Neural Horizons: Exploring Matryoshka Policy Gradients

January 19, 2024

Master project in applied mathematics. Dept. of Mathematics, Statistical Field Theory (CSFT) chair, EPFL university.

Student: Konstantin Medyanikov
Supervisor: François Gaston Ged
Supervisor: Clément Hongler

Abstract

This Master's these focuses on the role of neural networks as function approximators in Reinforcement Learning (RL), specifically examining the Matryoshka policy gradient algorithm. It aims to deepen the understanding of the algorithm's mechanisms and its implications in RL, drawing connections between RL algorithms and kernel methods to analyze the convergence properties of function approximators. Additionally, the study introduces a novel design of the Matryoshka algorithm, enhancing its adaptability and efficacy, particularly when neural networks serve as function approximators. The research provides a comprehensive analysis of both theoretical aspects and practical applications, striving to bridge the gap between theoretical models and real-world applications in RL.

Notations Specification

Greek Letters

η	Learning rate
γ	Discount factor
ψ	Feature map
ϕ	Feature map
θ	Parameters
Θ	Kernel associated to the feature map (coincide with NTK)
Σ	Conjugate Kernel
$\vartheta^{(i)}$	Parameters of i-nth step horizon
C	Constant value
C_i	Cumulative regularized reward for i-nth horizon

Notation for Sets

$\mathcal{S}$	State space
$\mathcal{S}_0$	Initial state space
$\mathcal{A}$	Action space
$\mathcal{P}$	Stationary policy space
$\mathcal{P}_n$	Vector of n stationary policies, $\pi \in \mathcal{P} \times \mathcal{P} \ldots \times \mathcal{P}$
$\mathcal{H}_\theta$	Reproducible Kernel Hilbert Space of Θ
$\mathcal{P}^{\mathcal{H}}$	Policies, which preference functions lies in $\mathcal{H}$

Probability measures and *MDP*

p	Transition probability
p_{rew}	Reward distribution
$m^{(i)}$	State distribution of an i-nth step horizon
ϑ or $m^{(0)}$	Initial distribution
μ or m^{∞}	Stationary distribution
$\bar{\pi}$	Base-line policy
π	Policy

Notation Rules

For the objects: $\theta, \vartheta, \pi, m, \Theta, \Sigma, J, Q, V$, we apply following rules. Let O represents one of the mentioned objects.

O_θ	Parameterized by θ
O_{θ_t}	Parameterized by θ at time t
O_π	Generated by policy π
O_{π_θ}	Generated by policy π which is parameterized by θ
$O_{\pi_{\theta_t}}$, O_{π_t} or O_t	Generated by policy π which is parameterized by θ_t
$O^{(i)}$	Of i-nth step horizon
$\hat{O}$	Estimated

Contents

1 Introduction

Reinforcement Learning (RL) has proven to be one of the most influential paradigms in modern machine learning, with applications spanning from game playing to robotic control. A crucial aspect of RL lies in its ability to learn optimal strategies through interactions with an environment. In this work, we delve deep into a specific category of RL, focusing on the Max-entropy fixed horizon policy, its foundations, and its implications. Through this lens, we explore the Matryoshka Policy Gradient, an innovative approach that promises convergence and global optimality. Our contributions further expand upon these foundations, aiming to provide insights, algorithms, and practical methodologies that can push the boundaries of what RL can achieve.

1.1 Policy gradient in Max-entropy & Fixed-horizon RL

Policy gradient RL

Policy Gradient methods in Reinforcement Learning offer a way to directly optimize the policy without relying on an action-value function. These methods belong to the family of model-free algorithms, meaning they do not require explicit knowledge of the environment's dynamics. Instead, they learn optimal strategies directly from interactions with the environment.

Mathematically, the objective of policy gradient methods is to maximize the expected cumulative reward, by updating the policy parameters, θ, in the direction of the gradient of the expected reward:

$$J(\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t R_t\right]$$
$$\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta),$$

Where π_θ represents the policy parameterized by θ, γ is discount factor and α is the learning rate. For the more formal introduction of the Objective function, we refer reader to the following 1.2 "Set-up" section.

Max-entropy RL

Max-entropy Reinforcement Learning aims to not only maximize the expected return but also the entropy of the policy. By promoting stochastic behaviors, Max-entropy RL encourages better exploration, making it less likely for the agent to get stuck in sub-optimal strategies.

In Max-entropy RL, the reward function is redefined to incorporate the entropy term. For (a, s) being agent's action/state pair, we write

$$\tilde{r}(s, a) = r(s, a) + \tau H(\pi(\cdot|s)) \tag{1}$$

Where H denotes the entropy and τ is a coefficient that controls the trade-off between maximizing reward and entropy. Sometimes, we refer to it as "temperature" factor. The term comes from physics, where τ plays the role of the temperature. As we are going to see, increase in τ would motivate better exploration (making policy be more stochastic). New reward, would bring the new objective function:

$$J(\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t \left(R_k + \tau H(\pi(\cdot|S_k))\right)\right].$$

Fixed-horizon RL

Fixed-horizon Reinforcement Learning involves making decisions by considering only a fixed number of future steps, rather than the entire future. This approach has two main benefits. Firstly, it makes the optimization problem more tractable by bounding the depth of consideration. Secondly, in many real-world scenarios, agents operate under constraints that inherently introduce a horizon. While, on the surface, fixed-horizon might appear restrictive, it naturally arises in various environments or games. Examples include scenarios with a set time limit or games where the objective is to reach a treasure within a certain distance. Additionally, the fixed-horizon approach can be particularly useful in scenarios with non-stationary policies, providing a more adaptable and flexible learning mechanism.

Function approximation and Global convergence

Function approximation in the context of policy gradient methods is the use of parametric models to estimate the value and Q-functions, especially when dealing with large or continuous state spaces where tabular methods are impractical. Common function approximators include neural networks, decision trees, or linear models that map states (and possibly actions) to action values (policy). A common form of function approximation for Q-values is the linear model, which can be expressed mathematically as:

$$\hat{Q}(s,a) = \boldsymbol{\psi}(s,a)^T \boldsymbol{\theta}, \tag{2}$$

where $\boldsymbol{\psi}(s,a)$ is the feature vector representing the state-action pair, and $\boldsymbol{\theta}$ is a weight vector of the linear model. The choice of $\boldsymbol{\psi}(s,a)$ can vary widely:

- In a tabular setting, $\boldsymbol{\psi}(s,a)$ could be a one-hot encoded vector with a length equal to the number of state-action pairs, effectively mimicking a lookup table.
- For simple function approximation, $\boldsymbol{\psi}(s,a)$ might include hand-crafted features or polynomials of the raw state and action variables.
- In deep learning, $\boldsymbol{\psi}(s,a)$ could be a complex, high-dimensional representation learned by a neural network. Often, the equation (2) is encoded inside of the NN, in the form of the last Linear layer.

The softmax function is then used to produce the π_θ policy:

$$\pi_\theta\left(a \mid s\right) = \frac{\exp\left(\hat{Q}(s,a)\right)}{\int_A \exp\left(\hat{Q}(s,a)\right) da} \tag{3}$$

While function approximation enables generalization and scalability, it can introduce several disadvantages regarding convergence. In the tabular case, where each state-action pair is discretely mapped to a value, convergence under certain conditions is mathematically guaranteed. However, when function approximators are used, particularly with policy gradient methods, convergence is no longer assured due to several factors:

- The *bias-variance tradeoff* inherent in the choice of function approximator can lead to suboptimal policies.

- The use of bootstrapping methods, where the agent updates its value estimates based on other estimated values, can propagate errors through the approximator.
- Off-policy learning, where the policy being improved upon is different from the policy generating the data, can exacerbate the instability caused by the function approximator.

These factors contribute to what is known in the reinforcement learning literature as the *deadly triad*. Beside this, our goal is not only to escape this triad, but as well, to establish the convergence to the global optima. Which is hard task, even under tabular conditions. Only in recent years, we could have seen a progress progress in this field. For example work done by Alekh Agarwal et al. [1], that study global convergence for various settings, in particular for policy gradient, with discounted reward and tabular-softmax parametrization (see theorem 10 in [1]). Notable work, published by Sara Klein et al. [7] has shown the global convergence to the optimal policy for the regularized Softmax policy gradient method by reaching the PL inequality. Remarkable the fact, that they as well had to adapt the Fixed Horizon structure. This brings us to the important point, the Fixed Horizon is great theoretical concept or better to say framework, that make it possible to escape the deadly triad, but as well to establish the convergence, by propagating the optimally from shorter to longer range horizons. Unfortunately, the results delivered by Sara Klein et al, have been obtained for finite MDP settings and tabular parametrisation. To push this boundaries even further, we make a call to the work of Francois Ged and Maria Han Veiga, where the same concept has been realized in continuous Space/Actions and Linear function approximation, providing theoretical findings of global convergence (see Theorem 1 in [3]). In their work, they as well extend the idea, to the case of the Neural Network. In particular, their results on the global convergence could be easily adapted for a NN trained under ntk regime[1].

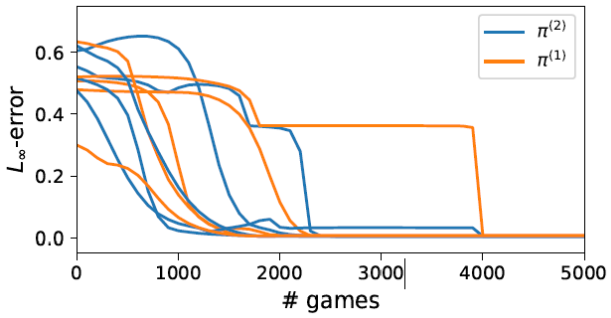


Figure 1: Figure demonstrate the convergence (of 5 independent agents) to the optimal policy for the Matryoshka Policy Gradient algorithm, measured by the L_∞-norm. The results were produced for the first and second step horizon.

As mentioned earlier, tabular approaches are unpractical in many modern scenarios. To this end, we will delve into the function approximation settings. And, in particular our analysis and methodologies are made on top of the Matryoshka Policy Gradient algorithm and theory constructed behind this approach.

[1]The ntk or lazy regime refers to a phenomenon where, during training, the weights of a neural network change very little from their initial values.

Set of Policies

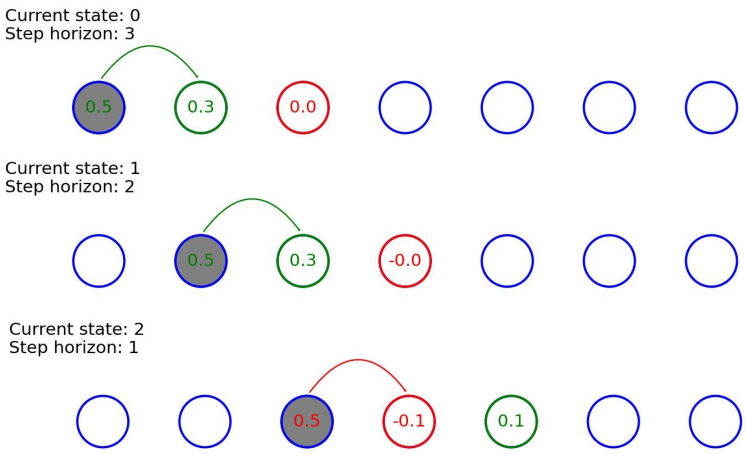
In contrast to the conventional approach of employing stationary policies, this study focuses on non-stationary policies, which vary depending on the horizon step. We conceptualize a set of stationary policies $\{\pi^{(i)}\}_{i=1}^{n}$, where n denotes the final step and the index i represents the "step horizon." In this model, for a state at time $t \in \mathbb{N}$, the agent's behavior is governed by the policy $\pi^{(n-t)}$:

$$A_t \sim \pi^{(n-t)}(\cdot \mid S_t)$$

It is crucial to note the reversed ordering for the step horizon: for the initial game state S_0, the policy $\pi^{(n)}$ is applied, while for the final step S_{n-1}, the policy $\pi^{(1)}$ is utilized.

Introduction to the "Toy" Testing Environment To acquaint the reader with the fixed horizon setting, we introduce our principal testing environment, termed "Toy". Its design is purposefully crafted to highlight the benefits of employing a fixed horizon.

The environment consists of a set of states $\mathcal{S} = \{0, 1, 2, \ldots, m-1\}$, with a randomly selected initial state. At each time step $t \in \mathbb{N}$, the agent may choose between two actions $\mathcal{A} = \{+1, +2\}$. The state transitions are defined as either $S_{t+1} = (S_t + 1) \mod m \in \mathcal{S}$ or $S_{t+1} = (S_t + 2) \mod m \in \mathcal{S}$. Correspondingly, for each pair $(a, s) \in \mathcal{A} \times \mathcal{S}$, an immediate reward $r_{a,s} \in \mathbb{R}$ is assigned. Consider, for instance, the case where $\mathcal{S} = \{0, 1, 2, 3, 4, 5, 6\}$, the maximum horizon $n = 3$, and the initial state $S_0 = 0$ and Agent that select an action at random.



From the visualizations provided, we observe the decision-making process of the Agent within the "Toy" environment. Initially, the Agent, guided by the random policy $\pi^{(3)}$, selects the action $a_0 = +1$ when in state $s_0 = 0$ with a probability of 50%. This decision is strategically advantageous, as it yields a potential reward of 0.3, in contrast to a nil reward for the alternative action $a_0 = +2$. In the subsequent state $s_1 = 1$, the Agent, again took right action by selecting the $a_1 = +1$ with a probability of 50%, as dictated by the policy $\pi^{(2)}$. In the final step, the Agent makes the wrong choice, and selects $a_2 = +1$.

The overarching aim of the policy gradient is not merely to observe the Agent's behavior but to systematically train the Agent to enhance its decision-making capability. The primary goal is to develop and refine a set of policies, denoted as $\pi = (\pi^{(3)}, \pi^{(2)}, \pi^{(1)})$, that are tailored to maximize the cumulative reward by the end of the game. This involves an in-depth exploration and

optimization of the policies to ensure that each decision taken by the Agent, at every step-horizon maximizes the potential cumulative reward.

1.2 Set-up

In this section, we elaborate on the formalization of the environment and the fundamental concepts underlying our approach to fixed-horizon max-entropy RL.

- **States** $\mathcal{S}$: The set of all possible states in which the agent can find itself, $\mathcal{S}$ is assumed to be a continuous closed subset of $\mathcal{S} \subset \mathbb{R}^d$.
- **Actions** $\mathcal{A}$: Analogous to states, the action space is the set of all possible actions the agent can take and is also a closed subset of $\mathcal{A} \subset \mathbb{R}^{d'}$, typically considered state-independent for simplicity.
- **Markov Decision Process (MDP)**: Governed by the tuple $(\mathcal{S}, \mathcal{A}, p, p_{rew})$, an MDP captures the dynamics of the environment through the transition probabilities $p(s, a, s')$ and the reward function $p_{rew}(r|s, a)$.
- **Policy** π: A simple (stationary) policy $\pi : \mathcal{A} \times \mathcal{S} \to [0, 1]$ is a function where $\pi(\cdot|s)$ is the probability distribution over the action space A for each state s. The set of all simple policies is denoted by $\mathcal{P}$.
- **Set of Policies** $\mathcal{P}_n$: For a fixed horizon n, $\mathcal{P}_n$ represents the set of non-stationary policies $\pi = (\pi^{(1)}, \pi^{(2)} \ldots, \pi^{(n)})$ where each $\pi(i) \in \mathcal{P}$. An agent adheres to a policy $\pi \in \mathcal{P}_n$ by selecting actions according to $\pi^{(n)}$ through $\pi^{(1)}$ in a sequential manner for each episode of length n, thereby generating a path $S_0, A_0, S_1, A_1, \ldots, A_{n-1}, S_n$ with $A_i \sim \pi^{(n-i)}(\cdot|S_i)$ and $S_{i+1} \sim p(S_i, A_i, \cdot)$.
- **Value Function** V and **Action Function** Q:

  Value (V): The value function, denoted as $V(s)$, represents the expected return when starting from state s and following the agent's policy thereafter. It reflects the quality of a state, giving the long-term expected reward for being in that state. In general, the value function is defined as:

$$V_\pi(s) = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_k \mid S_0 = s \right],$$

  where r_k is the reward received after $k + 1$ time steps, γ is discount factor.
  However, in a case of the Max-entropy and fixed-horizon RL, we adapt slightly different definition. As in equation (1), we use rewards, that are regularized by an entropy factor, where the H penalty term is presented in form of the KL divergence[2] between the agent's

[2]The Kullback-Leibler (KL) divergence between two policies π and π', is defined as:

$$D_{KL}(\pi||\pi')(s) := \int_A \log \frac{\pi(a|s)}{\pi'(a|s)} \pi(da)$$

policy and a baseline policy $\bar{\pi}$. The n-step value function $V_\pi^{(n)}$ induced by a policy $\pi \in \mathcal{P}_n$, is defined as:

$$V_\pi^{(n)}(s) = \mathbb{E}_\pi \left[\sum_{k=0}^{n-1} \left(R_k - \tau D_{KL}(\pi^{(n-k)} || \bar{\pi})(S_k) \right) \middle| S_0 = s \right], \tag{4}$$

where, the obtained trajectory is sampled according to $A_k \sim \pi^{(n-i)}(\cdot \mid S_k)$, $S_{k+1} \sim p(S_k, A_k, \cdot)$ and $R_k \sim p_{rew}(S_k, A_k, \cdot)$ Typically, we will get rid-off discount factor and consider a set of Value functions per each state: $V_\pi(s) = (V_\pi^{(1)}(s), V_\pi^{(2)}(s), \dots V_\pi^{(n)}(s))$

Remark. The base line policy $\bar{\pi}$, could be used to provide some additional information, if needed. In this research, we would typically consider it to be uniformly distributed, to stimulate better exploration.

Q-value (Q): The Q-value or action-value function, denoted as $Q_\pi(s, a)$, gives the expected utility of taking an action a in a state s, and then following the agent's policy. It can be directly defined from a Value function:

$$Q_\pi^{(n)}(a, s) = r(a, s) + \int_S p(s, a, ds') V_\pi^{(n-1)}(s') \tag{5}$$

Remark. In real world practice, often we consider a deterministic transition probability $p(s_k, a_k, s_{k+1}) = \delta_{s', s_{k+1}}$ for some $s' \in \mathcal{S}$. In this case, we can rewrite equation (5), in simplified manner:

$$Q_\pi^{(n)}(a, s) = r(a, s) + V_\pi^{(n-1)}(s') \tag{6}$$

- **Objective Function**: Designed to measure the expected performance of a policy over a fixed horizon, the objective function is integral to the optimization process in RL.

$$J_n(\pi) := \int_{\mathcal{S}} V_\pi^{(n)}(s) \nu_0(\, \mathrm{d}s), \tag{7}$$

where we assume that the initial state distribution ν_0 has full support on $\mathcal{S}$.

