \dots, \sigma_d^2)$ is a constant diagonal matrix, using the following number of samples $(\boldsymbol{\theta}^{(n)}, \tau^{(n)})$:*

$$N = \frac{(1-\gamma^T)^2}{\epsilon^2\delta(1-\gamma)^2}\left(R_{\max}^2 - R_{\min}^2\right)\sum_{i=1}^{d}\sum_{j=i}^{d}\frac{1}{\sigma_i^2\sigma_j^2},$$

the Hessian estimate $H_{\boldsymbol{\rho}}\widehat{\mathcal{J}}(\boldsymbol{\rho}, b_H^*)$ generated by N-PGPE is such that with probability $1-\delta$

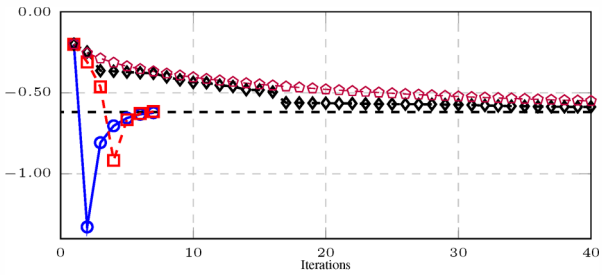
$$\left|H_{\boldsymbol{\rho}}\widehat{\mathcal{J}}(\boldsymbol{\rho}, b_H^*) - H_{\boldsymbol{\rho}}\mathcal{J}(\boldsymbol{\rho}, b_H^*)\right| \leq \epsilon.$$

V. Experiments

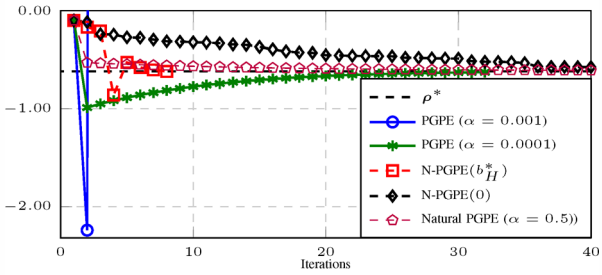
In this section, results related to the numerical simulations of the proposed algorithms, in two continuous domains, are presented. While the Linear–Quadratic regulator (LQR) is a standard benchmark domain, the second domain, corresponding to a full dynamical model of a quadrotor, aims at showing the capabilities of the algorithm in a complex domain. When necessary, we have used standard optimization techniques to preserve the correct direction of the Newton step [12].

A. Linear-Quadratic Regulator

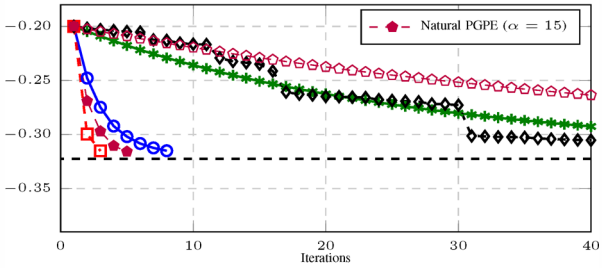
The first case of study is a discrete-time Linear-Quadratic regulator (LQR) with continuous state and action spaces [2]. The LQR problem is characterized by a transition model $x_{t+1} = Ax_t + Bu_t$, a deterministic policy $u_t = \boldsymbol{\theta} \cdot x_t$ and a quadratic reward $r_t = -Qx_t^2 - Ru_t^2$. Since the aim of this domain is mere instructive we have limited our attention to the scalar scenario. If not explicitly given, the following parameters are used: $A = B = Q = R = 1$.



(a) Standard domain



(b) Costly domain $Q = R = 2$



(c) Contractive domain $A = B = 0.6$

Fig. 2: Parameters $\boldsymbol{\rho}$ of PGPE and N-PGPE as function of iterations, under different domain parametrizations. Where not explicitly stated, parameters are set to default values. Dashed horizontal line denotes the optimal parameter $\boldsymbol{\rho}^*$. N-PGPE(b_H^*) and N-PGPE(0) are used to denote Newton PGPE with and without baseline.

