Policy Gradient Approaches for Multi–Objective Sequential Decision Making: A Comparison

Simone Parisi, Matteo Pirotta, Nicola Smacchia, Luca Bascetta and Marcello Restelli

Abstract— **This paper investigates the use of policy gradient techniques to approximate the Pareto frontier in Multi–Objective Markov Decision Processes (MOMDPs). Despite the popularity of policy–gradient algorithms and the fact that gradient–ascent algorithms have been already proposed to numerically solve multi–objective optimization problems, especially in combination with multi-objective evolutionary algorithms, so far little attention has been paid to the use of gradient information to face multi-objective sequential decision problems. Three different Multi-Objective Reinforcement-Learning (MORL) approaches are here presented. The first two, called *radial* and *Pareto following*, start from an initial policy and perform gradient–based policy–search procedures aimed at finding a set of non–dominated policies. Differently, the third approach performs a single gradient–ascent run that, at each step, generates an improved continuous approximation of the Pareto frontier. The parameters of a function that defines a manifold in the policy parameter space are updated following the gradient of some performance criterion so that the sequence of candidate solutions gets as close as possible to the Pareto front. Besides reviewing the three different approaches and discussing their main properties, we empirically compare them with other MORL algorithms on two interesting MOMDPs.**

I. Introduction

Many real–world control problems (e.g., economic systems, water resource problems, robotic systems, just to mention a few) are characterized by the presence of multiple, conflicting objectives. Such problems are often modeled as Multi-Objective Markov Decision Processes (MOMDPs), where the concept of optimality typical of MDPs is replaced by the one of *Pareto optimality*. A policy is Pareto optimal if it is not dominated (i.e., it does not perform worse w.r.t. all the objectives) by any other policy. Solving a MOMDP means to find the set of all the Pareto–optimal policies forming the so–called *Pareto front*.

In the last decades, Reinforcement Learning (RL) [1] has been established as an effective and theoretically–grounded framework that allows to solve single–objective MDPs whenever either no (or little) prior knowledge is available about system dynamics, or the dimensionality of the system to be controlled is too high for classical optimal control methods. Despite the successful developments in RL theory and a high demand for multi–objective control applications, Multi–Objective Reinforcement Learning (MORL) is still a relatively young and unexplored research topic. For recent and complete surveys of MORL, we refer the reader to [2].

S. Parisi, M. Pirotta, N. Smacchia, L. Bascetta and M. Restelli are with the Department of Electronics, Information and Bioengineering, Politecnico di Milano, Piazza Leonardo da Vinci, 32, 20133, Milan, Italy (email: {luca.bascetta, matteo.pirotta, marcello.restelli}@polimi.it, {simone.parisi, nicola.smacchia}@mail.polimi.it).

Traditionally, MORL approaches are based on value–function techniques that provides a quality measure for each state–action pair. The main problem of such approaches are: 1) provide quality measure for each state–action pair, 2) bootstrapping may introduce bias in the measure when functional approximation are used, and 3) value functions are usually discontinuous, i.e., difficult to represent with standard functional approximator. Policy gradient–or better, policy search–techniques are able to overcome such limits by directly performing a search in the policy parameter space. In the last years, policy–gradient methods have been established as the most effective RL techniques for complex real–world control problems with continuous, high–dimensional, and partially–observable properties, such as robotic control systems [3]. Given a parameterized policy space, usually designed to incorporate domain knowledge, policy–gradient algorithms update policy parameters along an estimated ascent direction of the expected return. Under some mild assumptions on the *learning rate* used to update the parameters, policy–gradient methods are guaranteed to converge at least to a locally optimal solution. Despite the effectiveness of policy–gradient methods, their application to multi–objective sequential decision problems has been largely overlooked until recently.

In [4] the authors propose two MORL algorithms that exploit policy gradients to build a discrete approximation of the Pareto front. The first, named *radial (RA)*, builds a p–dimensional[1] approximation of the Pareto front by performing gradient–ascent optimization along p different (uniformly spaced) directions within the ascent simplex defined by the convex combination of single-objective gradients. The second approach, named *Pareto–following (PFA)*, starts by performing a single–objective optimization in order to move the solution into the Pareto front and then it moves along the Pareto front by following individual gradient directions. Although such methods proved to be effective in several scenarios, they are not guaranteed to uniformly cover the Pareto front (as all the other gradient approaches).

To overcome this issue a new approach that builds a continuous approximation of the Pareto front was proposed in [5]. The idea is to exploit a gradient–based approach to optimize the parameters of a function that defines a manifold in the policy parameter space so that the corresponding image in the objective space gets as close as possible to the Pareto frontier. The effectiveness of such approach is strongly affected by the metric used to evaluate the quality of the

[1] p is the number of solutions that define the Pareto approximation.

candidate solutions. Several metrics have been proposed in multi–objective optimization literature, but all of them have pros and cons, so that the definition of a general measure for the goodness of a candidate frontier is still an open problem.

In this paper, we review these gradient approaches and perform a deep empirical analysis to show advantages and limitations of each algorithm. In particular, we focus our attention on the sample complexity, meant as the number of simulation steps needed to approximate the Pareto front.

II. Preliminaries

Multi-objective Markov Decision Processes (MOMDPs) are an extension of the MDP model, where several pairs of reward functions and discount factors are defined, one for each objective. Formally, a MOMDP is described by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathbf{R}, \boldsymbol{\gamma}, D \rangle$, where $\mathcal{S} \subseteq \mathbb{R}^n$ is the continuous state space, $\mathcal{A} \subseteq \mathbb{R}^m$ is the continuous action space, $\mathcal{P}$ is a Markovian transition model where $\mathcal{P}(s'|s,a)$ defines the transition density between state s and s' under action a, $\mathbf{R} = [\mathcal{R}_1, \ldots, \mathcal{R}_q]^\intercal$ and $\boldsymbol{\gamma} = [\gamma_1, \ldots, \gamma_q]^\intercal$ are q-dimensional column vectors of reward functions $\mathcal{R}_i : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ and discount factors $\gamma_i \in [0, 1)$, respectively, and D is the initial state distribution from which the initial state is drawn. In MOMDPs, any policy π is associated to q expected returns $\mathbf{J}^\pi = \left[J_1^\pi, \ldots, J_q^\pi\right]$, where

$$J_i^\pi = E\left\{\sum_{t=0}^{H} \gamma_i^t r_i(t+1) | x_0 \sim D, \pi\right\},$$

being $r_i(t+1) = \mathcal{R}_i(s_t, a_t, s_{t+1})$ the i–th immediate reward obtained when state s_{t+1} is reached from state s_t and action a_t, and H the finite or infinite horizon.

Despite what happens in classical MDPs, in MOMDPs a single policy that dominates all the others usually does not exist; in fact, when conflicting objectives are considered, no policy can simultaneously maximize all the objectives. For these reasons, in Multi-Objective Optimization (MOO) a different dominance concept has been defined.