- **Optimal Policy** π_*: A policy that maximizes the objective function over the fixed horizon is considered optimal. We would say that policy π_* is optimal if and only if

$$J_n(\pi_*) \geq J_n\left(\pi'\right) \text{ for all } \pi' \in \mathcal{P}_n.$$

- **Function approximation**: Let $\Theta^{(i)} : (\mathcal{A} \times \mathcal{S})^2 \to \mathbb{R}$ be a positive-semidefinite kernel.

  Following the notations and vocabulary used by Francois Ged, we define the preference $h_{\theta^{(i)}}^{(i)}(a, s) := \hat{Q}^{(i)}(a, s)$. Where, for each step horizon we consider linear parametrisation: for $i \in \{1, \dots, n\}$, let $\theta^{(i)} \in \mathbb{R}^{P_i}$ be the parameters of a linear model $h_{\theta^{(i)}}^{(i)} : \mathcal{A} \times \mathcal{S} \to \mathbb{R}$:

$$h_{\theta^{(i)}}^{(i)}(a, s) := \theta^{(i)} \cdot \psi^{(i)}(a, s),$$

where $\psi^{(i)} : \mathcal{A} \times \mathcal{S} \to \mathbb{R}^{P_i}$ is a feature map that is associated to the reproducible kernel Hilbert space (RKHS), denoted $\mathcal{H}_{\Theta^{(i)}}$. Then, i-step policy $\pi_{\theta^{(i)}}^{(i)}$ is defined as a Softmax policy with respect to the preference $h^{(i)}$:

$$\pi_{\theta^{(i)}}^{(i)}(a \mid s) := \bar{\pi}(a \mid s) \frac{\exp\left(h_{\theta^{(i)}}^{(i)}(a, s)/\tau\right)}{\int_{\mathcal{A}} \exp\left(h_{\theta^{(i)}}^{(i)}(a', s)/\tau\right) \bar{\pi}\left(\mathrm{d}a' \mid s\right)}.$$

Uniqueness and Existence of the Optimal Policy

To complete the theoretical framework of our setup, it is imperative to establish the fundamental properties of the optimal policy within the context of our model. This necessitates the formulation of certain technical lemmas and propositions that elucidate the existence and uniqueness of the optimal policy for the Max-entropy fixed horizon RL problem.

To delineate the construction of the "optimal" policy π_*, we consider the recursive definition of π_* across a fixed horizon. The policy is constructed as follows:

$$\pi_*^{(1)}(a \mid s) = \frac{\bar{\pi}(a \mid s)\exp(r(a,s)/\tau)}{\mathbb{E}_{\bar{\pi}}[\exp(r(A,s)/\tau)]}, \quad Q_{\pi_*}^{(2)}(a,s) = r(a,s) + \int_S p(s,a,ds')V_{\pi_*^{(1)}}(s'),$$

$$\pi_*^{(i)}(a \mid s) = \frac{\bar{\pi}(a \mid s)\exp(Q_*^{(i)}(a,s)/\tau)}{\mathbb{E}_{\bar{\pi}}[\exp(Q_*^{(i)}(A,s)/\tau)]}, \quad Q_{\pi_*}^{(i+1)}(a,s) = r(a,s) + \int_S p(s,a,ds')V_{\pi_*^{(i)}}(s'). \tag{8}$$

The initial policy $\pi_*^{(1)}$ is determined solely by the immediate reward function $r(a,s)$, scaled by the temperature parameter τ, and normalized by the expected exponential of the reward under the baseline policy $\bar{\pi}$. For subsequent horizons i, the policy $\pi_*^{(i)}$ is defined similarly, but it is based on the optimal action-value function $Q_*^{(i)}$, estimated using previous pilicy $\pi_*^{(i-1)}$.

This recursive formulation encapsulates the Bellman optimality[3] in an entropy-regularized setting, providing a clear pathway to the synthesis of the optimal policy through an iterative refinement process that balances immediate rewards with long-term strategic considerations.

Remark. Throughout this work, we will denote the "optimal" value and action-value functions, which are traditionally represented as V_{π_*} and Q_{π_*} respectively, by V_* and Q_*.

Lemma 1. *For all $n \geq 1$, all $\pi \in \mathcal{P}_n$ and all $s \in \mathcal{S}$, it holds that*

$$V_\pi^{(n)}(s) - V_*^{(n)}(s) = -\mathbb{E}_\pi\left[\sum_{i=0}^{n-1} D_{\mathrm{KL}}\left(\pi^{(n-i)} \| \pi_*^{(n-i)}\right)(S_i) \mid S_0 = s\right].$$

Proof. We commence the proof by rewriting the value function, that can be expanded as the expected immediate reward plus the expected value of the subsequent state:

$$\begin{aligned}
V_\pi^{(n)}(s) &= \mathbb{E}_\pi\left[\sum_{k=0}^{n-1}\left(R_k - \tau D_{KL}(\pi^{(n-k)}||\bar{\pi})(S_k)\right)\middle| S_0 = s\right] \\
&= \mathbb{E}_{a\sim\pi^{(n)}}\left[\mathbb{E}_\pi\left[\sum_{k=0}^{n-1}\left(R_k - \tau D_{KL}(\pi^{(n-k)}||\bar{\pi})(S_k)\right)\middle| S_0 = s, A_0 = a\right]\right] \\
&= \int_{\mathcal{A}} \pi^{(n)}(\mathrm{d}a \mid s)\left(\int_{\mathcal{S}} p\left(s,a,\ \mathrm{d}s'\right)\mathbb{E}_\pi\left[\sum_{k=1}^{n-1}(\ldots)\middle| S_1 = s'\right] + r(s,a) - \tau\log\frac{\pi^{(n)}}{\bar{\pi}}(a \mid s)\right) \\
&= \int_{\mathcal{A}} \pi^{(n)}(\mathrm{d}a|s)\left(r(a,s) - \tau\log\frac{\pi^{(n)}}{\bar{\pi}}(a|s) + \int_{\mathcal{S}} p(s,a,\mathrm{d}s')V_\pi^{(n-1)}(s')\right).
\end{aligned}$$

[3]Bellman Optimality Equation, a key equation used in the formulation of optimal control problems in Reinforcement Learning, see Chapter 3.6 in [8]

Now, utilizing the expression for the optimal policy $\pi_*^{(n)}$ (8), we substitute $V_*^{(n)}(s)$ and apply the definition of the Kullback-Leibler divergence to rewrite the expression:

$$V_\pi^{(n)}(s) = \int_{\mathcal{A}} \pi^{(n)}(\mathrm{d}a|s) \left(V_*^{(n)}(s) - \tau \log \frac{\pi^{(n)}}{\pi_*^{(n)}}(a|s) + \int_{\mathcal{S}} p(s,a,\mathrm{d}s') \left(V_\pi^{(n-1)}(s') - V_*^{(n-1)}(s') \right) \right).$$

Now, we focus on the deviation of the policy π from optimality:

$$V_\pi^{(n)}(s) - V_*^{(n)}(s) = -D_{\mathrm{KL}} \left(\pi^{(n)} \| \pi_*^{(n)} \right)(s) + \mathbb{E}_\pi \left[V_\pi^{(n-1)}(S_1) - V_*^{(n-1)}(S_1) | S_0 = s \right].$$

The negative KL divergence term represents the loss due to deviation from the optimal policy at the current step. The expectation term captures the cumulative effect of this deviation in the future.

By applying the principle of induction, we extend this argument to all steps from 0 to $n-1$, thus establishing the lemma. □

Proposition 1. *For all $s \in \mathcal{S}$ and $n \geq 1$, it holds that*

$$V_*^{(n)}(s) = \log \mathbb{E}_{\bar{\pi}} \left[\exp \left(Q_*^{(n)}(A,s)/\tau \right) \right].$$

Proof. Recall the derivation for the value function in lemma 1 and write:

$$\begin{aligned} V_*^{(n)}(s) &= \int_{\mathcal{A}} \pi_*^{(n)}(\mathrm{d}a|s) \left(r(a,s) - \tau \log \frac{\pi_*^{(n)}}{\bar{\pi}}(a|s) + \int_{\mathcal{S}} p(s,a,\mathrm{d}s') V_\pi^{(n-1)}(s') \right). \\ &= \int_{\mathcal{A}} \pi_*^{(n)}(\mathrm{d}a \mid s) \left(Q_*^{(n)}(a,s) - \tau \log \frac{\pi_*^{(n)}}{\bar{\pi}}(a \mid s) \right), \end{aligned}$$

plug-in the expression for the π_* (8):

$$\begin{aligned} V_*^{(n)}(s) &= \int_{\mathcal{A}} \bar{\pi}(\mathrm{d}a \mid s) \frac{\exp \left(Q_*^{(n)}(a,s)/\tau \right)}{\mathbb{E}_{\bar{\pi}} \left[\exp \left(Q_*^{(n)}(A,s)/\tau \right) \right]} \tau \log \mathbb{E}_{\bar{\pi}} \left[\exp \left(Q_*^{(n)}(A,s)/\tau \right) \right] \\ &= \tau \log \mathbb{E}_{\bar{\pi}} \left[\exp \left(Q_*^{(n)}(A,s)/\tau \right) \right]. \end{aligned}$$

□

Remark. Thanks to the Proposition 1 one, we can express optimal policy concisely:

$$\pi_*^{(i)}(a \mid s) = \bar{\pi}(a \mid s) \exp \left(\left(Q_*^{(i)}(a,s) - V_*^{(i)}(s) \right) / \tau \right)$$

Although we have delineated the construction of π_* in prior sections, we have yet to formally establish its optimality. The subsequent theorem will addresses this issue.

Theorem 1 (Existence and Uniqueness of the Optimal Policy)**.** *There exists a unique optimal policy (Lebesgue almost-everywhere), denoted by* $\pi_* = (\pi_*^{(1)}, \dots, \pi_*^{(n)}) \in \mathcal{P}_n$.

Proof. Existence, of such policy follows directly from the contraction (8). To proof optimality and uniqueness, we racall the Lemma 1, where the difference in value functions between any policy π and the optimal policy π_* is given by the sum of KL divergences. Given that the KL divergence is always non-negative, it follows that the value function $V_\pi^{(n)}(s)$ is maximized when these divergences are zero, which occurs if and only if $\pi = \pi_*$ almost everywhere with respect to the Lebesgue measure.

Thus, the optimal policy π_* is unique (Lebesgue-almost everywhere) and maximizes the objective function J_n. □

Assumption 1. *For the MDP framework and policy evaluation to be well-defined and for the theoretical results to hold, the following assumptions are made:*

- *The MDP is irreducible; no state is transient under any policy in* $\mathcal{P}_n$.
- *The initial state distribution* μ_0 *has full support on* S*, ensuring that all states are reachable.*
- *The reward function* $r(a, s)$ *and the transition probabilities* $p(s, a, s')$ *are continuous with respect to the Euclidean metric. We assume that rewards are uniformally bounded, thus there is exist some positive constant* $U > 0$ *such that* $r(a, s) < U$ *for any* $(a, s) \in \mathcal{A} \times \mathcal{S}$.

1.3 Original work and Matryoshka gradient policy

The foundation of our discussion is built on the seminal work "Matryoshka Policy Gradient for Entropy-Regularized RL: Convergence and Global Optimality", by Francois and Maria [3]. This work presents a comprehensive theoretical understanding of the Max-entropy fixed horizon policy.

A cornerstone of this work is the theorem on the global convergence to the unique optimal policy. Furthermore, we get introduced to the Matryoshka Policy Gradient (MPG), a practical algorithm that estimates the gradient ascent of the proposed objective function (see eq. 7).

The MPG Update

In this paragraph, we introduce the Matryoshka Policy Gradient (MPG), a novel approach designed to update policies within the Max-entropy RL framework. The MPG algorithm is predicated on the idea of providing unbiased estimates for gradient ascent to optimize policies over a fixed horizon. The MPG update is formally defined by the following iterative process:

$$\theta_{t+1}^{(i)} = \theta_t^{(i)} + \eta \sum_{\ell=n-i}^{n-1} \left(R_\ell - \tau \log \frac{\pi_t^{(n-\ell)}}{\bar{\pi}} \left(A_\ell \mid S_\ell \right) \right) \nabla \log \pi_t^{(i)} \left(A_{n-i} \mid S_{n-i} \right), \tag{9}$$

At each training timestep t, the MPG algorithm updates the policy parameters θ_t for a fixed horizon n using the gradient of the expected return. Where η is the learning rate, τ is the temperature parameter controlling the trade-off between reward maximization and entropy, R_ℓ is the reward collected at step ℓ, and $\pi_t^{(i)}$ is the policy at timestep t for horizon i.

Algorithm 1 Matryoshka Policy Gradient (MPG) Update

Input: Learning rate η, temperature τ, initial policy parameters θ_0, fixed horizon n
Initialize policy parameters $\theta = \theta_0$
for each training timestep $t = 0, 1, 2, \ldots$ **do**
    Sample initial state $S_0 \sim \nu_0$
    for $i = 1$ to n **do**
        Sample action A_{n-i} according to $\pi_t^{(i)} \left(\cdot \mid S_{n-i} \right)$
        Collect reward $R_{n-i} \sim p_{\text{rew}} \left(\cdot \mid S_{n-i}, A_{n-i} \right)$
        Move to next state $S_{n-i+1} \sim p \left(S_{n-i}, A_{n-i}, \cdot \right)$
    end for
    for $i = 1$ to n **do**
        Compute cumulative reward and entropy term:
        $C_i = \sum_{\ell=n-i}^{n-1} \left(R_\ell - \tau \log \frac{\pi_t^{(n-\ell)}}{\bar{\pi}} \left(A_\ell \mid S_\ell \right) \right)$
        Update policy parameters:
        $\theta_{t+1}^{(i)} = \theta_t^{(i)} + \eta C_i \nabla \log \pi_t^{(i)} \left(A_{n-i} \mid S_{n-i} \right)$
    end for
    Update θ_t to θ_{t+1}
end for
Output: Optimized policy parameters θ

Theoretical Analysis of MPG

In this section, we succinctly recapitulate the key theoretical insights from Francois's research on the Matryoshka Policy Gradient (MPG).

Definition 1. *(Policy convergence) A sequence of policies $(\pi_t)_{t\in\mathbb{N}}$ converges to a policy $\pi \in \mathcal{P}_n$ if and only if*

$$\max_{i\in\{1,\dots,n\}} \int_{\mathcal{A}\times\mathcal{S}} m^{(i)}(\mathrm{d}s) \left|\pi_t^{(i)}(a \mid s) - \pi^{(i)}(a \mid s)\right| \mathrm{d}a \to 0 \quad \textit{as} \quad t \to \infty,$$

where, $m_\theta^{(i)}(\mathrm{d}s)$ denote the state distribution for i-th step horizon.

This definition lays the groundwork for our next significant theorem, which addresses the global convergence of policies under the MPG framework.

Theorem 2 (Global policy convergence)**.** *With MPG as defined, it holds that $\mathbb{E}[\theta_{t+1} - \theta_t] = \eta\nabla_\theta J_n(\pi_t)$. In addition, assuming an ideal MPG update and a learning rate $0 < \eta < \eta_0$, policy π_t converges as $t \to \infty$ to a policy $\pi_\infty \in \mathcal{P}_n$.*

Proof. (see Theorem 1 in [3]) □

The theorem asserts that, under ideal conditions[4], the MPG method ensures the convergence of policies to a certain limit within the policy space. However, this convergence is contingent on the realizability of the optimal policy, as captured by the following assumption:

Assumption 2 (Realizability)**.** *There exists $\theta_* \in \mathbb{R}^P$ such that $\pi_{\theta_*} = \pi_*$.*

The realizability assumption (2) posits the existence of a parameter set that characterizes the optimal policy. This assumption is critical in proving the convergence of MPG to the optimal policy, as stated in the next theorem.

Theorem 3 (Convergence to the optimal in the Realizable Case)**.** *Training with the ideal MPG update under the realizability assumption A1 converges to the optimal policy π_*.*

Proof. (see Theorem 2 in [3]) □

While the realizability assumption is a strong condition, we extend our analysis to scenarios where this assumption may not hold, termed the non-realizable cases. In these scenarios, the following theorem provides insights into the nature of convergence.

Theorem 4 (Global Convergence Beyond the Realizability Assumption)**.** *There exists $\eta_0 > 0$ such that for all $0 < \eta < \eta_0$, training with ideal MPG update converges and $\lim_{t\to\infty} \pi_t = \pi_{\theta_*}$, where $\pi_{\theta_*} = \mathrm{argmax}_{\pi_\theta\in\mathcal{P}_n} J_n(\pi_\theta)$ is unique and is the only critical point of J_n.*

Proof. (see Theorem 3 in [3]) □

[4]The ideal update equation $\theta_{t+1} - \theta_t = \eta\nabla_\theta J_n(\pi_t)$ is a common assumption in theoretical RL. It assumes perfect alignment of parameter updates with the gradient of the objective function. While ideal for theoretical clarity, real-world "model-free" scenarios, like ours, typically lack direct access to precise gradient information. In this project, our theoretical discussions assume this ideal update, but for practical experiments in Section 3.3, we employ a Stochastic Gradient Update. This practical approach approximates the ideal update, especially with large batches, as described by the expectation result $\mathbb{E}[\theta_{t+1} - \theta_t] = \eta\nabla_\theta J_n(\pi_t)$ obtained in Theorem 2.

Lastly, we define the projectional consistency property and establish its relation to the global optimum:

Definition 2. *(Projectional Consistency) Let $\theta \in \mathbb{R}^P$ and $\mathbf{m}^{(i)}$ denote the law of state S_{n-i} under the policy π_θ, with $\mathbf{m}^{(n)} = \nu_0$. For each $i \in \{1, \dots, n\}$, define $P_i : L^2\left(\mathbf{m}^{(i)}(\mathrm{d}s)\pi_\theta^{(i)}(\mathrm{d}a \mid s)\right) \to \mathcal{H}_{\Theta^{(i)}}$ as the orthogonal projection onto $\mathcal{H}_{\Theta^{(i)}}$ in the $L^2\left(\mathbf{m}^{(i)}(\mathrm{d}s)\pi_\theta^{(i)}(\mathrm{d}a \mid s)\right)$ sense. A policy π_θ is said to satisfy the projectional consistency property if and only if for all $i \in \{1, \dots, n\}$, the following holds:*

$$\pi_\theta^{(i)}(a \mid s) = \bar{\pi}(a \mid s)\frac{\exp\left(P_i Q_{\pi_\theta}^{(i)}(a, s)/\tau\right)}{\int_{\mathcal{A}} \bar{\pi}\left(\mathrm{d}a' \mid s\right) \exp\left(P_i Q_{\pi_\theta}^{(i)}\left(a', s\right)/\tau\right)}.$$

Proposition 2. *The global optimum π_{θ_*} from Theorem 4 is the only policy in $\mathcal{P}_n$ that satisfies the projectional consistency property.*

Proof. (see Proposition 1 in [3])

□

This chapter has presented a detailed exploration of the Matryoshka Policy Gradient (MPG), a significant contribution to the field of entropy-regularized reinforcement learning. Our discussion can be viewed through three distinct theoretical stages:

1. **Global Convergence:** We established the global policy convergence theorem, demonstrating that MPG leads to convergence of policies under ideal conditions.