The advantage of the hierarchical structure resides in the possibility to use deterministic policies moving the stochasticity into the upper–level policy $\pi(\boldsymbol{\theta}|\boldsymbol{\rho})$. Here we have chosen a Gaussian upper–level distribution $\pi(\boldsymbol{\theta}|\boldsymbol{\rho}) = \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\rho}; \Sigma)$ where $\boldsymbol{\theta}$ is the mean value and $\Sigma = 0.01^2$ is a constant variance.

The values of the parameters used for all the experiments are the following ones: $\gamma = 0.99$ and the initial state $x_0 = 10$. For the estimation of the gradient and Hessian values a total of 2000 policies are evaluated based on a single episode of 30 steps. Starting from $\boldsymbol{\rho}_0 = -0.1$, experiments are ended when an 0.01–accurate[4] upper–level distribution is found.

As mentioned in Section III, one of the advantages of the N-PGPE methods is the absence of the step size α. Here we show empirically that N-PGPE enjoys an almost

[4]Accuracy is evaluated w.r.t the optimal parameters $\boldsymbol{\rho}^*$ which can be computed analytically.

#Policies	Iterations N-PGPE(b^*_H)	N-PGPE(0)	Natural PGPE
100	29.7 ± 6.22	271.5 ± 8.73	11.66 ± 1.17
500	18.5 ± 4.59	128.4 ± 6.21	12.2 ± 0.66
1000	15.7 ± 2.70	99.0 ± 6.84	12.34 ± 0.42
5000	8.5 ± 2.23	52.1 ± 3.91	12.54 ± 0.16
10000	5.8 ± 1.19	37.4 ± 1.96	12.58 ± 0.15
20000	4.4 ± 0.07	35.3 ± 4.37	12.58 ± 0.14

TABLE I: Convergence speed in LQR scenario. The table reports the number of iterations required by the algorithms, starting from $\boldsymbol{\rho}_0 = -0.1$, to learn the optimal policy parameter $\boldsymbol{\rho}^* = -0.6037$ with an accuracy of 0.01, for different data set dimensions. Data are averaged over 100 runs and 95%–confidence intervals are reported.

"invariant" trend under domain changes. In contrast, PGPE and Natural PGPE require to tune the step size for the different domain parametrizations. Figure 2 compares the algorithms under slightly different domain parametrizations in a single explanatory run. While we have considered the state–of–art PGPE and Natural PGPE algorithms[5], we have compared both the variants of N-PGPE, i.e., with and without baseline. Figure 2a shows that, using a hand–tuned constant step size for the PGPE, it is possible to outperform the N-PGPE algorithm. However, the tuned step size is good only for the current domain and as soon as the domain changes, a new step size must be identified. This is clear in Figure 2b where the step size for the standard domain ($\alpha = 0.001$ in Figure 2a) leads to divergence of the PGPE algorithm when the reward is doubled. Better performances are obtained by reducing the step size to 0.0001. In the case in which the dynamics of the system are changed (Figure 2c), PGPE is less influenced by the initial step size since the model is slower than the original one, but better trends can be obtained using larger values. Natural PGPE showed similar shortcomings with respect to the choice of the step size. Given $\alpha = 0.5$, Natural PGPE performed very similar to N-PGPE(0) in the first and third domain (namely, Figures 2a and 2c), whereas in the second parametrization (Figure 2b) its behaviour mimics N-PGPE(b^*_H), but only in the initial iterations. However, a finer tuning of the step size α (set to 15) allowed Natural PGPE to show a rate of convergence very close to N-PGPE(b^*_H) in the contractive domain. Nevertheless, employing the same value for α led Natural PGPE to diverge in the other domains. Concerning Newton approaches, this experiment underlines the relevance of the introduction of the baseline. In our tests the N-PGPE with baseline has proven to be always faster and more stable than the version without baseline.