Definition 2.1: Policy π *dominates* policy π', which is denoted by $\pi \succ \pi'$, if:

$$\forall i \in \{1, \ldots, q\}, J_i^\pi \geq J_i^{\pi'} \ \wedge \ \exists i \in \{1, \ldots, q\}, J_i^\pi > J_i^{\pi'}.$$

Definition 2.2: If there is no policy π' such that $\pi' \succ \pi$, the policy π is Pareto–optimal.

In general, there are multiple Pareto–optimal policies. Solving a MOMDP is equivalent to determine the set of Pareto–optimal policies $\Pi^* = \left\{\pi \,|\, \nexists \pi', \pi' \succ \pi\right\}$, that maps to the so-called Pareto frontier $\mathcal{J}^* = \left\{\mathbf{J}^{\pi^*} | \pi^* \in \Pi^*\right\}$.[2]

III. Multi-objective Policy Gradient

In policy–gradient approaches, a parametrized space of policies $\Pi_{\boldsymbol{\theta}} = \left\{\pi_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^d\right\}$ (where $\pi_{\boldsymbol{\theta}}$ is a compact notation for $\pi(a|s, \boldsymbol{\theta})$) is considered.
Given a policy parametrization $\boldsymbol{\theta}$, we assume the policy performance $\mathbf{J} : \Theta \to \mathcal{F}$ to be at least C^2. $\mathbf{J}$ is defined as the expected reward over the space of all possible trajectories $\mathbb{T}$: $\mathbf{J}(\boldsymbol{\theta}) = \int_{\mathbb{T}} p(\tau|\boldsymbol{\theta}) \mathbf{r}(\tau) \mathrm{d}\tau$, where $\tau \in \mathbb{T}$ is a trajectory drawn from density distribution $p(\tau|\boldsymbol{\theta})$ with reward vector $\mathbf{r}(\tau)$ that represents the accumulated expected discounted reward over trajectory τ: $\mathbf{r}_i(\tau) = \sum_{t=0}^{T} \gamma_i^t r_i(t+1)$.

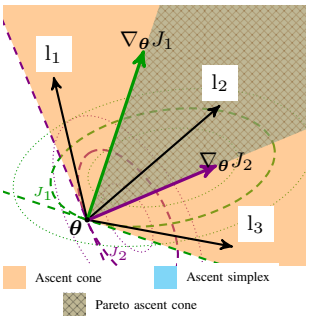


Fig. 1. The ascent cone and simplex in a 2–parameters, 2–objectives problem. The Pareto ascent cone equals the ascent simplex. $\mathbf{l}_2$ dominates $\mathbf{l}_1$ and $\mathbf{l}_3$.

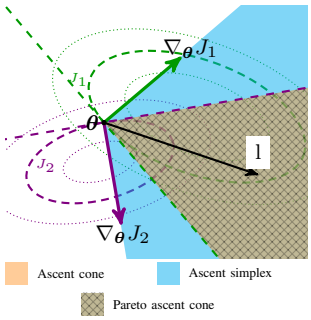


Fig. 2. The ascent cone and simplex in a 2–parameters, 2–objectives problem. The Pareto ascent cone equals the ascent cone. $\mathbf{l}$ represents a Pareto ascent direction.

In MOMDPs for each policy parameter $\boldsymbol{\theta}$, q gradient directions are defined [7]

$$\nabla_{\boldsymbol{\theta}} J_i(\boldsymbol{\theta}) = \int_{\mathbb{T}} \nabla_{\boldsymbol{\theta}} p(\tau|\boldsymbol{\theta}) r_i(\tau) \mathrm{d}\tau = \mathbb{E}\{\nabla_{\boldsymbol{\theta}} \log p(\tau|\boldsymbol{\theta}) r_i(\tau)\},$$

where each direction $\nabla_{\boldsymbol{\theta}} J_i$ is associated to a particular discount factor–reward function pair $< \gamma_i, \mathcal{R}_i >$. As shown in the previous equation, the differentiability of the performance measure is connected to the differentiability of the policy class by: $\nabla_{\boldsymbol{\theta}} \log p(\tau|\boldsymbol{\theta}) = \sum_{k=1}^{T} \nabla_{\boldsymbol{\theta}} \log \pi(a_k|s_k, \boldsymbol{\theta})$.

Given the q gradient vectors, [8] showed that it is possible to define the set of all ascent directions that *simultaneously* improve all the objectives. This set, that we call *Pareto ascent cone*, is the intersection of the *ascent cone* (i.e., the intersection of all positive half spaces) with the *ascent simplex* (defined by the convex combination of single–objective gradients). When the solution $\boldsymbol{\theta}$ is sufficiently distant from the Pareto frontier, gradients are likely to be highly correlated and the directions that lie in the ascent simplex will also lie in the ascent cone (Figure 1). However, as the solution approaches the Pareto frontier, gradient directions become more and more divergent and the width of the ascent cone decreases (Figure 2). A degenerate case is obtained when gradients are coplanar, in this case the ascent cone cannot be defined and the corresponding $\boldsymbol{\theta}$ is a (possibly local) Pareto–optimal solution.

Among all Pareto ascent directions, it is worth to consider the direction that maximizes the minimum improvement among all the individual objective improvements [8], i.e., the smallest ($L2$–norm) Pareto ascent direction. As done in [8, 4], such direction is the solution of a Quadratic Programming (QP) problem, in $d+1$ variables with q inequality constraints, that always admits a unique solution.

[2] As done in [6], we suppose that local Pareto–optimal solutions that are not Pareto–optimal do not exist.

IV. Multi-Objective Policy Gradient Pareto Approximation

This section reviews the MORL gradient algorithms *Radial* (RA) and Pareto–Following (PFA) proposed in [4]. They differ for the strategy followed for the generation of the approximate Pareto front.

A. Radial Algorithm (RA)

Following any Pareto ascent direction, a solution belonging to the Pareto front is reached. Consider the ascent simplex

$$S(\boldsymbol{\lambda}, \boldsymbol{\theta}) = \sum_{i=1}^{q} \lambda_i \nabla_{\boldsymbol{\theta}} J_i(\boldsymbol{\theta}) \quad \text{s.t.} \quad \sum_{i=1}^{q} \lambda_i = 1,\ \forall i,\ \lambda_i \geq 0.$$

Following the extreme directions (individual steepest ascent directions) one converges to the solution that maximizes one objective, neglecting the others. Any other direction in the ascent simplex will simultaneously increase at least two objectives, ignoring the others. A uniform sampling of the ascent simplex results in evenly distributed directions pointing to the Pareto front with the goal of generating Pareto-optimal solutions as evenly distributed as possible.

Let $p \geq q$ be the granularity of the sampling. The algorithm starts with computing the individual gradients at a single point $\boldsymbol{\theta}^{(0)}$ and identifies the set $\{\mathbf{l}_i\}_{i=1}^{p}$ that realizes the uniform partition of the ascent simplex. Note that every direction $\mathbf{l}_i$ intrinsically defines a preference over the objectives through $\boldsymbol{\lambda}_i$. As a consequence, every candidate solution $\boldsymbol{\theta}_i^{(t)}$, generated according to

$$\boldsymbol{\theta}_i^{(t+1)} = \boldsymbol{\theta}_i^{(t)} + \alpha \mathbf{l}_i^{(t)}, \qquad \mathbf{l}_i^{(t)} = S\left(\boldsymbol{\lambda}_i, \boldsymbol{\theta}_i^{(t)}\right),$$

will be associated to the original preference vector $\boldsymbol{\lambda}_i$.