2. **Convergence in the Realizable Case:** The realizability assumption, a crucial aspect of our theoretical framework, facilitated the proof that MPG converges to the optimal policy when this assumption holds.

3. **Convergence in Non-Realizable Cases:** Extending our analysis beyond the bounds of realizability, we explored the convergence of MPG to the unique global maximizer in non-realizable cases.

1.4 Contribution

Our research focuses on expanding the theoretical and practical framework of the Matryoshka Policy Gradient (MPG), originally developed by François Gaston Ged and Maria Han Veiga. While reobtaining and testing François's findings, our work primarily aims to advance the MPG methodology further. The MPG, as presented in original work, demonstrates significant potential in entropy-regularized reinforcement learning. However, a notable limitation lies in its parameter-heavy architecture, particularly for games with extensive horizons. The traditional MPG requires a separate parameter set for each step horizon, leading to impracticalities in large-horizon scenarios. Our primary objective is to address this by introducing a "shared parameters" version of the MPG. This new approach envisions a model where parameters for shorter horizons contribute to the outputs of longer horizons, thereby reducing the overall parameter burden.

Shared Parameters MPG

We propose a novel way of parametrization for the MPG, which we refer to as the "shared parameters" approach. This method aims to streamline the parameterization process, making it more efficient and practical, especially for complex problems with long horizons. We will adapt the three main theorems obtained by François to fit this new parametrization framework, thereby establishing a solid theoretical foundation for the shared parameters MPG.

Neural Network Architectures and Benchmark Testing

Building on the new parametrization approach, we plan to design several Neural Network architectures tailored to the shared parameters MPG. These architectures will be tested against popular benchmarks to evaluate their performance. This empirical investigation will provide insights into the practical applicability of the shared parameters approach in real-world scenarios.

Summary of Contributions

Our contributions to the field of reinforcement learning and policy optimization through MPG can be summarized as follows:

- Developing a "shared parameters" version of the MPG, thereby addressing the issue of heavy parameterization in large-horizon games.
- Designing and testing novel Neural Network architectures based on the shared parameters approach, evaluating their performance against established benchmarks.

2 Shared Parameters in MPG

The development of the Matryoshka Policy Gradient has marked a significant advancement in the field of Reinforcement Learning, particularly in the context of entropy-regularized, fixed-horizon policy optimization. Building on the foundational work of François Gaston Ged, this section introduces an innovative extension to the MPG framework - the concept of Shared Parameters. This approach addresses the complexity arising from the need for distinct parameter sets for each step horizon in large-horizon games, thus enhancing the MPG's practical applicability and computational efficiency.

2.1 Parametrization and Update with Shared Parameters

The core of this new approach lies in the reformulation of the preference function. The parameter sets are designed such that parameters for shorter horizons are subsets of those for longer horizons, ensuring a coherent and cumulative learning process across different horizons. For each $i \in \{1, \dots, n\}$, we decompose the parameters into two parts: the inherited parameters $\theta^{(i-1)}$ from the previous horizon, and the new trainable parameters $\vartheta^{(i)}$ unique to the current horizon. Specifically, $\theta^{(i)} \in \mathbb{R}^{P_1+\dots+P_i}$, the parameter vector for the i-step policy, is the concatenation of $\theta^{(i-1)} \in \mathbb{R}^{P_1+\dots+P_{i-1}}$ and $\vartheta^{(i)} \in \mathbb{R}^{P_i}$.

Example of Parametrization Consider an example with $n = 3, P_1 = 3, P_2 = 5, P_3 = 6$. The hierarchical construction of the preference functions can be represented as follows:

$$\begin{aligned} h^{(1)} &= \theta_1 e_1 + \theta_2 e_2 + \theta_3 e_3, \\ h^{(2)} &= \theta_1 e_1 + \theta_2 e_2 + \theta_3 e_3 + \theta_4 e_4 + \theta_5 e_5, \\ h^{(3)} &= \theta_1 e_1 + \theta_2 e_2 + \theta_3 e_3 + \theta_4 e_4 + \theta_5 e_5 + \theta_6 e_6. \end{aligned} \tag{10}$$

where the components that are updated during training are those highlighted in blue for the corresponding horizon, while the other come from shorter horizon policies.

Feature Maps and Preference Functions

Correspondingly, the feature map $\psi^{(i)} : \mathcal{A} \times \mathcal{S} \to \mathbb{R}^{P_1+\dots+P_i}$ for the i-step policy is a concatenation of $\psi^{(i-1)}(a, s) \in \mathbb{R}^{P_1+\dots+P_{i-1}}$ and $\phi^{(i)}(a, s) \in \mathbb{R}^{P_i}$. The preference function, $h_\theta^{(i)}(a, s)$, is then given by:

$$h_\theta^{(i)}(a, s) = h_\theta^{(i-1)}(a, s) + g_\theta^{(i)}(a, s), \tag{11}$$

where $g_\theta^{(i)}(a, s) := \vartheta^{(i)} \cdot \phi^{(i)}(a, s)$ represents the contribution of the new parameters to the preference function at horizon i.

Gradient Computation

The gradient computation for the i-step policy, with respect to $\vartheta^{(j)}$ ($\nabla_j = \nabla_{\vartheta^{(j)}}$), is defined as:

$$\nabla_j \pi_\theta^{(i)}(a \mid s) = \pi_\theta^{(i)}(a \mid s) \int_{\mathcal{A}} \left(\delta_a(da') - \pi_\theta^{(i)}(da' \mid s) \right) \nabla_j h_\theta^{(i)}(a', s),$$

where $\nabla_j h_\theta^{(i)}(a, s) = \phi^{(j)}(a, s)$ if $j \leq i$, and is zero otherwise.

$$\nabla_j \pi_\theta^{(i)}(a \mid s) = \pi_\theta^{(i)}(a \mid s) \int_{\mathcal{A}} \left(\delta_a(da') - \pi_\theta^{(i)}(da' \mid s) \right) \phi^{(j)}(a', s). \tag{12}$$

MPG Update with Shared Parameters

The algorithm delineated in (1.3) for the MPG update has been meticulously designed to ensure that $\mathbb{E}[\theta_{t+1} - \theta_t] = \nabla J$, as substantiated by the arguments provided in the proof of theorem (1). However, it is critical to acknowledge that the algorithm (1.3) does not precisely mirror the gradient of the objective function. To address this, we propose a revised update rule, predicated on a thorough examination of the gradient of the objective function. Before, we dive into the calculations, let us derive small result, that will facilitate further derivations.

Proposition 3. *(Softmax property) For all $s \in \mathcal{S}$ and any function $f : S \to \mathbb{R}$, it holds that*

$$\mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(A \mid s) f(s)\right] = 0.$$

Proof.

$$\mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(A \mid s) f(s)\right] = \int_{\mathcal{A}} \nabla_\theta \pi_\theta(\mathrm{d}a \mid s) f(s) = f(s) \nabla \left(\int_{\mathcal{A}} \pi_\theta(\mathrm{d}a \mid s) \right) = f(s) \nabla(1) = 0$$

□

For brevity we introduce $m^{(l)}(\cdot)$, that will represent the law of S_{n-l} and $m^{(n)}$ is the initial state distribution. Note, that due to the Markov property we can write:

$$m^{(l)}(s) = \prod_{i=0}^{l} p\left(s_{n-i} a_{n-i}, s_{n-i-1}\right) \pi^{(n-i)}\left(a_{n-i} | s_{n-i}\right) = \prod_{i=0}^{l} p\left(s_{n-i}, a_{n-i}, s_{n-i-1}\right) \prod_{i=0}^{l} \pi^{(n-i)}\left(a_{n-i} | s_{n-i}\right),$$

where $(s_0, a_0, s_1, s_2, ...)$ is an Agent's trajectory.

For $n \in \mathbb{N}$ final horizon, consider the objective gradient:

$$\begin{aligned}
\nabla_\theta\, J_n(\theta) &= \nabla_\theta \iint_{S \times \mathcal{A}} m^{(n)}(ds) \pi_\theta^{(n)}(da \mid s) \left(r(a,s) - \tau \log \frac{\pi^{(n)}(a \mid s)}{\bar{\pi}(a \mid s)} + \int_S p\left(a, s, ds'\right) V_\theta^{(n-1)}\left(s'\right) \right) = \\
&= \iint_{S \times \mathcal{A}} m^{(n)}(ds) \nabla_\theta \pi_\theta^{(n)}(da \mid s)(r(a,s) - \ldots) + \iint_{S \times \mathcal{A}} m^{(n)}(ds) \pi_\theta^{(n)}(da \mid s) \nabla (r(a,s) - \ldots) \\
&= \iint m^{(n)}(ds) \pi_\theta^{(n)}(da|s) \nabla \log \pi_\theta^{(n)} \left(r(a,s) - \tau \log \frac{\pi_\theta^{(n)}}{\bar{\pi}} + \int_S p\left(s, a, ds'\right) V_\theta^{(n-1)}\left(s'\right) - \tau \right) + \\
&+ \iint m^{(n)}\left(ds\right) \pi_\theta^{(n)}(da|s) \int_s p\left(s, a, ds'\right) \nabla_\theta V_\theta^{(n-1)}\left(s'\right)
\end{aligned}$$

Then, by applying the softmax property, we get:

$$\Rightarrow \nabla_\theta\, J_n(\theta) = \mathbb{E}_{\pi_\theta}\left[C_n \nabla_\theta \log \pi^{(n)}\left(A_0, S_0\right) \right] + \mathbb{E}_{\pi_\theta}\left[\nabla_\theta V_\theta^{(n-1)}\left(S_1\right) \right],$$

where, $C_n = r(a,s) - \tau \log \frac{\pi^{(n)}}{\bar{\pi}} + \int_S p(s,a,ds') V_\theta\left(s'\right)$. Let us consider the second term of the sum:

$$\begin{aligned}
\mathbb{E}_{\pi_\theta}\left[\nabla_\theta V_\theta^{(n-1)}\left(S_1\right) \right] = \iint_{S \times \mathcal{A}} m^{(n-1)}(ds) \nabla_\theta (\pi_\theta^{(n-1)}(da \mid s)(r(a,s) - \tau \log \frac{\pi^{(n-1)}(a \mid s)}{\bar{\pi}(a \mid s)} \\
+ \int_S p\left(a, s, ds'\right) V_\theta^{(n-2)}\left(s'\right))).
\end{aligned}$$

We notice the same pattern, and we can write in recursion for $i \in [n]$:

$$\nabla_\theta V_\theta^{(i)}(S_{n-i}) = \mathbb{E}_{\pi_\theta}\left[C_i \nabla_\theta \log \pi_\theta^{(i)}(A_{n-i} \mid S_{n-i})\right] + \mathbb{E}_{\pi_\theta}\left[\nabla_\theta V_\theta^{(i-1)}(S_{n-i+1})\right]$$

Then, this recursion bring us to the following expression:

$$\nabla_\theta J_n(\pi_\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{i=0}^{n-1} C_{n-i} \nabla_\theta \log \pi_\theta^{(n-i)}(A_i \mid S_i)\right].$$

Note, that we can write $\nabla_\theta \log \pi_\theta^{(n-i)}(A_i \mid S_i) = \nabla_{\theta^{(n-i)}} \log \pi_\theta^{(n-i)}(A_i \mid S_i)$. The our ideal gradient update would be:

$$\nabla_\theta J_n(\pi_\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{i=0}^{n-1} C_{n-i} \nabla_{\theta^{(n-i)}} \log \pi_\theta^{(n-i)}(A_i \mid S_i)\right]. \tag{13}$$

Now, using equation (13) we can derive formulation for the gradient w.r.t the subset of parameters:

- Gradient w.r.t $\theta^{(l)}$:

$$\nabla_{\theta^{(l)}} J_n(\pi_\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{i=0}^{n-1} C_{n-i} \nabla_{\theta^{(n-i)\wedge l}} \log \pi_\theta^{(n-i)}(A_i \mid S_i)\right].$$

- Gradient w.r.t $\vartheta^{(l)}$:

$$\nabla_{\vartheta^{(l)}} J_n(\pi_\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{i=0}^{n-l} C_{n-i} \nabla_{\vartheta^{(l)}} \log \pi_\theta^{(n-i)}(A_i \mid S_i)\right].$$

To derive the last statement we used the fact that $\nabla_{\vartheta^{(i)}} \log \pi_\theta^{(m)}(a \mid s) = 0$, for all $m < i$.

S-MPG

Following the same structure as in original MPG algorithm (1.3), we define the *S-MPG* (S stands for "Share") which represents practical implementation for the ideal gradient update (13). For $t \in \mathbb{N}$ and for $i = 1, \dots n$, we define:

$$\begin{aligned}\theta_{t+1}^{(i)} &= \theta_t^{(i)} + \eta \sum_{\ell=n-i}^{n-1} \left(R_\ell - \tau \log \frac{\pi_t^{(n-\ell)}}{\bar{\pi}}(A_\ell \mid S_\ell)\right) \nabla_{\theta^{(i)}} \log \pi_t^{(i)}(A_{n-i} \mid S_{n-i}), \\ &= \theta_t^{(i)} + \eta C_i \nabla_{\theta^{(i)}} \log \pi_t^{(i)}(A_{n-i} \mid S_{n-i}),\end{aligned} \tag{14}$$

where the trajectory $(S_0, S_1, A_1, \dots, S_n)$ was sampled under the Agent's policy π_{θ_t}. We note, that the whole subset of variables $\theta^{(i)}$ are moved, during the training of the step-horizon i and not only the parameters associated to the underlying horizon level.

Proposition 4. *For the S-MPG update defined in (2.1), it holds that* $\mathbb{E}[\theta_{t+1} - \theta_t] = \nabla_\theta J_n$

Proof. By construction. □

Algorithm 2 Shared Matryoshka Policy Gradient (S-MPG) Update

Input: Learning rate η, temperature τ, initial policy parameters θ_0, fixed horizon n
Initialize policy parameters $\theta = \theta_0$
for each training timestep $t = 0, 1, 2, \ldots$ **do**
  Sample initial state $S_0 \sim \nu_0$
  for $i = 1$ to n **do**
    Sample action A_{n-i} according to $\pi_t^{(i)}(\cdot \mid S_{n-i})$
    Collect reward $R_{n-i} \sim p_{\text{rew}}(\cdot \mid S_{n-i}, A_{n-i})$
    Move to next state $S_{n-i+1} \sim p(S_{n-i}, A_{n-i}, \cdot)$
  end for
  for $i = 1$ to n **do**
    Compute cumulative reward and entropy term:
    $C_i = \sum_{\ell=n-i}^{n-1} \left(R_\ell - \tau \log \frac{\pi_t^{(n-\ell)}}{\bar{\pi}}(A_\ell \mid S_\ell) \right)$
    Update policy parameters:
    $\theta_{t+1}^{(i)} = \theta_t^{(i)} + \eta C_i \nabla_{\theta^{(i)}} \log \pi_t^{(i)}(A_{n-i} \mid S_{n-i})$
  end for
  Update θ_t to θ_{t+1}
end for
Output: Optimized policy parameters θ

2.2 On the Convergence

In the spirit of the foundational theorem (2) presented in the original work, this section aims to establish an analogous result for the ideal gradient update within the shared parameters setting. To achieve this, we will introduce and expound upon a series of technical lemmas and properties. These foundational elements are crucial for not only demonstrating the convergence of the algorithm but also for extracting insights into the convergence rate.

In addition to the assumptions delineated in the original work by Francois (refer to 1), this study introduces a new set of assumptions. While these assumptions may impose more stringent constraints on the theoretical framework, they are not anticipated to impede the practical effectiveness.

Assumption 3.
For the case where $\mathcal{A}$ is a discrete action space, the following conditions are assumed:

- *The action space is finite, i.e., $|\mathcal{A}| < \infty$.*
- *For every pair $(a, s) \in \mathcal{A} \times \mathcal{S}$, there exists $\epsilon \in \mathbb{R}$ such that $\bar{\pi}(a|s) > \epsilon$.*

In scenarios where $\mathcal{A}$ is a continuous action space, the assumptions are as follows:

- *The action space $\mathcal{A}$ is bounded with respect to the Lebesgue measure, specifically $\mu(\mathcal{A}) < \infty$.*
- *For the baseline policy $\bar{\pi}$, only a uniform policy is considered, defined as $\bar{\pi}(da|s) = da/\mu(A)$.*

Our analysis begins by illustrating the L-smoothness property of the log-policy.

Lemma 2 (Lipschitz smooth log-policy)**.** *Let $t \in \mathbb{N}$ and $i \in \{1, \dots, n\}$. It holds, that for $\forall (a, s) \in \mathcal{A} \times \mathcal{S}$ with $|\mathcal{A}| \in \mathbb{N}$, the log policy $\log \pi_\theta^{(i)}(a|s)$ is L-Lipschitz gradient smooth w.r.t parameters θ:*

$$\left\| \nabla \log \pi_{\theta_2}^{(i)}(a \mid s) - \nabla \log \pi_{\theta_1}^{(i)}(a \mid s) \right\|_2 \leq L \left\| \theta_2 - \theta_1 \right\|_2 ,$$

with $L = \max_{\substack{(a,s)\in A\times S \\ i\in[n]}} \left\| \psi^{(i)}(a, s) \right\| / \tau^2$.

Remark. For simplicity, we will provide the proof for the discrete case, however the result generalize for the continuous case, as well.