We also tested the performance of the algorithms under different data set dimensions. In particular, since both the domain and the policy are deterministic, we have fixed the number of episodes and steps used to estimate $J_D(\boldsymbol{\theta}^{(n)})$ to 1 and 30, respectively. For the Natural PGPE, we manually tuned the step size so as to achieve the better performance in terms of convergence rate ($\alpha = 3$). The free parameter is the number of samples drawn from the upper–level policy. As shown in Table I the Natural PGPE has proved to be more robust than the Newton–based approaches, but only after a fine tuning of the parameter α. The experiments have also proved that N-PGPE(b^*_H) is much more robust than the N-PGPE(0) when few policies are drawn, obtaining good performance with only 100 samples. The gap decreases as the number of policies increases, but it still remains significant with an extremely high number (20000) policy samples. Clearly second–order approaches require more samples than first–order ones for a correct estimate of the search direction, due to the presence of additional d^2 values to be estimated (i.e., the Hessian and the Fisher Information matrices).

[5]It is well known that the best performances of the PGPE are obtained using the optimal baseline.

B. Quadrotor

The second case study is an under actuated 6 DoF quadrotor [25], whose mathematical model is described by the following differential equations:

$$\begin{cases} \ddot{x} &= \sum_{i=1}^{4} b\,\Omega_i^2 \dfrac{\cos\psi\cos\phi\sin\theta + \sin\phi\sin\psi}{m} \\ \ddot{y} &= \sum_{i=1}^{4} b\,\Omega_i^2 \dfrac{\sin\psi\cos\phi\sin\theta - \sin\phi\cos\psi}{m} \\ \ddot{z} &= \sum_{i=1}^{4} b\,\Omega_i^2 \dfrac{\cos\theta\cos\phi}{m} - g \\ \dot{p} &= \dfrac{I_y - I_z}{I_x} qr + \frac{b\,l}{I_x}(\Omega_4^2 - \Omega_2^2) - \frac{J_m}{I_x}\, q\,\Omega_R \\ \dot{q} &= \dfrac{I_z - I_x}{I_y} pr + \frac{b\,l}{I_y}(\Omega_3^2 - \Omega_1^2) - \frac{J_m}{I_y}\, p\,\Omega_R \\ \dot{r} &= \dfrac{I_x - I_y}{I_z} pq + \frac{d}{I_z}(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2) \\ \dot{\phi} &= p + q\sin\phi\tan\theta + r\cos\phi\tan\theta \\ \dot{\theta} &= q\cos\theta - r\sin\theta \\ \dot{\psi} &= q\frac{\sin\phi}{\cos\theta} + r\frac{\cos\phi}{\cos\theta} \end{cases} \tag{3}$$

These equations are derived based on Newton–Euler formalism. Let $\mathcal{I}$ be the right–hand inertial frame and $\mathcal{B}$ the right–hand body–fixed frame. The tuple $(x, y, z) \in \mathcal{I}$ is the position of the center of mass of the quadrotor and the Euler angles $(\phi, \theta, \psi) \in \mathcal{I}$ describe the orientation of the rigid body.

In the system (3), $I = \mathrm{diag}(I_x, I_y, I_z) \in \mathcal{B}$ and m are the constant inertia matrix and the overall mass of the flyer, l is the horizontal distance from the propeller center to CoG and g denotes the acceleration due to gravity. Furthermore, $\omega = [p, q, r] \in \mathcal{B}$ is the body angular velocity, Ω_i represents the i–th rotor angular velocity, Ω_R is the overall residual rotor angular speed and J_m is the total rotational moment of inertia around the propeller axes. Finally, b and d are the thrust and drag coefficients, respectively.