Note that p points are created at the first iteration and are successively updated according to the associated preference vectors until they reach the Pareto front, as shown in Figure 3. The pseudo code is reported in Algorithm 1.

B. Pareto-Following Algorithm

Pareto-Following Algorithm extends the concept of directed optimization on Pareto front, i.e., the ability of the search algorithm to reside on a neighborhood of the front throughout the optimization process, to the MORL scenario.

Moving a solution along the Pareto front improves some objectives and degrades other ones, according to the followed path. The main problem of directed algorithms is the choice of the search path. In the case of a 2-objective problem, the Pareto front is a line in the objective space, thus only two search directions exist. When $q \geq 3$, a solution on the Pareto front can be moved along an infinite number of directions.

The idea of the Pareto-Following Algorithm is to build a uniform approximation of the Pareto front by optimizing one objective at a time. In this way the choice of the search path is made unique for every parameterization $\boldsymbol{\theta}$.

The Pareto-Following Algorithm starts searching for an extreme point of the Pareto front, by optimizing the first objective $J_1(\boldsymbol{\theta})$. When such a Pareto–optimal solution is reached, it starts optimizing all the other objectives. Let $\boldsymbol{\theta}^{[i]}$ be a solution on the Pareto front obtained by considering only the i–th objective in the last step

$$\boldsymbol{\theta}^{[i]} = \Gamma\left(\boldsymbol{\theta} + \alpha \nabla_{\boldsymbol{\theta}} J_i(\boldsymbol{\theta})\right),$$

where $\boldsymbol{\theta}$ is a solution on the Pareto front and $\Gamma(\mathbf{x})$ is a function that given a candidate solution $\mathbf{x}$ returns a solution on the Pareto front (e.g., Γ can be obtained by repeatedly following the direction obtained from solving the QP problem in [8]). This correction is necessary because, even starting from a solution on the Pareto front, following the steepest ascent direction of some objective may produce a dominated solution. Pareto-Following Algorithm evaluates at most $(q-i+1)$ ascent directions associated to the i–th and to the other objectives (see Figure 4 for a 3–objective example). This means that for any solution $\boldsymbol{\theta}^{[i]}$, the Pareto-Following Algorithm generates $(q-i+1)$ points $\left\{\overline{\boldsymbol{\theta}}^{[k]}\right\}$ such that

$$\overline{\boldsymbol{\theta}}^{[k]} = \Gamma\left(\boldsymbol{\theta}^{[i]} + \alpha \nabla_{\boldsymbol{\theta}} J_k\left(\boldsymbol{\theta}^{[i]}\right)\right) \qquad \forall k = i, \ldots, q. \quad (1)$$

Then it recursively applies the same procedure to any solution $\overline{\boldsymbol{\theta}}^{[k]}$, neglecting solution $\boldsymbol{\theta}^{[i]}$ if it is optimal w.r.t. the i–th objective. The pseudo code of the recursive Pareto-Following Algorithm is reported in Algorithm 2.

V. Gradient on Policy Manifold for Continuous Pareto Frontier Approximation

Pareto–Following and Radial Algorithm overcome the curse of dimensionality of many state-of-the-art MORL algorithms and are able to identify concave frontiers, but return discrete frontiers and lack of guarantees of uniform covering. In this section we describe a gradient–based approach – that we name Policy Manifold Gradient Algorithm (PMGA)– designed to overcome these issues.

A. Parametric Pareto Frontier

It has been shown [9] that local Pareto–optimal solutions locally form a $(q-1)$–dimensional manifold, assuming $d > q$. For instance, in two–objective problems, the Pareto–optimal solutions can be described by curves both in decision and objective spaces. The idea behind this approach is to parameterize the local Pareto–optimal solution curve in the decision space, in order to produce a continuous representation of the Pareto front. The result is a continuous, functional representation of decision and objective spaces. Note that the mapping between the decision (i.e., policy) and objective space is represented by the expected discounted return $\mathbf{J}(\boldsymbol{\theta})$.

The critical point is the definition of the parameterization of the decision space. Let $\mathcal{T}$ be open in $\mathbb{R}^b$ with $b \leq q$ named *generative* space. The high–dimensional analogous of a parameterized curve is a smooth map $\phi_{\boldsymbol{\rho}} : \mathcal{T} \to \Theta$ of class $C^l(l \geq 1)$, where $\mathbf{t} \in \mathcal{T}$ and $\boldsymbol{\rho} \in P \subseteq \mathbb{R}^k$ are the free variable and the parameters, respectively. The set $\Theta_{\boldsymbol{\rho}}(\mathcal{T}) = \psi_{\rho}(\mathcal{T})$, together with the map ψ_{ρ} constitute a parameterized manifold [10] of dimension b, named (parametric) policy manifold. The image of this manifold through $\mathbf{J}(\boldsymbol{\theta})$ represents our approximation of the Pareto front $\mathcal{F}_{\boldsymbol{\rho}}(\mathcal{T}) = \mathcal{F}(\Theta_{\boldsymbol{\rho}}(\mathcal{T})) = \mathbf{J}(\Theta_{\boldsymbol{\rho}}(\mathcal{T}))$ (see Figure 5).

Algorithm 1 Radial Algorithm (RA)

Input: $\boldsymbol{\theta}^{(0)}$
$\{\boldsymbol{\lambda}_i\}_{i=1}^{p} \leftarrow$ uniform sampling of $\mathbb{R}^d$
$\mathbf{d}_i^{(0)} \leftarrow S\left(\boldsymbol{\lambda}_i, \boldsymbol{\theta}^{(0)}\right)$
for $i = 1, \ldots, p$ **do**
  $t = 1$
  while $\boldsymbol{\theta}_i^{(t-1)}$ not Pareto-optimal **do**
    $\boldsymbol{\theta}_i^{(t)} \leftarrow \boldsymbol{\theta}_i^{(t-1)} + \alpha \mathbf{d}_i^{(t-1)}$
    $\mathbf{d}_i^{(t)} \leftarrow S\left(\boldsymbol{\lambda}_i, \boldsymbol{\theta}_i^{(t)}\right)$
    $t \leftarrow t + 1$
  end while
end for

Algorithm 2 Pareto-Following Algorithm (PFA)

Input: the candidate solution $\boldsymbol{\theta}$, the index of the last gradient followed i, the points of the Pareto front $\mathcal{F}$

$c \leftarrow i$
if $\boldsymbol{\theta}$ is optimal w.r.t. the i-th objective **then**
  $c \leftarrow i + 1$
end if
for $k = c, \ldots, q$ **do**
  $\overline{\boldsymbol{\theta}} \leftarrow \Gamma\left(\boldsymbol{\theta} + \alpha \nabla_{\boldsymbol{\theta}} J_k\left(\boldsymbol{\theta}\right)\right)$
  $\mathcal{F} \leftarrow$ Pareto-Following Algorithm $\left(\overline{\boldsymbol{\theta}}, k, \mathcal{F}\right)$
end for
return $\mathcal{F} \cup \{\boldsymbol{\theta}\}$

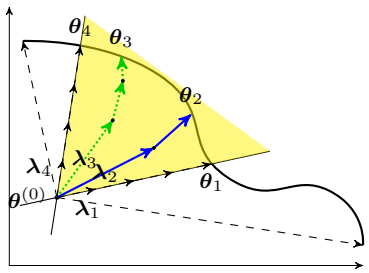


Fig. 3. Behavior of RA in a 2–objectives problem. Four preferences $\boldsymbol{\lambda}_i$ are selected from the ascent simplex in $\boldsymbol{\theta}^{(0)}$. When the initial point $\boldsymbol{\theta}^{(0)}$ is close to the frontier, only a subset of the Pareto frontier can be reached.