Proof. Recall the policy definition and write:

$$\begin{aligned}\nabla_\theta \log \pi_{\theta_1}^{(i)}(a \mid s) =& \nabla_\theta \log \left(\frac{e^{h_{\theta_1}^{(i)}(a,s)/\tau}}{\sum_{a' \in A} e^{h_{\theta_1}^{(i)}(a',s)/\tau}} \right) = \nabla_\theta \left(h_{\theta_1}^{(i)}(a, s)/\tau - \log \left(\sum_{a' \in A} e^{h_{\theta_1}^{(i)}(a',s)/\tau} \right) \right) = \\ &= \psi^{(i)}(a, s)/\tau - \nabla_\theta \log \left(\sum_{a' \in A} e^{h_{\theta_1}^{(i)}(a's)/\tau} \right)\end{aligned}$$

We abuse the notation and say, that $z_a^{\theta_1} := h_{\theta_1}^{(i)}(a, s)$ and $lse(z_a^{\theta_1}) := \log \left(\sum_{a \in A} e^{z_a^{\theta_1}} \right)$ then:

$$\left\| \nabla_\theta \log \pi_{\theta_2}(a \mid s) - \nabla_\theta \log \pi_{\theta_1}(a \mid s) \right\| = \left\| \nabla_\theta lse(z^{\theta_2}) - \nabla_\theta lse(z^{\theta_1}) \right\| .$$

Recall the differential rule:

$$\nabla_\theta lse(z^{\theta_1}) = \nabla_\theta \left(h_{\theta_1}^{(i)}(a, s) / \tau \right)^T \nabla_{z^{\theta_1}} lse(z^{\theta_1}) = \frac{1}{\tau} \psi^{(i)}(\cdot, s)^T \nabla_{z^{\theta_1}} lse(z^{\theta_1}),$$

then we can write

$$\left\| \nabla_\theta \log \pi_{\theta_2}(a \mid s) - \nabla_\theta \log \pi_{\theta_1}(a \mid s) \right\| \leq \max_{a \in \mathcal{A}} \left\| \psi^{(i)}(a, s)/\tau \right\| \left\| \nabla_{z^{\theta_2}} lse(z^{\theta_2}) - \nabla_{z^{\theta_1}} lse(z^{\theta_1}) \right\| .$$

We refer to proposition 4 in the paper [2], which shows that *lse* has a Lipschitz continuous gradient with Lipschitz constant $L' > 0$. Moreover, the constant L can be explicitly found, and equal to the inverse of the temperature factor, $L' = 1/\tau$. Then, we write:

$$\left\| \nabla \log \pi_{\theta_2}^{(i)}(a \mid s) - \nabla \log \pi_{\theta_1}^{(i)}(a \mid s) \right\|_2 \leq L \left\| \theta_2 - \theta_1 \right\|_2 ,$$

where, $L_i = \max_{(a,s)\in\mathcal{A}\times\mathcal{S}} \left\| \psi^{(i)}(a, s) \right\| / \tau^2$. Therefore, we set $L = \max_{i\in[n])} L_i$. □

Lemma 3 (log policy bound)**.** *For any $(a, s) \in \mathcal{A} \times \mathcal{S}$, there exist constant $B > 0$ such that for all $\theta \in \Theta$, holds:*

$$\left\| \nabla \log \pi_\theta^{(i)}(a|s) \right\| \leq B \qquad \text{for } i = 1, 2 \dots n,$$

with $B = \frac{2}{\tau} \max_{\substack{(a,s)\in A\times S \\ i\in[n]}} \left\| \psi^{(i)}(a, s) \right\|$.

Proof.

$$\|\nabla log\pi_\theta^{(i)}\| = \frac{1}{\tau}\| \int_{\mathcal{A}} \left(\delta_a(da') - \pi_\theta^{(i)}(da' \mid s)\nabla_\theta h_\theta^{(i)}(a', s)\right\| \\ \leq \frac{1}{\tau}\| \int_{\mathcal{A}} \left(|\delta_a(da')| + |\pi_\theta^{(i)}(da' \mid s)|\right) |\psi^{(i)}(a', s)|\| \leq \frac{2}{\tau} \max_{\substack{(a,s)\in A\times S \\ i\in[n]}} \left\|\psi^{(i)}(a, s)\right\|$$

□

Lemma 4 (Lipschitz Cumulative reward)**.** *For any $\theta_1, \theta_2 \in \Theta$, it holds that cumulative reward is Lipschitz, for some constant $L' > 0$:*

$$\|C_n^{\theta_2} - C_n^{\theta_1}\| \leq L'\|\theta_2 - \theta_1\|$$

with $L' = \tau(n-1)B$.
Remark. the notation C_n^θ explicit the policy regularization by θ.

Proof. Recall the expression for C_n^θ:

$$C_n^\theta = \sum_{l=0}^{n-1} \left(R_l - \tau log \frac{\pi_\theta^{(n-l)}}{\bar{\pi}}(A_l|S_l) \right).$$

Then,

$$\|C_n^{\theta_2} - C_n^{\theta_1}\| = \tau\| \sum \log \pi_{\theta_1}^{(n-l)}(A_l|S_l) - \sum \log \pi_{\theta_2}^{(n-l)}(A_l|S_l)\|$$

$$\leq \tau \sum_{l=0}^{n-1} \| \log \pi_{\theta_1}^{(n-l)}(A_l|S_l) - \log \pi_{\theta_2}^{(n-l)}(A_l|S_l)\| \leq \max_{\theta\in\Theta} \|\nabla_\theta \log \pi_\theta^{(n-l)}(A_l|S_l)\|\|\theta_2 - \theta_1\|$$

We recall the result of lemma 3 $\|\nabla \log \pi_\theta^{(i)}(a|s)\| \leq B$, which concludes the proof and we set

$$L' = \tau(n-1)B = 2(n-1) \max_{\substack{(a,s)\in A\times S \\ i\in[n]}} \left\|\psi^{(i)}(a, s)\right\|.$$

□

Lemma 5 (Cumulative reward bound)**.** *For any $\theta \in \Theta$, there exist constant $B' > 0$ that bounds entropy regularized cumulative reward:*

$$\mathbb{E}_{\pi_\theta}[|C_n^\theta|] \leq B'.$$

Proof.

$$\mathbb{E}_{\pi_\theta}[|C_n^\theta|] = \mathbb{E}_{\pi_\theta}\left[\left|\sum_{l=0}^{n-1} \left(R_l - \tau log \frac{\pi_\theta^{(n-l)}}{\bar{\pi}}(A_l|S_l) \right)\right|\right] \\ \leq (n-1)U + \tau \sum_{l=0}^{n-1} \mathbb{E}_{\pi_\theta}\left[\left|\log \frac{\pi_\theta^{(n-l)}}{\bar{\pi}}(A_l|S_l)\right|\right].$$

To get the first term of the sum, we recall the assumption (1) on the uniformly bound upon the reward function. Let, us take a closer look on the expectation term:

$$\mathbb{E}_{\pi_\theta}\left[\left|\log\frac{\pi_\theta^{(n-l)}}{\bar{\pi}}(A_l|S_l)\right|\right] = \int_S m^{(n-l)}(ds)\int_{\mathcal{A}}\pi^{(n-l)}(da|s)\left|\log\frac{\pi_\theta^{(n-l)}}{\bar{\pi}}(a|s)\right|$$

$$\leq \int_S m^{(n-l)}(ds)\int_{\mathcal{A}}\pi^{(n-l)}(da|s)\left(|\log\pi_\theta^{(n-l)}(a|s)| + |\log\bar{\pi}(a|s)|\right) \tag{15}$$

Recalling the new assumptions (3) regarding the baseline probability density function, it is established that for all $(a,s)\in\mathcal{A}\times\mathcal{S}$, the condition $\bar{\pi}(a,s)>\epsilon$ holds, in conjunction with the assumption that the action space is finite, $|\mathcal{A}|<\infty$. In the discrete case, we can derive an upper bound for our expression as follows:

$$\leq -\int_S m^{(n-l)}(ds)\int_{\mathcal{A}}\pi_\theta^{(n-l)}(a|s)\log\pi_\theta^{(n-l)}(a|s)da - \log\epsilon.$$

To bound the first term, we utilize the fact that $x\log x$ for $x\in(0,1)$ attains its minimal value at $x=e^{-1}$. Consequently, in the discrete scenario, we obtain:

$$\Rightarrow \mathbb{E}_{\pi_\theta}\left[\left|\log\frac{\pi_\theta^{(n-l)}}{\bar{\pi}}(A_l|S_l)\right|\right] \leq \frac{|\mathcal{A}|}{e} - \log\epsilon$$

In the continuous case, where the baseline policy is uniformly distributed over the action space, the first sum term under the integral in Equation 15 is similarly bounded by $\mu(A)/e$. To bound the second term, we express:

$$\int_{\mathcal{A}\times S} m^{(n-l)}(ds)\pi^{(n-l)}(da|s)\,|\log\bar{\pi}(a|s)| \leq \int_{\mathcal{A}\times S} m^{(n-l)}(ds)\bar{\pi}(da|s)\frac{e^{h_\theta(a,s)}}{\int_A\bar{\pi}(da'|s)e^{h_\theta(a',s)}}|\log(\bar{\pi}(a|s))|$$

$$\leq \mu(A)\int_{\mathcal{A}\times S} m^{(n-l)}(ds)\mathrm{softmax}(h_\theta)(a,s)|\bar{\pi}(a|s)\log(\bar{\pi}(a|s))| \leq \frac{\mu(A)^2}{e}$$

Thus, for the continuous case, the following bound is obtained:

$$\Rightarrow \mathbb{E}_{\pi_\theta}\left[\left|\log\frac{\pi_\theta^{(n-l)}}{\bar{\pi}}(A_l|S_l)\right|\right] \leq \frac{\mu(\mathcal{A})}{e}(1+\mu(A))$$

Ultimately, as this bound is independent of the step horizon, we can establish the overall bound:

$$\Rightarrow \textbf{Discrete: } \mathbb{E}_{\pi_\theta}[|C_n^\theta|] \leq (n-1)(U+\tau\frac{|\mathcal{A}|}{e} - \tau\log\epsilon).$$

$$\Rightarrow \textbf{Continuous: } \mathbb{E}_{\pi_\theta}[|C_n^\theta|] \leq (n-1)(U+\tau\frac{\mu(\mathcal{A})}{e}(1+\mu(A))).$$

This concludes the proof.

□

Lemma 6 (Lipschitz distribution law)**.** *Let $(s_0,a_0,\ldots,s_n)$ be the Agent's sampled path, then for all $l=1,\ldots n$ it holds that:*

$$\left|m_{\theta_2}^{(l)}(s_l) - m_{\theta_1}^{(l)}(s_l)\right| \leq nB\prod_{i=0}^{l} p\left(s_{n-i}a_{n-i}, s_{n-i-1}\right)\prod_{i=0}^{l}\pi_{\tilde{\theta}}^{(n-i)}\left(a_{n-i}\mid s_{n-i}\right)\|\theta_2-\theta_1\|,$$

where $\tilde{\theta}$ is such that, there exist $\lambda\in[0,1]$, $\tilde{\theta}=\lambda\theta_2+(1-\lambda)\theta_1$

Proof. Recall, the Markov property for the MDP and for $l \in [n-1]$), write:

$$\left| m_{\theta_2}^{(l)}(s_l) - m_{\theta_1}^{(l)}(s_l) \right|$$

$$= \left| \prod_{i=0}^{l} p\left(s_{n-i}a_{n-i}, s_{n-i-1}\right) \pi_{\theta_2}^{(n-i)}\left(a_{n-i} \mid s_{n-i}\right) - \prod_{i=0}^{l} p\left(s_{n-i}a_{n-i}, s_{n-i-1}\right) \pi_{\theta_1}^{(n-i)}\left(a_{n-i} \mid s_{n-i}\right) \right|$$

$$\leq \left| \prod_{i=0}^{l} p\left(s_{n-i}a_{n-i}, s_{n-i-1}\right) \right| \left| \prod_{i=0}^{l} \pi_{\theta_2}^{(n-i)}\left(a_{n-i} \mid s_{n-i}\right) - \prod_{i=0}^{l} \pi_{\theta_1}^{(n-i)}\left(a_{n-i} \mid s_{n-i}\right) \right|.$$

Then, the first term of the multiplication could be bound by 1, to bound the second term we refer to the derivation in [9] (page 23 (A.6)) and get following bound:

$$\left| \prod_{i=0}^{l} \pi_{\theta_2}^{(n-i)}\left(a_{n-i} \mid s_{n-i}\right) - \prod_{i=0}^{l} \pi_{\theta_1}^{(n-i)}\left(a_{n-i} \mid s_{n-i}\right) \right| \leq (l+1)B \prod_{i=0}^{l} \pi_{\tilde{\theta}}^{(n-i)}\left(a_{n-i} \mid s_{n-i}\right),$$

where, $\tilde{\theta}$ is vector lying between θ_2 and θ_1. For $l = n$, inequality still holds, since $m_{\theta}^{(n)}$ corresponds to the initial distribution and independent from parameters $\theta^{(i)}$:

$$\left| m_{\theta_2}^{(n)}(s_0) - m_{\theta_1}^{(n)}(s_0) \right| = 0,$$

with that we conclude the proof. □

Lemma 7 (Lipschitz smooth objective)**.** *For any fixed horizon $n \in \mathbb{N}$, the gradient of the objective function $J_n(\pi_\theta)$ is L-Lipschitz w.r.t the parameters θ:*

$$\left\| \nabla J_n\left(\pi_{\theta_2}\right) - \nabla J_n\left(\pi_{\theta_1}\right) \right\|_2 \leq L'' \left\| \theta_2 - \theta_1 \right\|_2,$$

where $L'' = B'L + \frac{n-1}{2}(nU + \tau(1 + e^{-1} + ne^{-1}))nB^2$

Proof. Write,

$$\left\| \nabla J_n\left(\pi_{\theta_2}\right) - \nabla J_n\left(\pi_{\theta_1}\right) \right\|_2 = \left\| \mathbb{E}_{\pi_{\theta_2}} \left[\sum_{i=0}^{n-1} C_{n-i}^{\theta_2} \nabla \log \pi_{\theta_2}^{(n-i)}\left(A_i \mid S_i\right) \right] - \mathbb{E}_{\pi_{\theta_1}} \left[\sum_{i=0}^{n-1} C_{n-i}^{\theta_1} \nabla \log \pi_{\theta_1}^{(n-i)}\left(A_i \mid S_i\right) \right] \right\|$$

$$\leq \sum_{i=0}^{n-1} \left\| \mathbb{E}_{\pi_{\theta_2}} \left[C_{n-i}^{\theta_2} \nabla \log \pi_{\theta_2}^{(n-i)}\left(A_i \mid S_i\right) \right] - \mathbb{E}_{\pi_{\theta_1}} \left[C_{n-i}^{\theta_1} \nabla \log \pi_{\theta_1}^{(n-i)}\left(A_i \mid S_i\right) \right] \right\|$$

Now, we will add and subtract additional terms,

$$= \sum_{i=0}^{n-1} \left\| \mathbb{E}_{\pi_{\theta_2}} \left[C_{n-i}^{\theta_2} \nabla \log \pi_{\theta_2}^{(n-i)}\left(A_i \mid S_i\right) \right] - \mathbb{E}_{\pi_{\theta_2}} \left[C_{n-i}^{\theta_2} \nabla \log \pi_{\theta_1}^{(n-i)}\left(A_i \mid S_i\right) \right] + \mathbb{E}_{\pi_{\theta_2}} \left[C_{n-i}^{\theta_2} \nabla \log \pi_{\theta_1}^{(n-i)}\left(A_i \mid S_i\right) \right] \right.$$

$$\left. - \mathbb{E}_{\pi_{\theta_2}} \left[C_{n-i}^{\theta_1} \nabla \log \pi_{\theta_1}^{(n-i)}\left(A_i \mid S_i\right) \right] + \mathbb{E}_{\pi_{\theta_2}} \left[C_{n-i}^{\theta_1} \nabla \log \pi_{\theta_1}^{(n-i)}\left(A_i \mid S_i\right) \right] - \mathbb{E}_{\pi_{\theta_1}} \left[C_{n-i}^{\theta_1} \nabla \log \pi_{\theta_1}^{(n-i)}\left(A_i \mid S_i\right) \right] \right\|$$

$$\leq \sum_{i=0}^{n-1} \left(\left\| \mathbb{E}_{\pi_{\theta_2}} \left[C_{n-i}^{\theta_2} \nabla \log \pi_{\theta_2}^{(n-i)}\left(A_i \mid S_i\right) \right] - \mathbb{E}_{\pi_{\theta_2}} \left[C_{n-i}^{\theta_2} \nabla \log \pi_{\theta_1}^{(n-i)}\left(A_i \mid S_i\right) \right] \right\| \right.$$

$$\left. + \left\| \mathbb{E}_{\pi_{\theta_2}} \left[C_{n-i}^{\theta_2} \nabla \log \pi_{\theta_1}^{(n-i)}\left(A_i \mid S_i\right) \right] - \mathbb{E}_{\pi_{\theta_1}} \left[C_{n-i}^{\theta_1} \nabla \log \pi_{\theta_1}^{(n-i)}\left(A_i \mid S_i\right) \right] \right\| \right).$$

Now, we will sequentially define an upper bound for these two terms, that appears in summation. Let us consider the first difference we want to bound:

$$I_1 := \|\mathbb{E}_{\pi_{\theta_2}}\left[C_{n-i}^{\theta_2}\nabla\log\pi_{\theta_2}^{(n-i)}\left(A_i \mid S_i\right)\right] - \mathbb{E}_{\pi_{\theta_2}}\left[C_{n-i}^{\theta_2}\nabla\log\pi_{\theta_1}^{(n-i)}\left(A_i \mid S_i\right)\right]\| \\ \leq \mathbb{E}_{\pi_{\theta_2}}\left[\left|C_{n-i}^{\theta_2}\right|\| \log\pi_{\theta_2}^{(n-i)}\left(A_i \mid S_i\right) - \nabla\log\pi_{\theta_1}^{(n-i)}\left(A_i \mid S_i\right)\|\right].$$

To derive the inequality above, we just used the fact that both expectation were driven under the same distribution, and then, we simply applied Jensen's inequality. Now, we recall the lemma (2) on the Lipschitz smooth log-policy and lemma (5) on the cumulative reward bound, and we get:

$$I_1 \leq \mathbb{E}_{\pi_{\theta_2}}\left[\left|C_{n-i}^{\theta_2}\right|\right] L\|\theta_2 - \theta_1\| \leq B'L\|\theta_2 - \theta_1\|$$

Next, we move to the second term and we apply similar strategy:

$$I_2 := \|\mathbb{E}_{\pi_{\theta_2}}\left[C_{n-i}^{\theta_2}\nabla\log\pi_{\theta_1}^{(n-i)}\left(A_i \mid S_i\right)\right] - \mathbb{E}_{\pi_{\theta_1}}\left[C_{n-i}^{\theta_1}\nabla\log\pi_{\theta_1}^{(n-i)}\left(A_i \mid S_i\right)\right]\|.$$