The quadrotor dynamics (3) is a highly non-linear, Multi–Input Multi–Output (MIMO), strongly coupled and under actuated. A common approach for quadrotors consists in a hierarchical control architecture: neglecting the control of the rotor rotational speed, the lowest level is in control of vehicle attitude, and the top level is in control of position along a trajectory. Hence, the position control loop provides

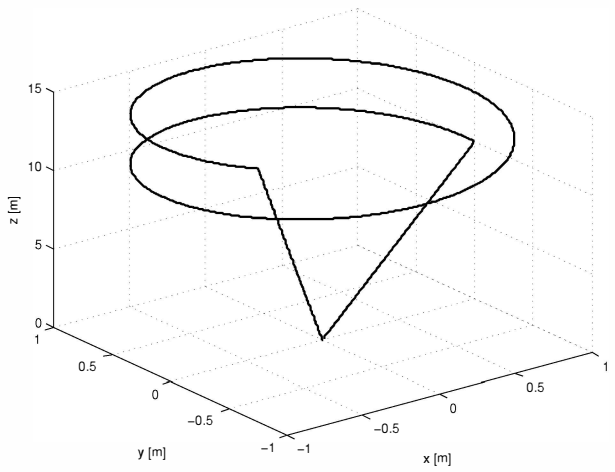


Fig. 3: Nominal trajectory $\tilde{\xi}(t)$.

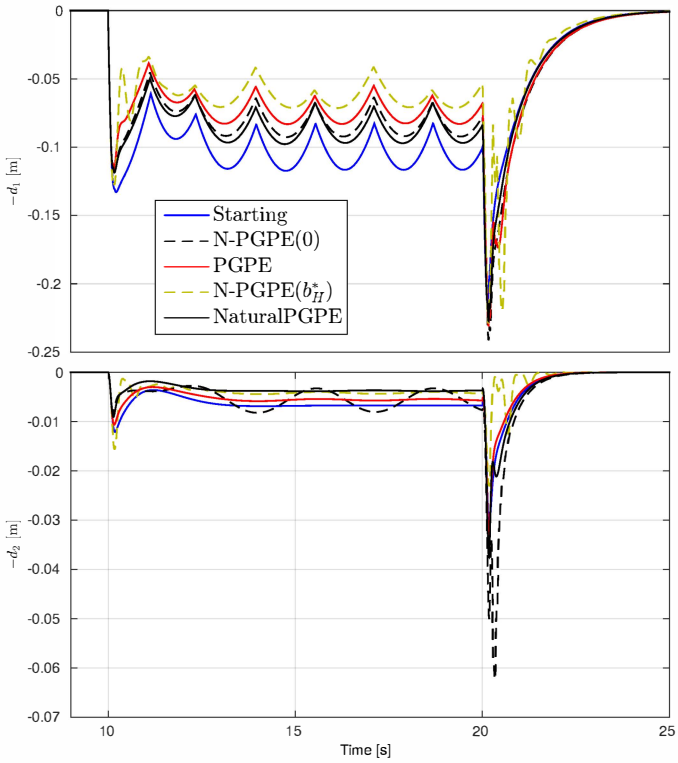


Fig. 4: Reward signal along the quadrotor trajectory, both with d_1 (top) and d_2 (bottom) distance functions. The simulation lasts $40s$, only the relevant part is shown in figure.

attitude set points for the attitude controller, resulting in nested feedback loops.

More precisely, the six degrees of freedom of the system, namely the position (x, y, z) and the orientation (ϕ, θ, ψ), are managed by six PD regulators which are obtained by means of a classical design in frequency domain. The gains of these six regulators represents the upper–level policy parameters $\boldsymbol{\rho} = [k_x, k_y, k_z, k_\phi, k_\theta, k_\psi]^\mathrm{T}$ to be optimized. Since the main

	PGPE	N-PGPE(b^*_H)	N-PGPE(0)	NaturalPGPE
$\mathcal{R}_{d_1}(\tau)$	-187	-152	-204.9	-192
$\mathcal{R}_{d_2}(\tau)$	-13.58	-11.7	-12.9	-11.5

TABLE II: Cumulative return associated to an entire trajectory. The values associated to the initial parameters $\boldsymbol{\rho}_0$ are $\mathcal{R}_{d_1}(\tau) = -225.6$ and $\mathcal{R}_{d_2}(\tau) = -14.45$, respectively for the distance function $d_1(\cdot,\cdot)$ and $d_2(\cdot,\cdot)$.

task of the quadrotor is to follow the desired trajectory $\tilde{\xi}(t) = (\tilde{x}(t), \tilde{y}(t), \tilde{z}(t))$, $t \geq 0$, in Figure 3, we set the one–step reward function as the distance between the measured trajectory $\xi(t) = (x(t), y(t), z(t))$ and the reference one, i.e. $r_t = -d(\xi_t, \tilde{\xi}_t)$.