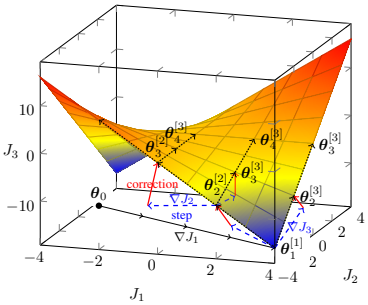


Fig. 4. Pictorial behavior of PFA in a 3–objectives problem. Blue dashed lines denote the step along the single gradient, while red lines denote the correction phase. Dotted lines denote the ideal steps on the frontier.

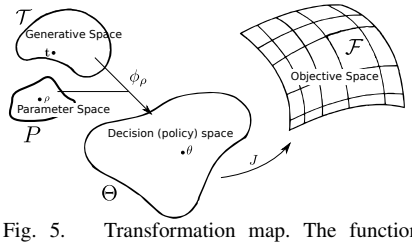


Fig. 5. Transformation map. The function $\phi_{\boldsymbol{\rho}}$, together with generative and parameter spaces, defines the parametrized policy manifold $\Theta_{\boldsymbol{\rho}}(\mathcal{T})$, that maps, through the expected return J, to the approximate Pareto manifold $\mathcal{F}_{\boldsymbol{\rho}}(\mathcal{T})$.

The goal is to find the best approximation, i.e., the parameters $\boldsymbol{\rho}$ that produce a front that is as similar as possible to the Pareto one: $\boldsymbol{\rho}^* = \arg\min_{\boldsymbol{\rho} \in P} \mathcal{I}^*\left(\mathcal{F}_{\boldsymbol{\rho}}\left(\mathcal{T}\right)\right)$, where $\mathcal{I}^* : \mathbb{R}^q \rightarrow \mathbb{R}$ is some loss function that measures the discrepancy between the Pareto–optimal front and $\mathcal{F}_{\boldsymbol{\rho}}\left(\mathcal{T}\right)$.

However, since the Pareto front is not known, a different indicator function is needed. We denote $\mathcal{I} : \mathcal{F}_{\boldsymbol{\rho}}\left(\mathcal{T}\right) \rightarrow \mathbb{R}$ a continuous indicator function that for each point of $\mathcal{F}_{\boldsymbol{\rho}}\left(\mathcal{T}\right)$ measures its Pareto–optimality. The definition of such metric is an open problem in literature. We discuss about it in Section V-B.

In general, MOO algorithms compute the value of the front as sum of the value of the points composing the discrete approximation. In our scenario, where a continuous front approximation is available, it maps to an integration on the Pareto manifold [5, Lemma 3.1]

$$
\begin{aligned}
J(\boldsymbol{\rho}) &= \int_{\mathcal{F}(\mathcal{T})} \mathcal{I}_V \mathrm{d}V \\
&= \int_{\mathcal{T}} \left(\mathcal{I} \circ \left(\mathbf{J} \circ \phi_{\boldsymbol{\rho}}\right)\right) Vol\left(D_{\boldsymbol{\theta}}\mathbf{J}(\boldsymbol{\theta}) D_{\mathbf{t}}\phi_{\boldsymbol{\rho}}(\mathbf{t})\right) \mathrm{d}\mathbf{t},
\end{aligned}
$$

where $\mathrm{d}V$ is a symbol used to denote the integral w.r.t. the volume of the manifold [10] and $Vol\left(X\right) = \left[det\left(X^{\mathrm{T}} \cdot X\right)\right]^{\frac{1}{2}}$. A standard way to maximize the previous equation is to perform gradient ascent, updating the parameters along the gradient direction: $\boldsymbol{\rho}_{t+1} = \boldsymbol{\rho}_t + \alpha_t \nabla_{\boldsymbol{\rho}} J\left(\boldsymbol{\rho}\right)$.

The associated gradient w.r.t. the policy manifold parameters $\boldsymbol{\rho}$ is given component–wise by

$$
\begin{aligned}
\frac{\partial J\left(\boldsymbol{\rho}\right)}{\partial \boldsymbol{\rho}_i} = &\int_{\mathcal{T}} \frac{\partial}{\partial \boldsymbol{\rho}_i} \left(\mathcal{I} \circ \left(\mathbf{J} \circ \phi_{\boldsymbol{\rho}}\right)\right) Vol\left(\mathbf{T}\right) \mathrm{d}\mathbf{t} \\
&+ \int_{\mathcal{T}} \left(\mathcal{I} \circ \left(\mathbf{J} \circ \phi_{\boldsymbol{\rho}}\right)\right) Vol\left(\mathbf{T}\right) \left(\mathrm{vec}\ \left(\mathbf{T}^{\mathrm{T}}\mathbf{T}\right)^{-\mathrm{T}}\right)^{\mathrm{T}} \cdot \\
&\cdot N_b \left(I_b \otimes \mathbf{T}^{\mathrm{T}}\right) D_{\boldsymbol{\rho}_i}\mathbf{T} \mathrm{d}\mathbf{t}
\end{aligned} \tag{2}
$$

where $\mathbf{T} = D_{\boldsymbol{\theta}}\mathbf{J}(\boldsymbol{\theta}) D_{\mathbf{t}}\phi_{\boldsymbol{\rho}}(\mathbf{t})$, $\otimes$ is the Kronecker product, $N_b = \frac{1}{2}\left(I_{b^2} + K_{bb}\right)$ is a symmetric $(b^2 \times b^2)$ idempotent matrix with rank $\frac{1}{2}b(b+1)$ and K_{bb} is a permutation matrix [11]. Note that

$$
\begin{aligned}
D_{\boldsymbol{\rho}_i}\mathbf{T} = &\left(D_{\mathbf{t}}\phi_{\boldsymbol{\rho}}(\mathbf{t})^{\mathrm{T}} \otimes I_q\right) D_{\boldsymbol{\theta}}\left(D\,\mathbf{J}(\boldsymbol{\theta})\right) D_{\boldsymbol{\rho}_i}\,\phi_{\boldsymbol{\rho}}(\mathbf{t}) + \\
&+ \left(I_b \otimes D_{\boldsymbol{\theta}}\mathbf{J}(\boldsymbol{\theta})\right) D_{\boldsymbol{\rho}_i}\left(D_{\mathbf{t}}\phi_{\boldsymbol{\rho}}(\mathbf{t})\right)
\end{aligned}
$$

where $D_{\boldsymbol{\theta}}\left(D\,\mathbf{J}(\boldsymbol{\theta})\right)$ is a transformation of the Hessian matrix $H\,\mathbf{J}(\boldsymbol{\theta})$ of the performance w.r.t. policy parameters, that is, they contain the same elements, but in different order $H_{\boldsymbol{\theta}}^{m,n} J_i = D_{\boldsymbol{\theta}}^{p,n}\left(D\,\mathbf{J}(\boldsymbol{\theta})\right)$, where $p = i + q(m-1)$ and q is the number of rows of the Jacobian matrix. For the definition of the Hessian matrix we refer to [5, Lemma 3.2].