Recall, that $C_{n-i} = \sum_{l=i}^{n-1}(R_l - \tau\log\frac{\pi^{(n-l)}}{\bar{\pi}}\left(A_l \mid S_l\right))$, where $S_l \sim m^{(n-l)}(\cdot)$ and $A_l \sim \pi_\theta^{(n-l)}(\cdot|S_l)$. We can write:

$$I_2 \leq \sum_{l=i}^{n-1}\|\mathbb{E}_{\pi_{\theta_2}}[\nabla\log\pi_{\theta_1}^{(n-i)}\left(A_i \mid S_i\right)\left(R_l - \tau\log\frac{\pi_{\theta_2}^{(n-l)}}{\bar{\pi}}\left(A_l \mid S_l\right)\right] \\ -\mathbb{E}_{\pi_{\theta_1}}\left[\nabla\log\pi_{\theta_1}^{(n-i)}\left(A_i \mid S_i\right)\left(R_l - \tau\log\frac{\pi_{\theta_1}^{(n-l)}}{\bar{\pi}}\left(A_l \mid S_l\right)\right)\right]\|$$

$$\leq \sum_{l=i}^{n-1}\|\int\nabla\log\pi_{\theta_1}^{(n-i)}\left(a|s\right)r(a,s)\left(m_{\theta_2}^{(n-l)}(ds)\pi_{\theta_2}^{(n-l)}(da|s) - m_{\theta_1}^{(n-l)}(ds)\pi_{\theta_1}^{(n-l)}(da|s)\right)\|$$

$$+\|\tau\int\nabla\log\pi_{\theta_1}^{(n-i)}\left(a|s\right)\left(m_{\theta_1}^{(n-l)}(ds)\pi_{\theta_1}^{(n-l)}(da|s)\log\pi_{\theta_1}^{(n-l)}(a|s) - m_{\theta_2}^{(n-l)}(ds)\pi_{\theta_2}^{(n-l)}(da|s)\log\pi_{\theta_2}^{(n-l)}(a|s)\right)\|.$$

To obtain the inequality above, we simply opened brackets inside expectations and rearranged terms. Now, let us pass norm inside the integral to get:

$$\leq \sum_{l=i}^{n-1}\int\|\nabla\log\pi_{\theta_1}^{(n-i)}\left(a|s\right)\||r(a,s)|\left|m_{\theta_2}^{(n-l)}(ds)\pi_{\theta_2}^{(n-l)}(da|s) - m_{\theta_1}^{(n-l)}(ds)\pi_{\theta_1}^{(n-l)}(da|s)\right|$$

$$+\tau\int\|\nabla\log\pi_{\theta_1}^{(n-i)}\left(a|s\right)\|\left|m_{\theta_1}^{(n-l)}(ds)\pi_{\theta_1}^{(n-l)}(da|s)\log\pi_{\theta_1}^{(n-l)}(a|s) - m_{\theta_2}^{(n-l)}(ds)\pi_{\theta_2}^{(n-l)}(da|s)\log\pi_{\theta_2}^{(n-l)}(a|s)\right|.$$

Again, by the assumption on the reward function (1) and lemma (3), we write following bound:

$$\leq \sum_{l=i}^{n-1}UB\int\left|m_{\theta_2}^{(n-l)}(s)\pi_{\theta_2}^{(n-l)}(a|s) - m_{\theta_1}^{(n-l)}(s)\pi_{\theta_1}^{(n-l)}(a|s)\right|dads$$

$$+\tau B\int\left|m_{\theta_1}^{(n-l)}(s)\pi_{\theta_1}^{(n-l)}(a|s)\log\pi_{\theta_1}^{(n-l)}(a|s) - m_{\theta_2}^{(n-l)}(s)\pi_{\theta_2}^{(n-l)}(a|s)\log\pi_{\theta_2}^{(n-l)}(a|s)\right|. \tag{16}$$

By the lemma (6), the integral in the first term, we can bound as follow:

$$\int \left| m_{\theta_2}^{(n-l)}(s)\pi_{\theta_2}^{(n-l)}(a|s) - m_{\theta_1}^{(n-l)}(s)\pi_{\theta_1}^{(n-l)}(a|s) \right| dads \leq nB \left\|\theta_2 - \theta_1\right\| \int \prod_{i=0}^{n-l} p\left(\ldots\right) \pi_{\tilde{\theta}}^{(n-i)}\left(\ldots\right) dads$$

$$\leq nB \left\|\theta_2 - \theta_1\right\|$$

We left to bound the second term for the inequality (16). However, before we do it, we need some more small results, let us start by showing that for all $l \in [n]$, $\pi_\theta^{(n-l)}(a|s) \log \pi_\theta^{(n-l)}(a|s)$ is Lipschitz w.r.t θ. We show, that is has bounded gradient term, which is with mean value theorem would directly implies the Lipschitz condition. Therefore, we consider the gradient:

$$\|\nabla \left(\pi_\theta^{(n-l)}(a|s) \log \pi_\theta^{(n-l)}(a|s) \right) \| = \|\nabla \pi_\theta^{(n-l)}(a|s) \log \pi_\theta^{(n-l)}(a|s) + \pi_\theta^{(n-l)}(a|s) \nabla \log \pi_\theta^{(n-l)}(a|s)\|.$$

Recall, the policy gradient expression (12) and write:

$$= \|\pi_\theta^{(n-l)}(a|s) \log \pi_\theta^{(n-l)}(a|s) \int_A \left(\delta_{da',a} - \pi_\theta^{(n-l)}(da'|s) \right) \psi^{(n-l)}(a', s)/\tau + \nabla \pi_\theta^{(n-l)}(a|s)\|,$$

$$\leq \left| \pi_\theta^{(n-l)}(a|s) \log \pi_\theta^{(n-l)}(a|s) \right| \int_A \left| \delta_{da',a} - \pi_\theta^{(n-l)}(da'|s) \right| \|\psi^{(n-l)}(a', s)/\tau\| + \|\nabla \pi_\theta^{(n-l)}(a|s)\|,$$

We know, that $\left| \pi_\theta^{(n-l)}(a|s) \log \pi_\theta^{(n-l)}(a|s) \right| \leq \left| e^{-1} \log e^{-1} \right|$, then:

$$\leq \left| e^{-1} \log e^{-1} \right| \frac{2}{\tau} \max_{\substack{(a,s)\in A\times S \\ i\in[n]}} \left\| \psi^{(i)}(a,s) \right\| + \frac{2}{\tau} \max_{\substack{(a,s)\in A\times S \\ i\in[n]}} \left\| \psi^{(i)}(a,s) \right\|$$

$$= \frac{2}{\tau} \max_{\substack{(a,s)\in A\times S \\ i\in[n]}} \left\| \psi^{(i)}(a,s) \right\| (1 + e^{-1}) = B(1 + e^{-1})$$

Then, we get that $\pi_\theta^{(n-l)}(a|s) \log \pi_\theta^{(n-l)}(a|s)$ is Lipschitz, with a constant $L'' = B(1 + e^{-1})$. Back to the inequality 16, we focus on the second term, where by rearranging we can get the following bound:

$$\leq \tau B \int \left| m_{\theta_1}^{(n-l)}(s) \right| \left| \pi_{\theta_1}^{(n-l)}(a|s) \log \pi_{\theta_1}^{(n-l)}(a|s) - \pi_{\theta_2}^{(n-l)}(a|s) \log \pi_{\theta_2}^{(n-l)}(a|s) \right| dads$$

$$+\tau B \int \left| \pi_{\theta_2}^{(n-l)}(a|s) \log \pi_{\theta_2}^{(n-l)}(a|s) \right| \left| m_{\theta_2}^{(n-l)}(s) - m_{\theta_1}^{(n-l)}(s) \right| dads.$$

To bound the first term, we use the result, that we just have derived. For second one, we again recall lemma (6) and we get:

$$\leq \tau B^2 (1 + e^{-1}) \left\|\theta_2 - \theta_1\right\| \int \left| m_{\theta_1}^{(n-l)}(s) \right| dsda$$

$$+\tau n B^2 e^{-1} \left\|\theta_2 - \theta_1\right\| \int \prod_{i=0}^{n-l} p\left(\ldots\right) \pi_{\tilde{\theta}}^{(n-i)}\left(\ldots\right) dads \leq \tau B^2 (1 + e^{-1} + ne^{-1}) \left\|\theta_2 - \theta_1\right\|$$

Finally, we can return to the inequality (16), and write:

$$I_2 \leq \sum_{l=i}^{n-1}(nU + \tau(1 + e^{-1} + ne^{-1}))B^2\|\theta_2 - \theta_1\| = (n-i)(nU + \tau(1 + e^{-1} + ne^{-1}))B^2\|\theta_2 - \theta_1\|$$

With this, we are ready to establish the final inequality:

$$\|\nabla J_n(\pi_{\theta_2}) - \nabla J_n(\pi_{\theta_1})\|_2 \leq \sum_{i=0}^{n-1}(I_1 + I_2)$$

$$\leq (B'L + \frac{n-1}{2}(nU + \tau(1 + e^{-1} + ne^{-1}))nB^2)\|\theta_2 - \theta_1\|_2$$

□

Finally, we have reached the pont, at which with few simple argument we

Theorem 5 (Global policy convergence with shared parameters)**.** *With MPG as defined, assuming an ideal MPG update and a learning rate $0 < \eta < \eta_0$, policy π_t converges as $t \to \infty$ to a policy $\pi_\infty \in \mathcal{P}_n$, at least at the rate $o(1/t)$.*

Proof. Remark. To adapt a standard argument for gradient descent updates, we are going to abuse the notation and consider modified objective $J_n := -J_n$, only for this proof.
We consider the lemma 7 on L''-smooth objective, where we set $\theta_2 = \theta_{t+1}$ and $\theta_1 = \theta_t$. Then since objective is L'' - smooth, we can write:

$$J_n(\theta_{t+1}) \leq J_n(\theta_t) + \nabla J_n(\theta_t)(\theta_{t+1} - \theta_t) + \frac{L''}{2}\|\theta_{t+1} - \theta_t\|_2^2.$$

Plug-in, the ideal S-MPG update:

$$J_n(\theta_{t+1}) \leq J_n(\theta_t) - \eta\|\nabla J_n(\theta_t)\|_2^2 + \frac{L''\eta^2}{2}\|\nabla J_n(\theta_t)\|_2^2.$$

Where, for $\eta \leq \frac{1}{L''}$, we have $\frac{L''\eta^2}{2} \leq \frac{\eta}{2}$, and:

$$J_n(\theta_{t+1}) \leq J_n(\theta_t) - \frac{\eta}{2}\|\nabla J_n(\theta_t)\|_2^2.$$

Rearranging terms, give us:

$$\|\nabla J_n(\theta_t)\|_2^2 \leq \frac{2}{\eta}\left(J_n(\theta_t) - J_n(\theta_{t+1})\right). \tag{17}$$

We note, that modified objective is monotone decreasing function w.r.t θ_t, $t \in \mathbb{N}$. And, we expect it's minimal value to be $J_n(\pi_*)$, that implies:

$$(J_n(\theta_t) - J_n(\theta_{t+1})) \leq (J_n(\theta_t) - J_n(\pi_*)) \qquad \text{for all } t \in \mathbb{N}.$$

Let us consider the average squared gradient norm, until time $t \in \mathbb{N}$, $\frac{1}{t}\sum_{u=0}^{t}\|\nabla J_n(\theta_u)\|_2^2$. Using the inequality (17), we can bound it by the telescoping sum, as follow:

$$\frac{1}{t}\sum_{u=0}^{t}\|\nabla J_n(\theta_u)\|_2^2 \leq \frac{1}{t}\left(\frac{2}{\eta}(J_n(\theta_0) - J_n(\theta_1)) + \frac{2}{\eta}(J_n(\theta_1) - J_n(\theta_2))\ldots + \frac{2}{\eta}(J_n(\theta_{t-1}) - J_n(\theta_t))\right)$$

By cancelling out similar terms and using the fact that minimum at $J_n(\pi_*)$, we write:

$$\frac{1}{t}\sum_{u=0}^{t}\|\nabla J_n(\theta_u)\|_2^2 \leq \frac{2\left(J_n(\theta_0)-J_n(\pi_*)\right)}{\eta\cdot t}$$

We notice, that average norm is decreasing function w.r.t $t\in\mathbb{N}$, therefore:

$$\|\nabla J_n(\theta_{t+1})\|_2^2 \leq \frac{1}{t}\sum_{u=0}^{t}\|\nabla J_n(\theta_u)\|_2^2 \leq \frac{2\left(J_n(\theta_0)-J_n(\pi_*)\right)}{\eta\cdot t}.$$

$\Rightarrow \nabla J_n(\theta_t)\to 0$ at the rate $o(\frac{1}{\eta\cdot t})$. With that we conclude the proof. □

2.3 Light MPG

Before delving into the discussion of optimal and unique solution theorems, specifically the equivalence of Theorem 3 and 4 as referenced in the original work [3], it is pertinent to introduce a novel objective function along with its corresponding gradient update algorithm, termed "Light-MPG". The primary impetus behind this new algorithm is to refine the training procedure, aiming for increased efficiency while maintaining a resemblance to the original objective J_n. Moreover, the paramount rationale lies in the complexity associated with handling the initial objective function when utilizing the pipeline and toolkit established in the original work to conduct the proofs. Consequently, proving solution optimality becomes an intricate task, necessitating a fresh approach, possibly incorporating a new set of tools and concepts. The concluding chapter of this section (2.7) is dedicated to outlining potential strategies and a roadmap to tackle this specific issue.

Conversely, the application of the alternative Light-MPG update facilitates the derivation of proofs for optimality and uniqueness, which, though not straightforward, are characterized by their suitability and elegance. We commence our discussion by examining the Light-MPG update in detail:

Light-MPG

In the S-MPG framework, for each horizon level i, there is a requirement to update the entire set of parameters, denoted as $\theta^{(i)}=\bigcup_{j=1}^{i}\vartheta^{(j)}$. In contrast, the proposed approach, akin to the original MPG update, necessitates updating only the parameters corresponding to the specific horizon step, $\vartheta^{(i)}$. This strategy intuitively allows the parameters $\vartheta^{(i)}$ to solely influence the behavior of the policy $\pi^{(i)}(\cdot\mid\cdot)$. The update mechanism is nearly identical to that of MPG, with the primary distinction being the substitution of $\theta^{(i)}$ with $\vartheta^{(i)}$:

$$\vartheta_{t+1}^{(i)}=\vartheta_t^{(i)}+\eta\sum_{\ell=n-i}^{n-1}\left(R_\ell-\tau\log\frac{\pi_t^{(n-\ell)}}{\bar{\pi}}\left(A_\ell\mid S_\ell\right)\right)\nabla_{\vartheta^{(i)}}\log\pi^{(i)}_{\theta_{t+1}^{(i-1)}\bigcup\vartheta_t^{(i)}}\left(A_{n-i}\mid S_{n-i}\right), \tag{18}$$

It is important to note that prior to proceeding with the update for the next horizon level, the parameters of the shorter horizons are updated without accumulating the update. To signify this process, we use the notation $\pi^{(i)}_{\theta_{t+1}^{(i-1)}\bigcup\vartheta_t^{(i)}}$. We hypothesize that this approach will contribute to enhanced stability of the algorithm.

Let us now define the objective for the Light-MPG update. We start by defying, what we call the "One step horizon policy objective", $J_n^{(i)}$.

Algorithm 3 Light-Shared Matryoshka Policy Gradient (Light-MPG) Update

Input: Learning rate η, temperature τ, initial policy parameters θ_0, fixed horizon n
Initialize policy parameters $\theta = \theta_0$
for each training timestep $t = 0, 1, 2, \dots$ **do**
  Sample initial state $S_0 \sim \nu_0$
  for $i = 1$ to n **do**
    Sample action A_{n-i} according to $\pi_t^{(i)}(\cdot \mid S_{n-i})$
    Collect reward $R_{n-i} \sim p_{\text{rew}}(\cdot \mid S_{n-i}, A_{n-i})$
    Move to next state $S_{n-i+1} \sim p(S_{n-i}, A_{n-i}, \cdot)$
  end for
  for $i = 1$ to n **do**
    Compute cumulative reward and entropy term:
    $C_i = \sum_{\ell=n-i}^{n-1} \left(R_\ell - \tau \log \frac{\pi_t^{(n-\ell)}}{\bar{\pi}}(A_\ell \mid S_\ell) \right)$
    Update policy parameters:
    $\vartheta_{t+1}^{(i)} = \vartheta_t^{(i)} + \eta C_i \nabla_{\vartheta^{(i)}} \log \pi^{(i)}_{\theta_{t+1}^{(i-1)} \bigcup \vartheta_t^{(i)}}(A_{n-i} \mid S_{n-i})$
  end for
end for
Output: Optimized policy parameters θ

Definition 3 (One step horizon policy objective $J_n^{(i)}$)**.** *For $n \in \mathbb{N}$ being the fixed horizon. For $i \in [n]$ and given the policy $\pi' = (\pi'^{(1)}, \dots \pi'^{(i-1)}, \pi'^{(i+1)}, \dots, \pi'^{(n)})$, we define 1-step horizon policy, as follow:*

$$J_n^{(i)}(\pi^{(i)} \mid \pi') := \int_{\mathcal{S}} m_{\pi'}^{(i)}(ds) \left(\mathbb{E}_{\pi^{(i)}} \left[Q_{\pi'}^{(i)}(A|s) \right] - D_{\text{KL}} \left(\pi^{(i)} \| \bar{\pi} \right)(s') \right)$$

We note, that if we want to maximize one step horizon policy objective for the given distribution π', we have to consider the state distribution $m_{\pi'}$ and the action value $Q_{\pi'}$ as independent functions from the target distribution $\pi^{(i)}$, for which we wish to maximize the objective function $J_n^{(i)}$.

In particular, for our case of the Light-MPG we will consider $\pi^{(i)}$ to be $\pi^{(i)}_{\vartheta^{(i)}}$ (we use notation abuse, to note $\pi^{(i)}_{\vartheta^{(i)}} = \pi^{(i)}_{\theta_{t+1}^{(i-1)} \bigcup \vartheta_t^{(i)}}$) and π' to be $(\pi^{(1)}_{\vartheta^{(1)}}, \dots \pi^{(i-1)}_{\vartheta^{(i-1)}}, \pi^{(i+1)}_{\vartheta^{(i+1)}}, \dots, \pi^{(n)}_{\vartheta^{(n)}})$. Then we can implicitly define the Light MPG objective, as follow

Definition 4 (Light objective)**.** *Let ϑ be the initial state distribution $\sim S_0$ and $n \in \mathbb{N}$ be the maximal horizon. Then, we say that the Light Objective is an objective that for $i \in [n]$ satisfies:*

$$\nabla_{\vartheta^{(i)}} J_n^{Light}(\pi_\theta) = \nabla_{\vartheta^{(i)}} J_n^{(i)}(\pi^{(i)}_{\vartheta^{(i)}} \mid \pi^{(1)}_{\vartheta^{(1)}}, \dots \pi^{(i-1)}_{\vartheta^{(i-1)}}, \pi^{(i+1)}_{\vartheta^{(i+1)}}, \dots, \pi^{(n)}_{\vartheta^{(n)}}).$$

One can easily check, that with Light objective our Light MPG update (2.3) for the step horizon i can be rewritten as follow:

$$\vartheta_{t+1}^{(i)} = \vartheta_t^{(i)} + \eta \nabla_{\vartheta^{(i)}} J_n^{(i)}(\pi^{(i)}_{\vartheta_t^{(i)}} \mid \pi^{(1)}_{\vartheta_{t+1}^{(1)}}, \dots \pi^{(i-1)}_{\vartheta_{t+1}^{(i-1)}}, \pi^{(i+1)}_{\vartheta_t^{(i+1)}}, \dots, \pi^{(n)}_{\vartheta_t^{(n)}}) := \vartheta_t^{(i)} + \eta \nabla_{\vartheta^{(i)}} J_n^{Light}(\pi_{\theta_{t+1}})$$

Again, we use the notation abuse and when we say $\pi_{\theta_{t+1}}$, one should always refer to the underlying context. In particular, in a case of the Light MPG update, depending on the step horizon the parameters set θ_{t+1} is different. For example, for the step horizon i we have: $\theta_{t+1} :=$

$\theta_{t+1}^{(i-1)} \bigcup \left(\bigcup_{j=i}^{n} \vartheta_t^{(j)} \right)$.