For the numerical experiments, the following parameters have been employed. The initial mean values of the upper–level distribution have been derived from an empirical study, and they are $\boldsymbol{\rho}_0 = [1, -1, 80, 0.8, 0.8, 1.6]^\mathrm{T}$, whereas the matrix $\Sigma = \mathbf{I} \cdot 0.01^2$ is the constant covariance. A number of 4000 policies are extracted from the upper–level distribution for the gradient, the Hessian and the Fisher matrices estimation, and a maximum of 100 iterations are allowed to achieve algorithm convergence. The dynamical system (3) is integrated by using Dormand-Prince integration method.

Table II reports the performance obtained by the quadrotor when the gains of PD regulators are optimized by means of the proposed algorithms.

In this test we considered two different distance measures, namely the 1–norm distance, $d_1(x, y) = \|x - y\|_1$, and the squared 2–norm distance $d_2(x, y) = \|x - y\|_2^2$. While all the algorithms improve the reward value associated to the initial parameters $\boldsymbol{\rho}_0$, the N-PGPE(b^*_H) outperforms both the N-PGPE(0) and the Natural PGPE, as well as the PGPE. A pictorial representation of this is shown in Figure 4.

Nevertheless, the step size $\alpha = 10^{-4}$ employed by PGPE when the d_1 distance function is used leads to unsatisfactory results when the reward function employs the distance d_2 (the upper-level distribution remains essentially the same). Indeed, a new step size $\alpha = 10^{-2}$ has been implemented to reach better results (see Table II). Also Natural PGPE required a tuning phase for the parameter α, respectively, set to 1 and 5 for the distance functions d_1 and d_2.

On the other hand, both the variants of N-PGPE (with and without baseline) are able to adapt to the change of the reward function without compromising their performances. Still, the introduction of the baseline has proved to make the difference in the convergence speed and in the quality of the obtained solution.

VI. Conclusions

While RL literature has mainly focused on standard and natural gradient methods, this work provides a wide analysis of Newton methods in RL. As we know, the only notable exception is the work by Furmston [14]. In Sections III and IV we have derived the hierarchical formulation of the Hessian matrix for the PGPE and shown (theoretically and empirically) that the variance of the estimate can be reduced by adding

a constant baseline, as happens for policy–gradient methods. Second–order methods mitigate the problems of the tuning of the step size that affects the gradient methods. Experiments confirm the intrinsic capacity of adaptation of N-PGPE to changes in the transition or reward model. However this advantage comes to the price of double sampling for obtaining an unbiased estimate of the Newton direction. Other techniques have been presented in literature in order to overcome this difficulty, e.g., two timescale algorithms. It will be interesting to analyze the applicability of such methods to the estimation of the Newton direction.

Finally, since the estimation of the Hessian matrix is known to be tough [12], we think that will be interesting to investigate Quasi– or modified Newton solution concepts. The former aims to provide robust optimization algorithms that leverage on the theoretical properties of the Hessian, the latter removes the problem of computing the Hessian by using an approximation based on change in the gradient between iterations.

REFERENCES

[1] M. P. Deisenroth, G. Neumann, and J. Peters, "A survey on policy search for robotics," *Foundations and Trends in Robotics*, vol. 2, no. 1-2, pp. 1–142, 2013.

[2] J. Peters and S. Schaal, "Reinforcement learning of motor skills with policy gradients," *Neural Networks*, vol. 21, no. 4, pp. 682–697, 2008.