The formulation of the gradient $\nabla_{\boldsymbol{\rho}}\mathbf{J}(\boldsymbol{\rho})$ is composed by terms related to the parameterization of the manifold in the policy space and terms related to the MDP. Since the map $\phi_{\boldsymbol{\rho}}$ is free to be designed, the associated terms (e.g., $D_{\mathbf{t}}\phi_{\boldsymbol{\rho}}(\mathbf{t})$) can be computed exactly. On the other hand,

the terms related to the MDP ($\mathbf{J}^{\boldsymbol{\theta}}$, $\nabla_{\boldsymbol{\theta}}\mathbf{J}(\boldsymbol{\theta})$ and $H\,\mathbf{J}(\boldsymbol{\theta})$) need to be estimated. While the trajectory–based estimate of the expected discounted reward and the associated gradient is an old topic in RL literature and several results have been proposed [12], the Hessian estimate has been recently addressed in [5]. Finally, the integral in Equation 2 can be estimated using standard Monte–Carlo techniques [13].

B. Metrics for Multi–objective Optimization

Although literature has proposed and analyzed several metrics to measure how good are frontier approximations [14, 15], no consensus exists on appropriate performance measure to be used for empirical evaluation. Recently, continuous metrics have been proposed and empirically compared in [5]. Inspired by such work, we have tried to mix different indicator concepts in order to obtain a metric with the desired properties: we want a frontier that includes solutions that are *accurate*, *evenly distributed* and *covering* a range similar to the Pareto one [16].

While the uniform distribution of the frontier is guaranteed by the continuity of the PMGA, accuracy can be guaranteed by minimizing the distance from the utopia point $\mathbf{p}_U$ (i.e., the point that optimizes all the objective functions). Generally, a simple indicator function can be defined up to a reference point $\mathbf{p}$ as follows:

$$\mathcal{I}_{ref}(\mathbf{J},\mathbf{p}) = \|\mathbf{J}-\mathbf{p}\|_2^2\,. \tag{3}$$

Although, utopia–based metric is able to guarantee accuracy of the solution (for every dominated point there exists a Pareto–optimal point that minimizes the distance from the utopia), it forces the solution to collapse into a single point.

In order to get the maximal extent of the frontier it is possible to maximize Equation (3) by considering the antiutopia ($\mathbf{p}_{AU}$) as reference point. However, the maximization of the latter indicator may lead to very large fronts, but with very bad dominated solutions. In fact, searching for solutions far from the antiutopia does not imply to find solutions close to the Pareto front.

Since such metrics, individually, do not satisfy all the desiderata, we have decided to mix utopia– and antiutopia–based metrics. We want solutions that are simultaneously far from the antiutopia and close to the utopia. So, we consider the following metric (to be maximized):

$$\overline{\mathcal{I}}(\mathbf{J}) = \beta_1 \frac{\mathcal{I}_{ref}(\mathbf{J},\mathbf{p}_{AU})}{\mathcal{I}_{ref}(\mathbf{J},\mathbf{p}_U)} - \beta_2,$$

where β_1 and β_2 are free parameters. Notice that, if we knew that the Pareto front is convex (i.e., the utopia is closer to any point on the Pareto front than the anitutopia is), then it would be reasonable to set β_1 and β_2 both to 1.

As for the hypervolume indicator [16], the definition of the reference point is critical and has a relevant impact on the final performances of the algorithm. However, compared to other continuous indicators, it enjoys all the desiderata and it is easy to differentiate. In the next section, we will show that the proposed metric is effective to drive PMGA close to the Pareto front both in exact and approximate scenarios.

VI. Experiments

In this section, results related to the numerical simulations of the proposed algorithms, in continuous domains, are presented. In particular, the performance of the proposed algorithms is compared against some existing algorithms [17, 18] using an extension of a previously defined metric [18] that measures the distance of an approximation of the Pareto front from a reference one. Since PMGA returns continuous fronts, they have been discretized to evaluate such loss function.

In PFA and RA experiments the learning rate is fixed by hand-tuning, while for PMGA an in–learning tuning is performed to avoid oscillations. As terminal condition, PFA and RA are stopped when the gradient norm gets below some threshold. PMGA instead performs a fixed number of iterations. Finally, all the results in the approximate setting are averaged over 10 runs.

A. Domains

To illustrate the performance of the algorithms, we consider the following MDPs.

1) LQG: The first case of study is a discrete–time Linear-Quadratic Gaussian regulator (LQG) with multidimensional and continuous state and action spaces [7]. For a complete description of the LQG problem and for the settings, we refer to [4]. This scenario is interesting since all the terms can be computed exactly and the precision of the proposed algorithms can be analyzed using both the exact and the estimated functions. Further, since the Pareto front is convex, a weighted sum (WS) method can be exploited in order to obtain a reference front used to compare our approaches.

2) Water Reservoir: A water reservoir can be modeled as a MOMDP with a continuous state variable s representing the water volume stored in the reservoir, a continuous action a that controls the water release, a state-transition model that depends also on the stochastic reservoir inflow ϵ, and a set of conflicting objectives. For a complete description of the problem, the reader can refer to [4].