Now, we see the purpose of the Light MPG update, which is at each tainting iteration is to optimize one step policy objective, for each horizon step given the current distribution. Finally, similarly to the S-MPG we can establish the following proposition.

Proposition 5. *For the Light-MPG update defined in (2.3), it holds that for all $i \in [n]$, $\mathbb{E}[\vartheta_{t+1}^{(i)} - \vartheta_t^{(i)}] = \eta \nabla_{\vartheta^{(i)}} J_n^{Light}$*

Proof. By construction. □

In the subsequent sections, namely Sections 2.4 and 2.5, we aim to establish the proofs for the uniqueness and optimality of the policy resulting from the Light-MPG algorithm. It is pivotal to mention that, unlike S-MPG, we lack a convergence result for Light-MPG. Consequently, our approach assumes to be at critical point, although in practice, convergence is not guaranteed. Subsequently, in Section 2.6, we will address the issue of the incomplete theoretical frameworks pertaining to both the S-MPG and Light-MPG algorithms.

2.4 Optimal solution with Realizability assumption

Remark. For simplicity we are going to abuse the notations and write π_t instead of π_{θ_t}.

Lemma 8 (Light-MPG Log increment)**.** *Let $i \in \mathbb{N}$ and $m \in \{1, \dots, n\}$. Suppose that $\pi_t^{(k)} = \pi_*^{(k)}$ for all $k = 1, \dots, m-1$. Then, it holds that:*

$$\begin{aligned}
&\log \pi_{t+1}^{(m)}(a \mid s) - \log \pi_t^{(m)}(a \mid s) \\
&\quad = -\frac{\eta}{\tau} \int_{\mathcal{S}} \mathbf{m}_{\pi_t}^{(m)} \left(\mathrm{d}s' \right) \int_{\mathcal{A}} \pi_t^{(m)} \left(\mathrm{d}a' \mid s' \right) \left(\log \frac{\pi_t^{(m)}}{\pi_*^{(m)}} \left(a' \mid s' \right) - D_{\mathrm{KL}} \left(\pi_t^{(m)} \| \pi_*^{(m)} \right) \left(s' \right) \right) \\
&\qquad \times \left(\Theta^{(m)} \left((a, s), (a', s') \right) - \mathbb{E}_{\pi_t^{(m)}} \left[\Theta^{(m)} \left((A, s), (a', s') \right) \right] \right) + o\left(\eta C\left(\theta_t \right) \right),
\end{aligned}$$

where $\mathbf{m}_{\pi_t}^{(m)}$ is the law of S_{n-m}.

Proof. Recall the parametrization and gradient of the policy:

$$h^{(m)}(a|s) = h^{(m-1)}(a|s) + \vartheta^{(m)} \phi^{(m)}(a|s) = \theta^{(m-1)} \psi^{(m-1)} + \vartheta^{(m)} \phi^{(m)}(a|s),$$

$$\text{where} \quad \theta^{(m)} = \theta^{(1)} \bigcup \left(\bigcup_{i=2}^{m} \theta^{(i)} \right).$$

Since, the parametrization is linear with respect to the preference function $h^{(m)}$, we can write that:

$$\nabla_\theta h^{(m)}(a|s) = \begin{bmatrix} \nabla_{\theta^{(m-1)}} h^{(m)}(a|s) \\ \nabla_{\vartheta^{(m)}} h^{(m)}(a|s) \\ 0 \end{bmatrix},$$

then, the log gradient policy can be read as:

$$\nabla_\theta \log \pi_t^{(m)}(a|s) = \frac{1}{\tau} \int_{\mathcal{A}} \left(\delta_{a,\ \mathrm{d}a'} - \pi_t^{(m)} \left(\mathrm{d}a' \mid s \right) \right) \begin{bmatrix} \nabla_{\theta^{(m-1)}} h^{(m)}(a|s) \\ \nabla_{\vartheta^{(m)}} h^{(m)}(a|s) \\ 0 \end{bmatrix}.$$

Now, consider the log policy increment, with respect to the full gradient update (all horzons were updated). Using a first-order Taylor approximation, we get:

$$\log \frac{\pi_{t+1}^{(m)}(a|s)}{\pi_t^{(m)}(a|s)} = (\theta_{t+1}^{(m)} - \theta_t^{(m)}) \cdot \nabla_\theta \log \pi_t^{(m)}(a|s) + o(\eta C(\theta_t))$$

$$= \left(\left[\theta_{t+1}^{(m-1)} - \theta_t^{(m-1)}, \vartheta_{t+1}^{(m)} - \vartheta_t^{m}, \dots \right] \right) \cdot \frac{1}{\tau} \int_{\mathcal{A}} \left(\delta_{a,\ \mathrm{d}a'} - \pi_t^{(m)} \left(\mathrm{d}a' \mid s \right) \right) \begin{bmatrix} \nabla_{\theta^{(m-1)}} h^{(m)}(a \mid s) \\ \nabla_{\vartheta^{(m)}} h^{(m)}(a \mid s) \\ 0 \end{bmatrix} + o\left(\eta C\left(\theta_t\right)\right)$$

$$= \left(\theta_{t+1}^{(m-1)} - \theta_t^{(m-1)} \right) \cdot \frac{1}{\tau} \int_{\mathcal{A}} \left(\delta_{a,\ \mathrm{d}a'} - \pi_t^{(m)} \left(\mathrm{d}a' \mid s \right) \nabla_{\theta^{(m-1)}} h^{(m)}(a \mid s) \right.$$
$$\left. + \left(\vartheta_{t+1}^{(m)} - \vartheta_t^{m} \right) \cdot \frac{1}{\tau} \int_{\mathcal{A}} \left(\delta_{a,\ \mathrm{d}a'} - \pi_t^{(m)} \left(\mathrm{d}a' \mid s \right) \right) \nabla_{\vartheta^{(m)}} h^{(m)}(a \mid s) + o\left(\eta C\left(\theta_t\right)\right) \right.$$

The first term of the sum, can be rewritten as follow:

$$\left(\theta_{t+1}^{(m-1)} - \theta_t^{(m-1)} \right) \cdot \frac{1}{\tau} \int_{\mathcal{A}} \left(\delta_{a,\ \mathrm{d}a'} - \pi_t^{(m)} \left(\mathrm{d}a' \mid s \right) \nabla_{\theta^{(m-1)}} h^{(m)}(a \mid s) \right. = \left(\theta_{t+1}^{(m-1)} - \theta_t^{(m-1)} \right) \cdot \nabla_\theta \log \pi_t^{(m-1)}(a|s),$$

which is exactly, the log policy increment of the previous step horizon:

$$\left(\theta_{t+1}^{(m-1)} - \theta_t^{(m-1)} \right) \cdot \nabla_\theta \log \pi_t^{(m-1)}(a|s) = \log \frac{\pi_{t+1}^{(m-1)}(a|s)}{\pi_t^{(m-1)}(a|s)} + o\left(\eta C\left(\theta_t\right)\right) \simeq 0,$$

the log increment is zero, due to the assumption on the shorter-horizon policies. Hence, we left with second term of the sum. Recalling the ideal light-MGP update , we obtain:

$$\log \frac{\pi_{t+1}^{(m)}(a|s)}{\pi_t^{(m)}(a|s)} = \left(\vartheta_{t+1}^{(m)} - \vartheta_t^{m} \right) \cdot \frac{1}{\tau} \int_{\mathcal{A}} \left(\delta_{a,\ \mathrm{d}a'} - \pi_t^{(m)} \left(\mathrm{d}a' \mid s \right) \right) \nabla_{\vartheta^{(m)}} h^{(m)}(a \mid s) + o\left(\eta C\left(\theta_t\right)\right)$$

$$= \eta \mathbb{E}_{\pi_t}[C_m \nabla_{\vartheta^{(m)}} \log \pi_t^{(m)}(A_{n-m} \mid S_{n-m})] \cdot \frac{1}{\tau} \int_{\mathcal{A}} \left(\delta_{a,\ \mathrm{d}a'} - \pi_t^{(m)} \left(\mathrm{d}a' \mid s \right) \right) \nabla_{\vartheta^{(m)}} h^{(m)}(a \mid s) + o\left(\eta C\left(\theta_t\right)\right)$$

$$= \eta \frac{1}{\tau^2} \mathbb{E}_{\pi_t}[C_m \int_{\mathcal{A}} \left(\delta_{a,\ \mathrm{d}a'} - \pi_t^{(m)} \left(\mathrm{d}a' \mid s \right) \right) \int_{\mathcal{A}} \left(\delta_{A_{n-m},\ \mathrm{d}a''} - \pi_t^{(m)} \left(\mathrm{d}a'' \mid S_{n-m} \right) \right) \Theta^{(m)}((a', s), (a'', S_{n-m}))] + o\left(\eta C\left(\theta_t\right)\right),$$

where $\Theta^{(m)}$ is the kernel w.r.t parameters $\vartheta^{(m)}$, thus $\Theta^{(m)} : ((a, s), (a', s')) \mapsto \phi^{(m)}(a, s) \cdot \phi^{(m)}(a', s')$.
By multiplying the expressions under integrals, we get:

$$= \eta \frac{1}{\tau^2} \mathbb{E}_{\pi_t}[C_m (\Theta^{(m)}((a, s), (A_{n-m}, S_{n-m})$$
$$- \int_{\mathcal{A}} \pi_t^{(m)} \left(\mathrm{d}a'' \mid S_{n-m} \right) \Theta^{(m)}((a, s), (a'', S_{n-m}) - \int_{\mathcal{A}} \pi_t^{(m)} \left(\mathrm{d}a' \mid s \right) \Theta^{(m)}((a', s), (A_{n-m}, S_{n-m})$$
$$+ \int_{\mathcal{A}^2} \pi_t^{(m)} \left(\mathrm{d}a' \mid s \right) \pi_t^{(m)} \left(\mathrm{d}a'' \mid S_{n-m} \right) \Theta^{(m)}((a', s), (a'', S_{n-m})))] + o\left(\eta C\left(\theta_t\right)\right),$$

that, can be simply rewritten as follow:

$$\eta \frac{1}{\tau^2} \mathbb{E}_{\pi_t} \left[C_m \left(\Theta^{(m)}\left((a, s), (A_{n-m}, S_{n-m})\right) \right. - \mathbb{E}_A \left[\Theta^{(m)}\left((A, s), (A_{n-m}, S_{n-m})\right) \right] \right.$$
$$\left. - \mathbb{E}_{A'} \left[\Theta^{(m)}\left((a, s), \left(A', S_{n-m}\right)\right) \right] + \mathbb{E}_{A, A'} \left[\Theta^{(m)}\left((A, s), \left(A', S_{n-m}\right)\right) \right] + o\left(\eta C\left(\theta_t\right)\right),$$

where, we note that A and A' are independent RVs with laws $\pi_t^{(m)}(\cdot \mid s)$ and $\pi_t^{(m)}(\cdot \mid S_{n-m})$. Therefore, we use a simple property of independent RVs, $\mathbb{E}[X(Y - \mathbb{E}[Y])] = \mathbb{E}(X - \mathbb{E}[X])Y]$, with $X = C_m$ and $Y = \Theta^{(m)} - \mathbb{E}_A[\Theta^{(m)}]$ we obtain:

$$\eta \frac{1}{\tau^2} \mathbb{E}_{\pi_t} \Big[(C_m - \mathbb{E}\left[C_m \mid S_{n-m}\right]) \Big(\Theta^{(m)}\left((a,s),\ (A_{n-m}, S_{n-m})\right. \\ \left. \mathbb{E}_A \left[\Theta^{(m)}\left((A,s),(A_{n-m},S_{n-m})\right) \right] \right). \tag{19}$$

Remark, the RV A' is conditionally obtained under S_{n-m} state. Thus, we get conditional expectation for our property. We write

$$\begin{aligned} \mathbb{E}\left[C_m \mid S_{n-m}\right] &= \mathbb{E}\left[\sum_{\ell=n-m}^{n} \left(R_\ell - \tau \log \frac{\pi_t^{(n-\ell)}}{\bar{\pi}}\left(A_\ell \mid S_\ell\right) \right) \mid S_{n-m} \right] \\ &= V_{\pi_t}^{(m)}\left(S_{n-m}\right) \\ &= V_*^{(m)}\left(S_{n-m}\right) - \tau D_{\mathrm{KL}}\left(\pi_t^{(m)} \| \pi_*^{(m)} \right)\left(S_{n-m}\right), \end{aligned}$$

where we used Lemma 1 and the fact that $\pi_t^{(i)} = \pi_*^{(i)}$ for all $i = 1, \ldots, m-1$. Similarly, we rewrite the expression for C_n:

$$\begin{aligned} C_m &= R_{n-m} - \tau \log \frac{\pi_t^{(m)}}{\bar{\pi}}\left(A_{n-m} \mid S_{n-m}\right) + \mathbb{E}\left[V_{\pi_t}^{(m-1)}\left(S_{n-m}\right) \mid S_{n-m}, A_{n-m} \right] \\ &= \mathbb{E}\left[V_{\pi_t}^{(m-1)}\left(S_{n-m+1}\right) - V_*^{(m-1)}\left(S_{n-m+1}\right) \mid S_{n-m}, A_{n-m} \right] \\ &\quad - \tau \log \frac{\pi_t^{(m)}}{\pi_*^{(m)}}\left(A_{n-m} \mid S_{n-m}\right) + V_*^{(m)}\left(S_{n-m}\right) \\ &= - \tau \log \frac{\pi_t^{(m)}}{\pi_*^{(m)}}\left(A_{n-m} \mid S_{n-m}\right) + V_*^{(m)}\left(S_{n-m}\right). \end{aligned}$$

Finally, we get desired equation:

$$\frac{\eta}{\tau} \mathbb{E}_{\pi_t} \Big[(D_{\mathrm{KL}}(\pi_t^{(m)} || \pi_*^{(m)})(A_{n-m}|S_{n-m}) - \log \tfrac{\pi_t^{(m)}}{\pi_*^{(m)}}(A_{n-m}|S_{n-m})) \left(\Theta^{(m)}\left((a,s),(A_{n-m},S_{n-m})\right) \right. \\ \left. - \mathbb{E}_A \left[\Theta^{(m)}\left((A,s),(A_{n-m},S_{n-m})\right) \right] \right) \Big] + o\left(\eta C\left(\theta_t\right)\right)$$

□

Lemma 9. *Let $\mathcal{H}_1, \mathcal{H}_2$ be two RKHSs on $\mathcal{A} \times \mathcal{S}$,*

- *The intersection $\mathcal{H}_1 \cap \mathcal{H}_2$ is an RKHS.*
- *For any element $f \in \mathcal{H}_1$, there exists a unique decomposition $f = g_\bullet + g_\perp$ such that $g_\bullet \in \mathcal{H}_1 \cap \mathcal{H}_2$ and $g_\perp \in \mathcal{H}_1 \cap (\mathcal{H}_2)^\perp$.*

We note $\mathcal{H}^\perp$ for the orthogonal complement of the $\mathcal{H}$.

Proof. See section "Reproducible kernel Hilbert spaces" [3], pages 16-18. □

Definition 5 ($I_K(f;\mu,\pi)$)**.** *For a probability measure of the form $\mu(\mathrm{d}s)\pi(\mathrm{d}a \mid s)$ on $\mathcal{A}\times\mathcal{S}$, where π is a policy, and for a positive-semidefinite kernel K on $\mathcal{A}\times\mathcal{S}$, we define the integral operator $I_K(\mu,\pi)$:*

$$I_K(f;\mu,\pi) : (a,s) \mapsto \int_{\mathcal{A}\times\mathcal{S}} \mu\left(\mathrm{d}s'\right) \pi\left(\mathrm{d}a' \mid s'\right) f\left(a',s'\right) K\left((a,s),\left(a',s'\right)\right)$$

Lemma 10. *Let $f \in L^2(\mu(\mathrm{d}s)\pi(\mathrm{d}a \mid s))$. It holds that $I_K(f;\mu,\pi)(a,s) = 0$ for all $a \in \mathcal{A}, s \in \mathcal{S}$ if and only if $f \in (\mathcal{H}_K)^{\perp}$.*

Proof. See section "Reproducible kernel Hilbert spaces" [3], pages 16-18. □

Definition 6 ($\widetilde{K}$)**.** *For any p.s.d kernel K on $\mathcal{A}\times\mathcal{S}$, we define:*

$$\widetilde{K}\left((a,s),\left(a',s'\right)\right) := K\left((a,s),\left(a',s'\right)\right) - \int_{\mathcal{A}} K\left((b,s),\left(a',s'\right)\right)\pi(\mathrm{d}b \mid s)$$
$$- \int_{\mathcal{A}} K\left((a,s),\left(b',s'\right)\right)\pi\left(\mathrm{d}b' \mid s'\right) + \int_{\mathcal{A}^2} K\left((b,s),\left(b',s'\right)\right)\pi(\mathrm{d}b \mid s)\pi\left(\mathrm{d}b' \mid s'\right)$$

Lemma 11. *It holds that, for p.s.d kernel K, the $\widetilde{K}$ is also positive-semidefinite kernel. Furthermore, any map $g \in \mathcal{H}_K \cap \left(\mathcal{H}_{\widetilde{K}}\right)^{\perp}$ is such that for every $s \in \mathcal{S}$, the map $a \mapsto g(a,s)$ is constant.*

Proof. See section "Reproducible kernel Hilbert spaces" [3], pages 16-18.