[3] D. P. Bertsekas, *Dynamic programming and optimal control. Volume I*, ser. Athena Scientific optimization and computation series. Belmont, Mass. Athena Scientific, 2005.

[4] J. Kober and J. Peters, "Policy search for motor primitives in robotics," *Machine Learning*, vol. 84, no. 1-2, pp. 171–203, 2011.

[5] E. Theodorou, J. Buchli, and S. Schaal, "A generalized path integral control approach to reinforcement learning," *The Journal of Machine Learning Research*, vol. 11, pp. 3137–3181, 2010.

[6] F. Sehnke, C. Osendorfer, T. Rückstieß, A. Graves, J. Peters, and J. Schmidhuber, "Parameter-exploring policy gradients," *Neural Networks*, vol. 23, no. 4, pp. 551–559, 2010.

[7] T. Zhao, H. Hachiya, G. Niu, and M. Sugiyama, "Analysis and improvement of policy gradient estimation," *Neural Networks*, vol. 26, no. 0, pp. 118–129, 2012.

[8] F. Sehnke, C. Osendorfer, T. Rueckstiess, A. Graves, J. Peters, and J. Schmidhuber, "Policy gradients with parameter-based exploration for control," in *Proceedings of the International Conference on Artificial Neural Networks (ICANN)*, 2008.

[9] M. Pirotta, G. Manganini, L. Piroddi, M. Prandini, and M. Restelli, "A particle-based policy for the optimal control of Markov decision processes," in *IFAC World Congress 2014*, Cape Town, South Africa, August 2014.

[10] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3–4, pp. 229–256, 1992.

[11] M. Pirotta, M. Restelli, and L. Bascetta, "Adaptive step-size for policy gradient methods," in *Advances in Neural Information Processing Systems 26*, 2013.

[12] J. Nocedal and S. Wright, *Numerical optimization*, 2nd ed. New York: Springer, 2006.

[13] S. Kakade, "A natural policy gradient," in *Advances in Neural Information Processing Systems 14, NIPS 2001, Vancouver, British Columbia, Canada*. MIT Press, 2001, pp. 1531–1538.

[14] T. Furmston and D. Barber, "A unifying perspective of parametric policy search methods for markov decision processes," in *Advances in Neural Information Processing Systems*, 2012, pp. 2717–2725.

[15] S. Amari and S. Douglas, "Why natural gradient?" in *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, vol. 2, May 1998, pp. 1213–1216 vol.2.

[16] V. Heidrich-Meisner and C. Igel, "Similarities and differences between policy gradient methods and evolution strategies." in *ESANN*. Citeseer, 2008, pp. 149–154.

[17] F. Stulp and O. Sigaud, "Robot skill learning: From reinforcement learning to evolution strategies," *Paladyn. Journal of Behavioral Robotics*, vol. 4, no. 1, pp. 49–61, September 2013.

[18] D. Wierstra, T. Schaul, J. Peters, and J. Schmidhuber, "Natural evolution strategies," in *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on*. IEEE, 2008, pp. 3381–3387.

[19] A. Miyamae, Y. Nagata, I. Ono, and S. Kobayashi, "Natural policy gradient methods with parameter-based exploration for control tasks," in *Advances in Neural Information Processing Systems 23*, 2010, pp. 1660–1668.

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

[21] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, ser. JMLR Proceedings, vol. 32, 2014, pp. 387–395.

[22] D. D. Castro, A. Tamar, and S. Mannor, "Policy gradients with variance related risk criteria," in *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK*, 2012.

[23] F. Sehnke, A. Graves, C. Osendorfer, and J. Schmidhuber, "Multimodal parameter-exploring policy gradients," in 9^{th} *Int. Conf. on Machine Learning and Applications (ICMLA)*, 2010, pp. 113–118.

[24] H. Kimura and S. Kobayashi, "Reinforcement learning for continuous action using stochastic gradient ascent," pp. 288–295, 1998.

[25] S. Bouabdallah, "Design and control of quadrotors with application to autonomous flying," 2007.