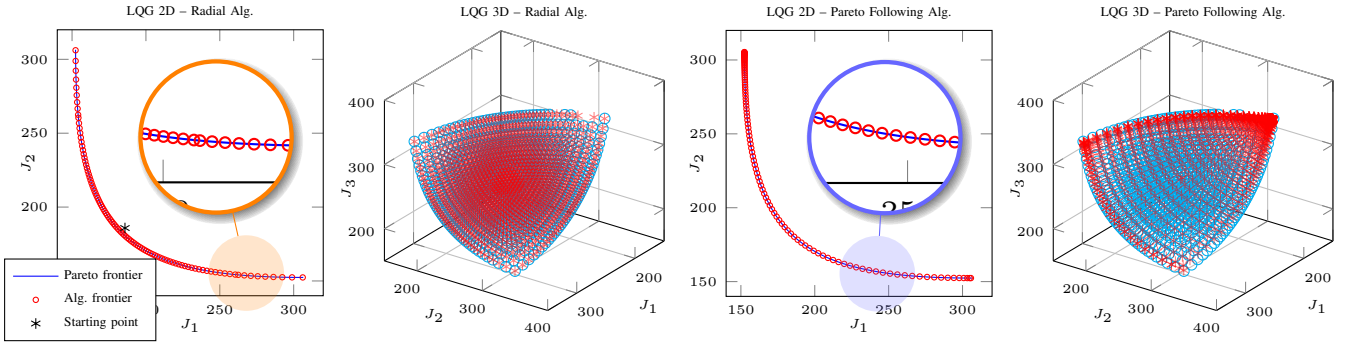
In this work we consider three objectives: flooding along the lake shores, irrigation supply and hydropower supply. Like in the original work, the discount factor is set to 1 for all the objectives. However, different settings are used for learning and evaluation. In the learning phase $1,000$ episodes by 10 steps with initial state $s_0 \sim Unif(0,160)$ are used for PFA and RA, and 100 episodes by 100 steps for PMGA. The evaluation phase is the same for all the algorithms: $10,000$ episodes by 100 steps with initial state drawn from a finite set. Policies are represented by parametric Gaussian distributions over the continuous action space $\pi(a|s,\boldsymbol{\theta}) = \mathcal{N}\left(\phi(s)^{\mathrm{T}}\kappa,\sigma\right)$, where $\phi : \mathcal{S} \to \mathbb{R}^{d-1}$ are the basis functions. In PMGA setting $\sigma = 0.1$, while in order to prevent the variance from becoming negative PFA and RA exploit a logistic variance as presented in [4].

Since the optimal policies for the objectives are not linear in the state variable, a radial basis approximation is used: $\phi(s) = \left[e^{-\|s\,-\,c_i\|_2/w_i}\right]_{i=1}^{d-1}$, where the centers c_i are placed at 0, 50, 120 and 160, and the widths are 50, 20, 40 and 50.

TABLE I

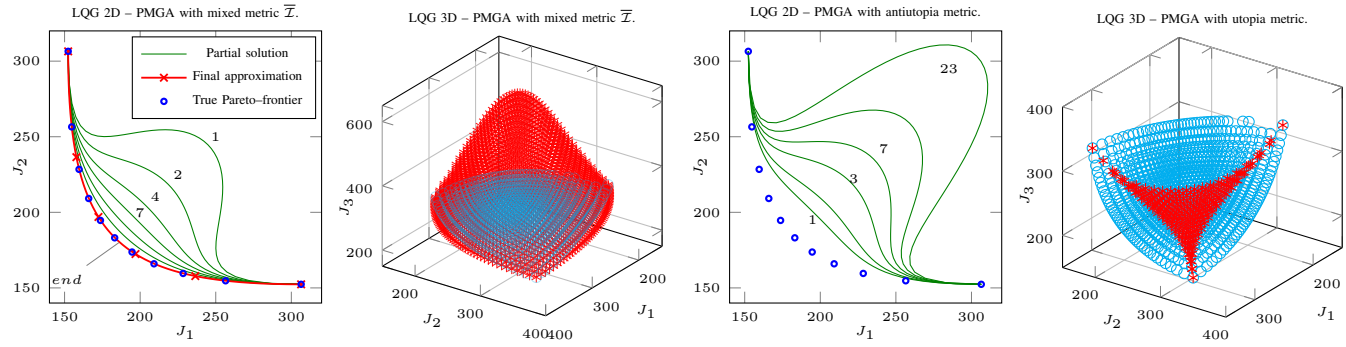
LQG: ALGORITHM COMPARISON IN EXACT AND APPROXIMATE SETTINGS

Algorithm	LQG 2-obj. Loss	LQG 2-obj. #Iterations	LQG 2-obj. #Solutions	LQG 3-obj. Loss	LQG 3-obj. #Iterations	LQG 3-obj. #Solutions
RA ex.	1.9888e-4	2,081	101	1.6921e-4	3,840	989
PFA ex.	5.6975e-5	288	161	4.0863e-4	2,317	943
PMGA ex.	4.2515e-4	46	∞	0.0045	141	∞
RA apx.	1.9868e-4 $\pm$ 3.7481e-5	1,654 $\pm$ 124.8047	94.6 $\pm$ 3.0067	XXX	XXX	XXX
PFA apx.	1.0202e-4 $\pm$ 2.6125e-5	637 $\pm$ 55.7512	406.3 $\pm$ 39.3956	2.64474-4 $\pm$ 3.1516e-5	6,664 $\pm$ 140.2	2,302 $\pm$ 180.3
PMGA apx.	1.3178e-4 $\pm$ 7.9188e-5	141	∞	5.2849e-4 $\pm$ 3.3856e-4	151	∞



(a) (b) (c) (d)

Fig. 6. Approximate Pareto front obtained by RA ((a) and (b)) and PFA ((c) and (d)) in both 2D and 3D LQG problem. Initial parametrization are $\boldsymbol{\theta}_0 = [-0.5, 0, 0, -0.5]^\intercal$ and $\boldsymbol{\theta}_0 = [-0.17, 0, 0, -0.17]^\intercal$, for 2D PFA and 2D RA scenarios, respectively. For the 3D LQG both algorithms starts from $\boldsymbol{\theta}_0 = [-0.5, 0, 0, 0, -0.5, 0, 0, -0.5]^\intercal$.



(a) (b) (c) (d)

Fig. 7. Continuous approximate Pareto front for the LQG problem using PMGA . The parameterization $\phi_{\boldsymbol{\rho}}$ used is forced to pass through the extreme points of the front. In the 2D case only few iterations have been reported, each one with the associated iteration number, where *end* denotes the front obtained when the terminal condition is reached. In Figures (a) and (b) indicator function $\overline{\mathcal{I}}$ is used, starting from $\boldsymbol{\rho}_0 = [2, 2]^\intercal$ and $\boldsymbol{\rho}_0 = \mathbf{0}$, respectively. Figures (c) and (d) report the results obtained using an antiutopia–based indicator ($\boldsymbol{\rho}_0 = [1, 1]^\intercal$) and an utopia–based indicator ($\boldsymbol{\rho}_0 = \mathbf{0}$).

B. Numerical results

For each of the previous domains, simulation results are reported here, comparing them with the results obtained using other algorithms. Since it is not possible to associate weights with policies found by the gradient algorithms, for each weight used in the reference solution, the policy that minimizes the loss function is selected.

1) LQG: Concerning Pareto Following and Radial algorithms, gradients $\nabla_{\boldsymbol{\theta}} J_i(\boldsymbol{\theta})$ are not normalized, neither are objectives since the initial state s_0 ensures they have the same order of magnitude. Radial algorithm follows 101 and 989 directions (2-objective case and the 3-objective case, respectively) uniformly sampled in the gradient simplex. About the settings, since LQG problem is defined only for control actions in the range $[-1, 0]$, our primary concern was related to the boundedness of the control actions. In order to guarantee this constraint, the following parameterization of the manifold in the policy space was used for the 2D LQG: $\phi^1_{\boldsymbol{\rho}}(\mathbf{t}) = -(1 + \exp(\rho_1 + \rho_2 t))^{-1}$ and $\phi^2_{\boldsymbol{\rho}}(\mathbf{t}) = -(1 + \exp(\rho_3 + \rho_4 t))^{-1}$ with $t \in [0, 1]$. Similarly, a complete second degree polynomial with $|\boldsymbol{\rho}| = 9$ and $\mathbf{t} \in simplex([0, 1]^2)$ is used for the 3D LQG. Finally, both parameterizations are forced to pass through the extreme points of the front. For more details we refer to [5].