□

Proposition 6. *Let $i \in [n]$, with n being the fixed horizon. For all $(a,s) \in \mathcal{A}\times\mathcal{S}$, holds:*

$$\log \frac{\pi^{(i)}_{\theta_t}}{\pi^{(i)}_{*}}(a \mid s) - D_{\mathrm{KL}}\left(\pi^{(i)}_{\theta_t} \| \pi^{(i)}_{*}\right) = h^{(i)}_{\theta_t}(a,s) - Q^{(i)}_{*}(a,s) - \mathbb{E}_{\pi^{(i)}_{\theta_t}}\left[h^{(i)}_{\theta_t}(A,s) - Q^{(i)}_{*}(A,s)\right]$$

Proof.

$$\log \frac{\pi^{(i)}_{\theta_t}}{\pi^{(i)}_{*}}(a \mid s) - D_{\mathrm{KL}}\left(\pi^{(i)}_{\theta_t} \| \pi^{(i)}_{*}\right) = \log\left(\frac{e^{h^{(i)}_{\theta_t}}}{e^{Q^{(i)}_{*}}}\frac{e^{\mathbb{E}[e^{Q^{(i)}_{*}}]}}{e^{\mathbb{E}[h^{(i)}_{\theta_t}]}}\right) - \int_{\mathcal{A}} \pi^{(i)}_{\theta_t}(da|s) \log\left(\frac{e^{h^{(i)}_{\theta_t}}}{e^{Q^{(i)}_{*}}}\frac{e^{\mathbb{E}[e^{Q^{(i)}_{*}}]}}{e^{\mathbb{E}[h^{(i)}_{\theta_t}]}}\right).$$

Since, exponential with expectations terms are independed of a we cam take them out from integral, and then they will be cancelled out:

$$= \log\left(e^{h^{(i)}_{\theta_t} - Q^{(i)}_{*}}\right) + \left(\log\frac{e^{\mathbb{E}[e^{Q^{(i)}_{*}}]}}{e^{\mathbb{E}[h^{(i)}_{\theta_t}]}} - \log\frac{e^{\mathbb{E}[e^{Q^{(i)}_{*}}]}}{e^{\mathbb{E}[h^{(i)}_{\theta_t}]}}\right) - \int_{\mathcal{A}} \pi^{(i)}_{\theta_t}(da|s) \log\left(e^{h^{(i)}_{\theta_t} - Q^{(i)}_{*}}\right).$$

Finally, by taking the logarithm, we get desired result.

□

This concise proposition plays a pivotal role in establishing the connection between the parameterized policy and its corresponding preference function. This connection is fundamental to the proof of solution optimality in Theorem (6). Let us now turn our attention to the final lemma required for proving the theorem. That shows, that all constant in a maps, are within $(\mathcal{H}_{\tilde{K}})^{\perp}$, for any K p.s.d kernel

Lemma 12. *If $f \in L^2(\mu(\mathrm{d}s)\pi(\mathrm{d}a \mid s))$, and f is constant for action variable $a \Rightarrow f \in (\mathcal{H}_{\tilde{K}})^{\perp}$*

Proof. By lemma 10, we simply have to show:

$$I_{\tilde{K}}(f;\mu,\pi)(a,s) = 0 \qquad \text{for all } (a,s) \in \mathcal{A}\times\mathcal{S}.$$

Then, since f is independed of a factor, we can write

$$I_{\tilde{K}}(f;\mu,\pi)(a,s) = \int_{\mathcal{A}\times\mathcal{S}} \mu\left(\mathrm{d}s'\right)\pi\left(\mathrm{d}a' \mid s'\right) f\left(s'\right)\tilde{K}\left((a,s),\left(a',s'\right)\right)$$

$$= \int_{\mathcal{S}} \mu\left(\mathrm{d}s'\right) f\left(s'\right)\left[\int_{\mathcal{A}} \pi\left(\mathrm{d}a' \mid s'\right)\tilde{K}\left((a,s),\left(a',s'\right)\right)\right],$$

recall $\tilde{K}$ definition 6

$$= \int_{\mathcal{S}} \mu\left(\mathrm{d}s'\right) f\left(s'\right)\mathbb{E}_{A'}\left[K\left((a,s),\left(A',s'\right)\right) - \mathbb{E}_{A}\left[K\left((A,s),\left(a',s'\right)\right)\right]\right.$$
$$\left. - \int_{\mathcal{S}} \mu\left(\mathrm{d}s'\right) f\left(s'\right)\mathbb{E}_{A'}\left[K\left((a,s),\left(A',s'\right)\right)\right] + \mathbb{E}_{A,A'}\left[K\left((A,s),\left(A',s'\right)\right)\right].\right.$$

We notice, that all expectation terms cancel each other $\Rightarrow I_{\tilde{K}}(f;\mu,\pi)(a,s) = 0$ for all $(a,s) \in \mathcal{A}\times\mathcal{S}$. With that, we proof the lemma.

□

Remark. Prior to presenting the proof of solution optimality, it is imperative to address an alternate definition for the Realizability Assumption (2). Conventionally, we assert the existence of a parameter set $\theta_* \in \mathbb{R}^P$ such that $\pi_{\theta^*} = \pi_*$. However, this can equivalently be formulated as the existence of a mapping $C_i : \mathcal{S} \to \mathbb{R}$, ensuring that the expression $(Q_*^{(i)}(a,s) - Q_*^{(i-1)}(a,s) + C_i(s))$ belongs to the Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}_{\Theta^{(i)}}$, where $\Theta^{(i)} : ((a,s),(a',s')) \mapsto \phi^{(i)}(a,s)\cdot\phi^{(i)}(a',s')$. Given that the map C_i is independent of the action variable a, it does not alter the outcome of the policy. This implies that once a solution is found within our RKHS, an infinite number of solutions can be derived by adjusting the preference function by a factor that is independent of the action variable.

Theorem 6 (Convergence to the optimal in the Realizable Case and with shared parameters)**.** *Training with the ideal Light-MPG update under the Realizability assumption (2.4). If Light MPG converge $\pi_t \to \pi_\infty$, then the critical point is an optimal policy $\pi_\infty = \pi_*$.*

Proof. Let θ_t be a critical point of the objective $J_n(\theta)$, then by lemma (8) on the log increment, for all $(a,s) \in \mathcal{A}\times\mathcal{S}$ is holds:

$$0 = \log\pi_{t+1}^{(1)}(a \mid s) - \log\pi_t^{(1)}(a \mid s)$$
$$= -\frac{\eta}{\tau}\int_{\mathcal{S}} \mathbf{m}_{\pi_t}^{(1)}\left(\mathrm{d}s'\right)\int_{\mathcal{A}} \pi_t^{(1)}\left(\mathrm{d}a' \mid s'\right)\left(\log\frac{\pi_t^{(1)}}{\pi_*^{(1)}}\left(a' \mid s'\right) - D_{\mathrm{KL}}\left(\pi_t^{(1)}\|\pi_*^{(1)}\right)\left(s'\right)\right)$$
$$\times\left(\Theta^{(1)}\left((a,s),\left(a',s'\right)\right) - \mathbb{E}_{\pi_t^{(1)}}\left[\Theta^{(1)}\left((A,s),\left(a',s'\right)\right)\right]\right) + o\left(\eta C\left(\theta_t\right)\right),$$

where the expression is true for any $\eta > 0$, that implies:

$$\int_{\mathcal{S}} \mathbf{m}_{\pi_t}^{(1)}(\mathrm{d}s') \int_{\mathcal{A}} \pi_t^{(1)}(\mathrm{d}a' \mid s') \left(\log \frac{\pi_t^{(1)}}{\pi_*^{(1)}}(a' \mid s') - D_{\mathrm{KL}}\left(\pi_t^{(1)} \| \pi_*^{(1)}\right)(s') \right) \times \left(\Theta^{(1)}\left((a,s),(a',s')\right) - \mathbb{E}_{\pi_t^{(1)}}\left[\Theta^{(1)}\left((A,s),(a',s')\right)\right] \right) = 0.$$

We recall the proposition (6), to get:

$$\int_{\mathcal{S}} \mathbf{m}_{\pi_t}^{(1)}(\mathrm{d}s') \int_{\mathcal{A}} \pi_t^{(1)}(\mathrm{d}a' \mid s') \left(h_t^{(1)}(a,s) - Q_*^{(1)}(a,s) - \mathbb{E}_{\pi_t^{(1)}}\left[h_t^{(1)}(A,s) - Q_*^{(1)}(A,s)\right] \right) \times \left(\Theta^{(1)}\left((a,s),(a',s')\right) - \mathbb{E}_{\pi_t^{(1)}}\left[\Theta^{(1)}\left((A,s),(a',s')\right)\right] \right) = 0.$$

By applying the trick $\mathbb{E}[X(Y - \mathbb{E}[Y])] = \mathbb{E}[(X - \mathbb{E}[X])Y]$, we can rewrite equation above, as follow:

$$\int_{\mathcal{S}} \mathbf{m}_{\pi_t}^{(1)}(\mathrm{d}s') \int_{\mathcal{A}} \pi_t^{(1)}(\mathrm{d}a' \mid s') \left(h_t^{(1)}(a,s) - Q_*^{(1)}(a,s) \right) \times \left(\Theta^{(1)}\left((a,s),(a',s')\right) - \mathbb{E}_{\pi_t^{(1)}}\left[\Theta^{(1)}\left((A,s),(a',s')\right)\right] - \mathbb{E}_{\pi_t^{(1)}}\left[\Theta^{(1)}\left((a,s),(A',s')\right)\right] + \mathbb{E}_{\pi_t^{(1)}}\left[\Theta^{(1)}\left((A,s),(A',s')\right)\right] \right) = 0.$$

One could notice, in second brackets we actually get the expression for $\tilde{\Theta}\left((a,s),(a',s')\right)$ (6). Then we get:

$$\int_{\mathcal{S}} \mathbf{m}_{\pi_t}^{(1)}(\mathrm{d}s') \int_{\mathcal{A}} \pi_t^{(1)}(\mathrm{d}a' \mid s') \left(h_t^{(1)}(a,s) - Q_*^{(1)}(a,s) \right) \times \tilde{\Theta}^{(1)}\left((a,s),(a',s')\right) = 0$$

Now, recall the operator $I_K(f;\mu,\pi)$ we defined in (5). The expression above actually correspond to the:

$$I_{\tilde{\Theta}^{(1)}}\left(\left(h_t^{(1)} - Q_*^{(1)}\right); m_t^{(1)}, \pi_t^{(1)}\right) = 0.$$

Then, by lemma (10), we get that $\left(h_t^{(1)} - Q_*^{(1)}\right) \in \left(\mathcal{H}_{\tilde{\Theta}^{(1)}}\right)^{\perp}$, and since $h_t^{(1)} \in \mathcal{H}_{\Theta^{(1)}}$, as well as $Q_*^{(1)} \in \mathcal{H}_{\Theta^{(1)}}$ by the assumption (2) $\Rightarrow \left(h_t^{(1)} - Q_*^{(1)}\right) \in \mathcal{H}_{\Theta^{(1)}} \cap \left(\mathcal{H}_{\tilde{\Theta}^{(1)}}\right)^{\perp}$. Finally, by lemma (11) we get that the map $\left(h_t^{(1)}(a,s) - Q_*^{(1)}(a,s)\right)$ is constant in $a \Rightarrow \pi_t^{(1)} = \pi_*^{(1)}$.

Since, $\pi_t^{(1)} = \pi_*^{(1)}$, we can again apply the log increment lemma (8), but this time for $m = 2$, and proceed with exactly the same pipeline. This, would bring us to the following equation:

$$I_{\tilde{\Theta}^{(2)}}\left(\left(h_t^{(2)} - Q_*^{(2)}\right); m_t^{(2)}, \pi_t^{(2)}\right) = 0.$$

This time, we can't directly conclude the argument, since $Q_*^{(2)}$ and $h_t^{(2)}$ are not in $\mathcal{H}_{\tilde{\Theta}^{(2)}}$. However, we can do a simple trick: we add and subtract $Q_1^{(1)}$:

$$I_{\tilde{\Theta}^{(2)}}\left(\left(h_t^{(2)} - Q_*^{(1)} + Q_*^{(1)} - Q_*^{(2)}\right); m_t^{(2)}, \pi_t^{(2)}\right) = 0.$$

We note, that $h_t^{(2)}(a,s) = h_t^{(1)}(a,s) + g_t^{(2)}(a,s) = Q_*^{(1)}(a,s) + C_1(s) + g_t^{(2)}(a,s)$, by plug-in this expression, we get:

$$I_{\tilde{\Theta}^{(2)}}\left(\left(C_1 + g_t^{(2)} + Q_*^{(1)} - Q_*^{(2)}\right); m_t^{(2)}, \pi_t^{(2)}\right)$$

Using the linear property of an operator $I_{\tilde{\Theta}^{(2)}}$, we get

$$= I_{\tilde{\Theta}^{(2)}}\left(C_1; m_t^{(2)}, \pi_t^{(2)}\right) + I_{\tilde{\Theta}^{(2)}}\left(\left(g_t^{(2)} + Q_*^{(1)} - Q_*^{(2)}\right); m_t^{(2)}, \pi_t^{(2)}\right) = 0.$$

Let us consider the first term of the sum. Actually, we are applying operator $I_{\tilde{\Theta}^{(2)}}$ on the constant (in a) map C_1. Where, by lemma 12, we know that $C_1 \in (\mathcal{H}_{\tilde{\Theta}^{(2)}})^{\perp}$ and therefore, $I_{\tilde{\Theta}^{(2)}}\left(C_1; m_t^{(2)}, \pi_t^{(2)}\right) = 0$.

$$\Rightarrow I_{\tilde{\Theta}^{(2)}}\left(\left(g_t^{(2)} + Q_*^{(1)} - Q_*^{(2)}\right); m_t^{(2)}, \pi_t^{(2)}\right) = 0.$$

We remark, that by the Realizybility assumption $Q_*^{(1)}(a,s) - Q_*^{(2)}(a,s) \in \mathcal{H}_{\tilde{\Theta}^{(2)}}$ and $g_t^{(2)}$ as well in $\mathcal{H}_{\tilde{\Theta}^{(2)}} \Rightarrow \left(g_t^{(2)} + Q_*^{(1)} - Q_*^{(2)}\right) \in \mathcal{H}_{\Theta^{(1)}} \cap \left(\mathcal{H}_{\tilde{\Theta}^{(1)}}\right)^{\perp}$ by lemma (10) $\Rightarrow \pi_t^{(2)} = \pi_*^{(2)}$.
We repeat the same argument for m = 3, 4 ... . With that, we proof the theorem by an induction. □

2.5 Optimal solution beyond the realizable assumption

Theorem 7 (Global Convergence Beyond the Realizability Assumption)**.** *Training with ideal Light-MPG update converges and if the algorithm converges $\pi_t \to \pi_\infty$. Then $\pi_\infty = \pi_{\theta_*}$, where $\pi_{\theta_*} = \operatorname{argmax}_{\pi_\theta \in \mathcal{P}_n} J_n(\pi_\theta)$ is unique.*

Proof. Let θ_∞ be a critical point for the Light-MPG objective function. Then, from our policy π_{θ_∞} we can contract the action value function $Q_{\theta_\infty}^{(i)}$ for all $i \in [n]$. Important to note, that $Q_{\theta_\infty}^{(i)}$ is not necessary within our RKHS $\mathcal{H}_{\Theta^{(i)}}$. And therefore, the policy build using the algorithm (8), that we we will call $\hat{\pi}_{\theta_\infty}^{(i)}$, won't be present in our set of policies $\mathcal{P}_{\Theta^{(i)}}$. Why we care about $\hat{\pi}_\infty^{(i)}$? It turns out, that $\hat{\pi}_\infty^{(i)}$ is a policy that maximizes the objective $J_n^{(i)}$ (which is objective relevant for the $\vartheta^{(i)}$ set of parameters) under the given action-state distribution, which is $\{m_{\theta_\infty}^{(i)} \pi_{\theta_\infty}^{(i)}\}_{i=1}^n$. Let us have a closer look on the $J_n^{(i)}$ objective:

$$J_n^{(i)}(\pi^{(i)} \mid \pi_{\theta_\infty}) := \int_{\mathcal{S}} m_{\pi_\infty}^{(i)}(ds)\left(\mathbb{E}_{\pi^{(i)}}\left[Q_{\pi_\infty}^{(i)}(A,s)\right] - \tau D_{\mathrm{KL}}\left(\pi^{(i)} \| \bar{\pi}\right)\right)$$

$$= \tau \int_{\mathcal{S}} m_{\pi_\infty}^{(i)}(ds)\left(\log\left(\mathbb{E}_{\bar{\pi}}\left[e^{Q_{\pi_\infty}^{(i)}(A,s)/\tau}\right]\right) - D_{\mathrm{KL}}\left(\pi^{(i)} \| \hat{\pi}_{\theta_\infty}^{(i)}\right)\right).$$

We note, that:

$$\hat{\pi}_{\theta_\infty}^{(i)} = \operatorname{argmin}_{\pi^{(i)}} D_{\mathrm{KL}}\left(\pi^{(i)} \| \hat{\pi}_{\theta_\infty}^{(i)}\right) = \operatorname{argmax}_{\pi^{(i)}} J_n^{(i)}(\pi^{(i)} \mid \pi_{\theta_\infty})$$

$\Rightarrow$ for all $i \in [n], \hat{\pi}_{\theta_\infty}^{(i)}$ maximizes the objective function.

Now, let us define the optimal solution within our policy set:

$$\pi_{\vartheta_*^{(i)}}^{(i)} = \operatorname*{argmin}_{\pi_\vartheta^{(i)} \in \mathcal{P}_{\Theta^{(i)}}} \int_{\mathcal{S}} \mathbf{m}_{\pi_\vartheta}^{(i)}(\mathrm{d}s) D_{\mathrm{KL}}\left(\pi_\vartheta^{(i)} \| \widehat{\pi}^{(i)}\right)(s).$$

In fact, the map $D^{(i)}\left(\pi_\vartheta^{(i)}, \widehat{\pi}^{(i)}\right) := \int_{\mathcal{S}} \mathbf{m}_{\pi_\vartheta}^{(i)}(\mathrm{d}s) D_{\mathrm{KL}}\left(\pi_\vartheta^{(i)} \| \widehat{\pi}^{(i)}\right)(s)$ defines the Bregman divergence (see Appendix C in [3]). Whereas, the theorem on "Uniqueness of projections" (see Theorem 7 in [6]) implies that, the projection on statistical sub-manifold ($\mathcal{P}_{\Theta^{(i)}}$ in our case) w.r.t Bregman divergence is unique and in addition satisfies Pythagorean identity:

- $\pi^{(i)}_{\vartheta^{(i)}_*}$ is unique projection.
- $D^{(i)}\left(\pi^{(i)}_{\vartheta^{(i)}}, \hat{\pi}^{(i)}_{\theta_\infty}\right) = D^{(i)}\left(\pi^{(i)}_{\vartheta^{(i)}}, \pi^{(i)}_{\vartheta^{(i)}_*}\right) + D^{(i)}\left(\pi^{(i)}_{\vartheta^{(i)}_*}, \hat{\pi}^{(i)}_{\theta_\infty}\right)$.