Figures 6(a) and 6(c) show the approximated front obtained with the exact RA and PFA. It can be noticed that the approximations are not completely uniform as the first has more solutions concentrated near the knee of the front, while the second is more dense at the extremity. This behavior can be also observed in the 3-objective scenario. In the case of PFA, this is due to the learning rate, that highly influences solutions distribution. It is interesting that, even starting near to the center of the front (Figure 6(a)), Radial Algorithm is still able to reach its extremity. Compared to PFA, RA also

proved to be faster since it can use a larger learning rate and still obtain a uniform front, while PFA needs smaller learning rates to prevent sparse fronts. Better approximations can be obtained with adaptive learning rates [12], but this does not ensure to get a uniform approximation of the front.

On the contrary, with PMGA (Figure 7) we are able to obtain continuous and accurate fronts using the mixed metric proposed in Section V-B ($\beta_1 = 1$ and $\beta_2 = 1$; $\mathbf{p}_U = [150, 150]$ and $\mathbf{p}_{AU} = [310, 310]$ for the 2D LQG, $\mathbf{p}_U = [195, 195, 195]$ and $\mathbf{p}_{AU} = [360, 360, 360]$ for the 3D LQG). In the 2-objective case PMGA returns a good approximation of the Pareto front, while in the 3-objective case dominated solutions are generated (Figure 7(b)). Figures 7(c) and 7(d) show the fronts obtained using antituopia–based and utopia–based indicators, respectively. In the first case solutions tend to diverge from the Pareto front, while in the second they concentrate on the knee of the front.

2) Water Reservoir: To evaluate the effectiveness of the gradient algorithms we have analyzed their performances against the solution found by Stochastic Dynamic Programming (SDP, chosen as reference front, since the optimal Pareto front is not available) using the weighted sum method [18], Multi-Objective FQI [18] (using $20,000$ samples with a dataset of $200,000$ tuples) and the standard FQI [17]. Both MOFQI and FQI have been trained using $10,000$ samples with a dataset of $50,000$ tuples for the 2–objectives problem and $20,000$ samples with a dataset of $500,000$ tuples for the 3–objective problem. FQI scalarizes the objectives using the same weights as SDP.

Concerning Pareto Following and Radial algorithms, since the objectives $\mathbf{J}$ have different magnitude, they have been normalized. Radial algorithm follows 21 and 66 directions (2-objective case and the 3-objective case, respectively) uniformly sampled in the simplex. Furthermore, due to the fact that the transition function limits the action in the range of admissible values ($a_t \in [\underline{a}_t, \bar{a}_t]$), there are infinite policies with equal performance that allow the agent to release more than the reservoir level or less than zero. A penalty term in the reward $p = -\max(a_t - \bar{a}_t, \underline{a}_t - a_t)$ is thus introduced, in order to allow Pareto Following and Radial algorithms to learn. In order to be able to compare the results with the original work, the penalty is considered only in the learning phase and not in the evaluation of the policy. On the contrary, the penalty was not necessary for PMGA .

PMGA exploits the mixed indicator function ($\beta_1 = 1$ and $\beta_2 = 1$; $\mathbf{p}_U = [-0.5, -9, -0.001]$ and $\mathbf{p}_{AU} = [-2.5, -11, -0.7]$) and a complete first–degree polynomial with one ρ_i for each of the five $\phi^i_{\boldsymbol{\rho}}(\mathbf{t})$ ($\boldsymbol{\rho}_0 = -\mathbf{20}$) and $t \in [0, 1]$ for the 2–objective case. Similarly, for the 3–objective case a complete second–degree polynomial with three ρ_i for each $\phi^i_{\boldsymbol{\rho}}(\mathbf{t})$ and $\mathbf{t} \in simplex([0, 1]^2)$ is used ($\boldsymbol{\rho}_0 = \mathbf{0}$). In both cases the parameterization is forced to pass near the extreme points of the front.

Figure 8 shows a graphical representation of the Pareto points obtained by the algorithms when only the first two objectives are considered, and Table II reports the loss

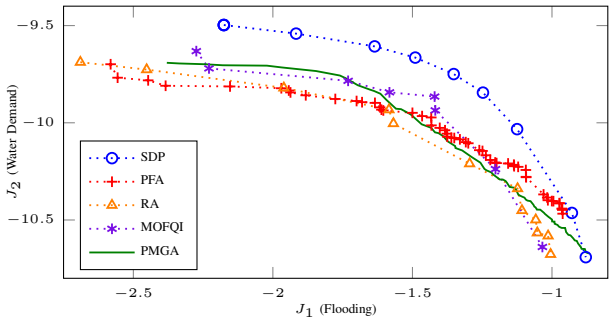


Fig. 8. Approximate Pareto fronts for the 2–objective water reservoir.

achieved by the algorithms w.r.t. the SDP approximation. Gradient algorithms achieved a better loss than MOFQI and FQI. It is interesting to notice that PMGA attains the best performance both in the 2–objective and 3–objective cases.

C. Empirical sample complexity analysis

Here we provide an empirical analysis of the sample complexity of PMGA in the 2–dimensional LQR domain by varying the number of policies used to estimate the integral, the number of episodes for each policy and the number of steps for each episode. As indicators, we take the number of iterations and the number of evaluations of the generative model required to reach a loss from the WS approximation less than $5 \cdot 10^{-4}$. This terminal condition must be reached in $1,000$ iterations otherwise the algorithm is forced to terminate. The symbol $\perp$ is used to represent the latter case. Table III reports the results averaged over 10 runs. Note that each row shows results obtained by varying a single parameter (the one specified in the first column), while the other two are fixed to 30.

From Table III results that the most relevant parameter is the number of episodes used to estimate MDP terms: $\mathbf{J}(\boldsymbol{\theta})$, $D_{\boldsymbol{\theta}}\mathbf{J}(\boldsymbol{\theta})$ and $H\mathbf{J}(\boldsymbol{\theta})$. This parameter controls the variance in the estimate, i.e., the accuracy of gradient estimate $\nabla_{\boldsymbol{\theta}}\mathbf{J}(\boldsymbol{\rho})$. By increasing the number of episodes, the estimation process is less prone to generate misleading directions, as happens, for instance, in the 1–episode case where parameters move into wrong direction. On the contrary, the numbers of steps and points used to estimate the integral (denoted in table by **#t**) seem to have no significant impact on the final performance of the algorithm, but influence the number of model evaluations needed to reach the prescribed accuracy.

The best behavior, from a sample–based perspective, has been obtained by exploiting only one point for the integral estimate. Although, it can be surprising, a simple explanation exists. By forcing the parameterization to pass through the single–objective optima, the information of correct estimate of gradient direction of a single point $\mathbf{t}$ is sufficient to move the entire front toward the Pareto one, i.e., to move the parameters to the optimal ones. Under the settings of Table I, if we force the PFA and RA to use the same sample budget ($3.52 \cdot 10^5$ samples), they would have performed only 70 iterations, that correspond to 4 and 44 solutions for RA and PFA, respectively. Clearly, an initial optimization phase is required to obtain the single–objective optimal parameterizations that have been exploited by PGMA.

TABLE II

WATER RESERVOIR: ALGORITHM COMPARISON WITH 2 AND 3 OBJECTIVES

Algorithm	Water 2-obj. Loss	#Iterations	#Solutions	Water 3-obj. Loss	#Iterations	#Solutions
RA	0.1315 ± 0.0047	974.5 ± 42.7	12.1 ± 0.6	0.0218 ± 0.0010	$2,441 \pm 125.9$	64.3 ± 0.4
PFA	0.1013 ± 0.0069	347.5 ± 46.2	55.1 ± 2.8	0.0224 ± 0.0018	$2,096 \pm 330.2$	1126.7 ± 101.2
PMGA	0.0826 ± 0.0010	401	∞	0.0216 ± 0.0005	81	∞
MOFQI [18]	0.1870 ± 0.0090	-	-	0.0540 ± 0.0061	-	-
FQI [17]	0.1910 ± 0.0100	-	-	0.0292 ± 0.0010	-	-

TABLE III

LQG 2–OBJ.: SAMPLE COMPLEXITY

		PMGA Parameters 1	5	10	20	30	40	50
#t	#Iterations	390.6 ± 50.0	307.8 ± 37.9	221.3 ± 32.9	158.8 ± 9.4	157.0 ± 20.4	149.2 ± 21.9	111.5 ± 9.7
	#Samples (10^5)	$\mathbf{3.5 \pm 0.5}$	13.9 ± 1.7	19.9 ± 3.0	28.6 ± 1.7	42.4 ± 5.5	53.7 ± 7.9	50.2 ± 4.4
#Episodes	#Iterations	$\perp$	$\perp$	$\perp$	$\perp$	127.6 ± 11.0	71.0 ± 4.5	68.4 ± 5.0
	#Samples (10^5)	$\perp$	$\perp$	$\perp$	$\perp$	34.5 ± 3.0	$\mathbf{25.6 \pm 1.6}$	30.8 ± 2.2
#Steps	#Iterations	$\perp$	155.2 ± 17.0	185.9 ± 17.4	135.3 ± 13.0	139.1 ± 6.1	157.2 ± 11.3	162.5 ± 18.9
	#Samples (10^5)	$\perp$	$\mathbf{7.0 \pm 0.8}$	16.7 ± 1.6	24.4 ± 2.3	37.6 ± 1.6	56.6 ± 4.1	73.1 ± 8.5

VII. CONCLUSIONS

Multi–objective problems are relevant in many real–world applications. However, few attention has been posed on MOMDPs in the field of RL. This paper has investigated three recent multi–objectives gradient–based methods; Two approaches (RA and PFA) produce a discrete approximation of the Pareto front, while the third (PMGA) learns a continuous approximation. An extensive empirical analysis has been carried out to investigate the performance of the individual algorithms. In particular a sample complexity analysis has been performed for the PMGA approach.

Future research will further address the study of metrics that can produce good results in general settings. Another interesting research direction consists in using importance sampling techniques for reducing the sample complexity in the gradient estimate. Since the frontier is composed of a continuum of policies, it is likely that a trajectory generated by some policy can be partially used also for the estimation of quantities related to similar policies, thus decreasing the number of samples needed for the estimate of the integral.

REFERENCES

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

[2] D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley, "A survey of multi-objective sequential decision-making," *JAIR*, vol. 48, pp. 67–113, 2013.

[3] J. Peters and S. Schaal, "Policy gradient methods for robotics," in *IROS*. IEEE, 2006, pp. 2219–2225.

[4] S. Parisi, M. Pirotta, N. Smacchia, L. Bascetta, and M. Restelli, "Policy gradient approaches for multi-objective sequential decision making," in *IJCNN 2014, Beijing, China, July 6-11, 2014*. IEEE, 2014, pp. 1–7.

[5] M. Pirotta, S. Parisi, and M. Restelli, "Multi–objective reinforcement learning with continuous pareto frontier approximation," *arXiv:1406.3497*, 2014.

[6] K. Harada, J. Sakuma, and S. Kobayashi, "Local search for multiobjective function optimization: pareto descent method," in *GECCO*, 2006, pp. 659–666.

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

[8] M. Brown and R. E. Smith, "Directed multi-objective optimization," *Int. J. Comput. Syst. Signal*, vol. 6, no. 1, pp. 3–17, 2005.

[9] K. Harada, J. Sakuma, S. Kobayashi, and I. Ono, "Uniform sampling of local pareto-optimal solution curves by pareto path following and its applications in multi-objective ga," in *Proceedings of GECCO '07*. New York, NY, USA: ACM, 2007, pp. 813–820.

[10] J. Munkres, *Analysis On Manifolds*, ser. Adv. Books Classics Series. Westview Press, 1997.

[11] J. Magnus and H. Neudecker, *Matrix Differential Calculus with Applications in Statistics and Econometrics*, ser. Wiley Ser. Probab. Statist.: Texts and References Section. Wiley, 1999.

[12] M. Pirotta, M. Restelli, and L. Bascetta, "Adaptive step-size for policy gradient methods," in *NIPS 26*. Curran Associates, Inc., 2013, pp. 1394–1402.

[13] C. P. Robert and G. Casella, *Monte Carlo statistical methods*. New York: Springer, 2004, vol. 319.

[14] P. Vamplew, R. Dazeley, A. Berry, R. Issabekov, and E. Dekker, "Empirical evaluation methods for multi-objective reinforcement learning algorithms," *Machine Learning*, vol. 84, no. 1-2, pp. 51–80, 2011.

[15] T. Okabe, Y. Jin, and B. Sendhoff, "A critical survey of performance indices for multi-objective optimisation," in *CEC '03.*, vol. 2, Dec 2003, pp. 878–885 Vol.2.

[16] E. Zitzler, L. Thiele, M. Laumanns, C. Fonseca, and V. da Fonseca, "Performance assessment of multiobjective optimizers: an analysis and review," *Evolutionary Computation, IEEE Transactions on*, vol. 7, no. 2, pp. 117–132, 2003.

[17] D. Ernst, P. Geurts, and L. Wehenkel, "Tree-based batch mode reinforcement learning," *Journal of Machine Learning Research*, vol. 6, pp. 503–556, 2005.

[18] F. Pianosi, A. Castelletti, and M. Restelli, "Tree-based fitted q-iteration for multi-objective markov decision processes in water resource management," *Journal of Hydroinformatics*, vol. 15, no. 2, pp. 258–270, 2013.