On the other side, let us see what happens when we consider the gradient of the Light-MPG objective, with respect to the $\vartheta^{(i)}$ parameters:

$$\nabla_{\vartheta^{(i)}} J_n = \nabla_{\vartheta^{(i)}} \sum_{j=1}^{n} J_n^{(l)}(\pi^{(l)}_{\vartheta^{(l)}} \mid \pi_{\theta_\infty}) = \nabla_{\vartheta^{(i)}} J_n^{(i)}(\pi^{(i)}_{\vartheta^{(i)}} \mid \pi_{\theta_\infty})$$

$$= \nabla_{\vartheta^{(i)}} \tau \int_{\mathcal{S}} m^{(i)}_{\pi_\infty}(ds)\left(\log\left(\mathbb{E}_{\bar{\pi}}\left[e^{Q^{(i)}_{\pi_\infty}(A,s)/\tau}\right]\right) - D_{\mathrm{KL}}\left(\pi^{(i)}_{\vartheta^{(l)}} \| \hat{\pi}^{(i)}_{\theta_\infty}\right)(s)\right)$$

$$= \tau \int_{\mathcal{S}} \nabla_{\vartheta^{(i)}}\left(m^{(i)}_{\pi_\infty}(ds)\log\left(\mathbb{E}_{\bar{\pi}}\left[e^{Q^{(i)}_{\pi_\infty}(A,s)/\tau}\right]\right)\right) - \tau \int_{\mathcal{S}} \nabla_{\vartheta^{(i)}}\left(m^{(i)}_{\pi_\infty}(ds) D_{\mathrm{KL}}\left(\pi^{(i)}_{\vartheta^{(l)}} \| \hat{\pi}^{(i)}_{\theta_\infty}\right)(s)\right)$$

By the way we have defined the objective, the gradient of the distribution $\nabla_{\vartheta^{(i)}} m^{(i)}_{\pi_\infty}(ds)$ is zero. In addition, $Q^{(i)}_{\pi_\infty}(A,s) = r(a,s) + \int_{\mathcal{S}} p(s,a,ds') V^{(i-1)}_{\pi_\infty}(s')$ and do not depend on the $\vartheta^{(i)} \Rightarrow \nabla_{\vartheta^{(i)}}\left(m^{(i)}_{\pi_\infty}(ds)\log\left(\mathbb{E}_{\bar{\pi}}\left[e^{Q^{(i)}_{\pi_\infty}(A,s)/\tau}\right]\right)\right) = 0$. Therefore, we get:

$$\nabla_{\vartheta^{(i)}} J_n = -\tau \int_{\mathcal{S}} m^{(i)}_{\pi_\infty}(ds) \nabla_{\vartheta^{(i)}} D_{\mathrm{KL}}\left(\pi^{(i)}_{\vartheta^{(l)}} \| \hat{\pi}^{(i)}_{\theta_\infty}\right)(s) = -\tau \nabla_{\vartheta^{(i)}} D^{(i)}\left(\pi^{(i)}_{\vartheta^{(i)}}, \hat{\pi}^{(i)}_{\theta_\infty}\right)$$

We note, that the objective gradient coincide with the gradient of Bregman divergence. Since, we are at the critical point, using Pythagorean identity we have:

$$0 = \nabla_{\vartheta^{(i)}} J_n = -\tau \nabla_{\vartheta^{(i)}} D^{(i)}\left(\pi^{(i)}_{\vartheta^{(i)}}, \hat{\pi}^{(i)}_{\theta_\infty}\right) = -\tau \nabla_{\vartheta^{(i)}}\left(D^{(i)}\left(\pi^{(i)}_{\vartheta^{(i)}}, \pi^{(i)}_{\vartheta^{(i)}_*}\right) + D^{(i)}\left(\pi^{(i)}_{\vartheta^{(i)}_*}, \hat{\pi}^{(i)}_{\theta_\infty}\right)\right)$$

$$\Rightarrow 0 = \nabla_{\vartheta^{(i)}} D^{(i)}\left(\pi^{(i)}_{\vartheta^{(i)}}, \pi^{(i)}_{\vartheta^{(i)}_*}\right).$$

Whereas, $\pi^{(i)}_{\vartheta^{(i)}_*}$ is within our policy space $\mathcal{P}_{\Theta^{(i)}} \Rightarrow$ by the theorem (6) $\pi^{(i)}_{\vartheta^{(i)}} = \pi^{(i)}_{\vartheta^{(i)}_*}$. It shows that for all $i \in [n]$, $\vartheta^{(i)}_\infty$ maximizes the gradient objective $J^{(i)}_n$:

$$J^{(i)}_n\left(\pi^{(i)}_{\vartheta_\infty}\right) = \max_{\vartheta^{(i)} \in \mathbb{R}^{P_i}} J^{(i)}_n\left(\pi^{(i)}_{\vartheta^{(i)}} \mid \pi^{(1)}_{\theta^{(1)}_\infty}, \ldots, \pi^{(n)}_{\theta^{(n)}_\infty}\right).$$

Since this is true for every $i = 1, \ldots, n$ and since maxima can be taken in any order, we have that

$$J_n(\pi_{\theta_\infty}) = \max_{\theta \in \mathbb{R}^P} J_n(\pi_\theta)$$

θ_∞ maximizes the Light MPG objective and by theorem on "Uniqueness of projections" such policy is unique. □

Definition 7 (Projectional Consistency in Light MPG). *Let $\theta \in \mathbb{R}^P$ and $\mathbf{m}^{(i)}$ denote the law of state S_{n-i} under the policy π_θ, with $\mathbf{m}^{(n)} = \nu_0$. For each $i \in \{1, \ldots, n\}$, define $P_i : L^2\left(\mathbf{m}^{(i)}(\mathrm{d}s)\pi^{(i)}_\theta(\mathrm{d}a \mid s)\right) \to \mathcal{H}_{\Theta^{(i)}}$ as the orthogonal projection onto $\mathcal{H}_{\Theta^{(i)}}$ in the $L^2\left(\mathrm{m}^{(i)}(\mathrm{d}s)\pi^{(i)}_\theta(\mathrm{d}a \mid s)\right)$ sense, where $\Theta^m((a,s),(a',s')) = \phi^{(m)}(a,s) \cdot \phi^{(m)}(a',s')$. A policy π_θ is said to satisfy the projectional consistency property if and only if for all $i \in \{1, \ldots, n\}$, the following holds:*

- $Q_{\cdot}^{(i)}(a,s) := Q_{\cdot}^{(i-1)}(a,s) + P_{i+1}\left(Q_{\pi_\theta}^{(i)}(a,s) - Q_{\cdot}^{(i-1)}(a,s)\right)$
- $\pi_\theta^{(i)}(a \mid s) = \bar{\pi}(a \mid s)\frac{\exp\left(Q_{\cdot}^{(i)}(a,s)/\tau\right)}{\int_{\mathcal{A}} \bar{\pi}(\mathrm{d}a'|s)\exp\left(Q_{\cdot}^{(i)}(a',s)/\tau\right)}$.

Proposition 7. *The global optimum π_θ from Theorem (7) is the only policy in $\mathcal{P}_n$ that satisfies the projectional consistency property (7) for Light MPG.*

Proof. See [3] page 25. The reasoning is almost identical.

□

2.6 Connection between Light and Original Objective Functions

A pertinent question arises: What exactly are we optimizing by employing the Light-MPG update (refer to 2.3)? As previously introduced, the Light-MPG's objective function closely resembles the original one. It turns out, that we can show that the Light-MPG also optimizes the initial objective function. In other words, the critical point of the Light objective is a policy that maximizes the original objective. Bearing this in mind, let us proceed to prove this fact.

Proposition 8. *Assuming Realizable assumption (see 2), let $\theta = \{\vartheta^{(i)}\}_{i=1}^n$ be a critical point of the Light objective $J_n^{Light} \Rightarrow \theta$ is a critical point of the original objective J_n.*

Proof. We will show that for all $i \in [n]$, $\nabla_{\vartheta^{(i)}} J_n(\pi_\theta) = 0$. Then, let us start by considering the gradient w.r.t $\vartheta^{(n)}$. It turns out, that both original and light objective share the same gradient expression and $\nabla_{\vartheta^{(n)}} J_n = \nabla_{\vartheta^{(n)}} J_n^{Light} = 0$. Which can be seen from the $\nabla_{\vartheta^{(n)}} J_n$ definition (see 2.1). In particularly, it's not True for any other set of parameters $\vartheta^{(l)}$ $l = 1, \dots n-1$.

So, to conduct the proof, we will use the fact that by the Theorem 7, the critical point of the Light objective is unique and optimal for underlying distribution, with thus we can use Proposition 1 (see remark 1 below the Proposition).

$$\pi_\theta^{(i)}(a \mid s) = \bar{\pi}(a \mid s)\exp\left(\left(Q_\theta^{(i)}(a,s) - V_\theta^{(i)}(s)\right)/\tau\right) \qquad \text{for } i = 1 \dots n.$$

With this said, we will begin our proof.

Without loss of generality, we will demonstrate the proof only for $\vartheta^{(n-2)}$ set of parameters. Since, for shorter horizon notations gets quite involved. In addition,the same approach trivially generalize for any horizon step.

$$\nabla_{\vartheta^{(n-2)}} J_n(\pi_\theta) = \nabla_{\vartheta^{(n-2)}} \iint_{S\times A} m^{(n)}(ds)\pi_\theta^{(n)}(da|s)\left(r(a,s) - \tau\log\frac{\pi_\theta^{(n)}}{\hat{\pi}} + \int_S p(s,a,ds')V_\theta^{(n-1)}(s')\right)$$

$$= \iint_{S\times A} m^{(n)}(ds)\pi_\theta^{(n)}(da|s)\nabla_{\vartheta^{(n-2)}}\log\pi_\theta^{(n)}(a|s)\left(r(a,s) - \tau\log\frac{\pi_\theta^{(n)}}{\hat{\pi}} + \int_S p(s,a,ds')V_\theta^{(n-1)}(s')\right)$$

$$+ \iint_{S\times A} m^{(n)}(ds)\pi_\theta^{(n)}(da|s)\left(-\tau\nabla_{\vartheta^{(n-2)}}\log\pi_\theta^{(n)} + \int_S p(s,a,ds')\nabla_{\vartheta^{(n-2)}}V_\theta^{(n-1)}(s')\right)$$

Using Soft-max property (3), we get:

$$= \iint_{S\times A} m^{(n)}(ds)\pi_\theta^{(n)}(da|s)\nabla_{\vartheta^{(n-2)}}\log\pi_\theta^{(n)}(a|s)\left(r(a,s)-\tau\log\frac{\pi_\theta^{(n)}}{\hat{\pi}}+\int_S p(s,a,ds')V_\theta^{n-1}(s')\right)$$

$$+\iint_{S\times A} m^{(n)}(ds)\pi_\theta^{(n)}(da|s)\int_S p(s,a,ds')\nabla_{\vartheta^{(n-2)}}V_\theta^{(n-1)}(s')$$

By the Proposition 1, we discussed above, we can write:

$$\tau\log\frac{\pi_\theta^{(n)}}{\hat{\pi}} = V_\theta^{(n)}(s)-(r(a,s)+\int_S p(s,a,ds')V_\theta^{n-1}(s')).$$

Plug-in into the first term of the sum, we get:

$$\iint m^{(n)}(ds)\pi_\theta^{(n)}(da|s)\nabla\log\pi_\theta^{(n)}(a|s)V_\theta^{(n)}(s)+\iint m^{(n)}(ds)\pi_\theta^{(n)}(da|s)\int p(s,a,ds')\nabla V_\theta^{(n-1)}(s')$$

Again, by the Soft-max property the first term is zero.

$$\Rightarrow \nabla_{\vartheta^{(n-2)}}J_n(\pi_\theta) = \iint m^{(n)}(ds)\pi_\theta^{(n)}(da|s)\int p(s,a,ds')\nabla V_\theta^{(n-1)}(s')$$

$$\iiint m^{(n)}(ds)\pi_\theta^{(n)}(da|s)p(s,a,ds')\int\nabla\left(\pi_\theta^{(n-1)}(da'|s')\left(r(a',s')-\tau\log\frac{\pi_\theta^{(n-1)}}{\hat{\pi}}+\int p(s',a',ds'')V_\theta^{(n-2)}(s'')\right)\right)$$

Again, using Soft-max property, we obtain:

$$= \iint m^{(n-1)}(ds')\pi_\theta^{(n-1)}\nabla\log\pi_\theta^{(n-1)}\left(r(a',s')-\tau\log\frac{\pi_\theta^{(n-1)}}{\hat{\pi}}+\int p(s',a',ds'')V_\theta^{(n-2)}(s'')\right)$$

$$+\iint m^{(n-1)}(ds')\pi_\theta^{(n-1)}\int p(s',a',ds'')\nabla V_\theta^{(n-2)}(s'')$$

By the same argument, of the Proposition 1, the first term is zero by the Soft-max property. Then,

$$\Rightarrow \nabla_{\vartheta^{(n-2)}}J_n(\pi_\theta) = \iint m^{(n-1)}(ds')\pi_\theta^{(n-1)}\int p(s',a',ds'')\nabla_{\vartheta^{(n-2)}}V_\theta^{(n-2)}(s'')$$

$$= \mathbb{E}_{\pi_\theta}[\nabla_{\vartheta^{(n-2)}}V_\theta^{(n-2)}(S)] = \nabla_{\vartheta^{(n-2)}}\mathbb{E}_{\pi_\theta}[V_\theta^{(n-2)}(S)] = \nabla_{\vartheta^{(n-2)}}J_n^{(n-2)}(\pi_{\vartheta^{(n-1)}}|\pi_\theta) = \nabla_{\vartheta^{(n-2)}}J_n^{Light}(\pi_\theta),$$

whereas, π_θ is critical point of the Light objective $\Rightarrow \nabla_{\vartheta^{(n-2)}}J_n^{Light}(\pi_\theta)=0$.

We see that by iterating arguments of the Soft-max property and Proposition 1, we can obtain the same result for the gradient w.r.t all $\vartheta^{(l)}, l\in[n]$. With that we conclude the proof. □

2.7 Complete framework

This section serves as a summary and conclusion for Chapter 2. We aim to consolidate all the theoretical results proven thus far and draw conclusions. Additionally, our goal is to construct a comprehensive theoretical framework and enumerate a list of conjectures that could address the identified theoretical gaps.

Let's begin by revisiting the derived results and identifying these gaps.

Incomplete theoretical pipeline

Our research has examined two algorithms: S-MPG (refer to 2.1) and Light-MPG (refer to 2.3). From a practical standpoint, the Light version appears more favorable. However, in terms of theoretical achievements, both algorithms present limitations. For the S-MPG algorithm, we established a well-defined objective function J_n, which facilitated the use of a common approach to prove global convergence (see 5). Nevertheless, we could not establish results for optimality and uniqueness. In the case of Light-MPG, with rather complex and implicitly defined objective function, we did not achieve a result on the global convergence. However, we were able to prove that, upon convergence assumption, the critical point is unique and optimal with respect to the projection (see 7). Additionally, we derived a new projection consistency property for this critical point (see 7). Thus, we have one algorithm with assured convergence, and another algorithm "if" converge $\Rightarrow$

optimal. In the next subsection, we will leverage all our findings to construct a complete sound theoretical pipeline.

Our proposition

Optimality Check. Given that each algorithm does not guarantee satisfactory results, whether in terms of convergence or policy optimality, a practical approach involves what we term as the "optimality check." This method entails using a combination of S-MPG and Light-MPG updates. Specifically, one can train the algorithm using S-MPG, and then verify if the derived critical point is also a critical point for Light-MPG by applying the Light-MPG update. If the parameters remain static, this confirms that it is indeed a critical point for both algorithms. This procedure may be viewed as an iterative pipeline, alternating between the two updates. Once such a point is identified, Theorem 7 can be employed to establish optimality. Subsequently, we can evaluate whether the derived solution is the desired one or if there is still room for improvement.

Simply Apply Light-MPG. While hoping for convergence with respect to the Light-MPG update is viable, theoretical guarantees of convergence cannot be provided. Nevertheless, it is often the case in practice. Without resource constraints, this approach is certainly worth considering. Additionally, in our practical tests with some popular benchmark examples, convergence was consistently achieved. More broadly, we believe that a theoretical result for convergence could also be established for Light-MPG. Various machine learning techniques, such as dynamically decreasing the learning rate or weights normalization, can be employed to facilitate convergence. Moreover, I hypothesize that by introducing scaling with respect to the step horizon learning rate, $\eta^{(i)}$, we can enhance stability. Specifically, $\eta^{(i)} \sim \mathbb{E}_{\pi_\theta}\left[C_i \nabla_{\theta^{(i-1)}} \log \pi_\theta^{(i)}\left(A_{n-i} \mid S_{n-i}\right)\right] \cdot \eta$, the idea being to amplify the update for higher step horizons to counterbalance the impact of shorter horizons on longer ones, which I believe would lead to improved algorithm stability.

Roadmap for a Complete Framework

In this chapter, we present four conjectures/hypotheses that are instrumental in establishing a complete theoretical framework. Throughout this project, we have endeavored to prove them; however, these endeavors require a fresh approach, employing novel ideas and techniques. Currently, these remain as open questions for further exploration.

1. S-MPG Uniqueness. The first conjecture concerns the uniqueness of the critical point for the S-MPG update. We already know that one critical point is the optimal solution by Proposition 8. Demonstrating that S-MPG has a unique critical point would suggest that it coincides with the critical point of the Light objective and is thus optimal. For extension to the non-realizable case, Proposition 8 needs to be proven for that scenario.

2. Inverse of Proposition 8. Similar to the first conjecture, we explore the idea that the critical point of the original objective is also a critical point of the Light objective. We aim to establish an "if and only if" condition for Proposition 8. Such a result would imply the optimality of the Light objective on the critical points of the original one.

3. Optimality of S-MPG. Another avenue is to establish Theorem 6 (Theorem on the optimal solution in Realizable settings) for the ideal update with the original objective. Specifically, we aim to show that, at least under the Realizable assumption, the solution is optimal. The challenge here is the complexity of notation and the emergence of non-symmetrical s.p.d kernels in our standard pipeline.

4. Convergence of Light-MPG. Focusing on the Light objective, the remaining task is to prove its convergence. One approach could be to demonstrate the L-smoothness of the objective. Alternatively, the *positive angle* approach involves proving that for any $\theta \in \mathbb{R}^P$, the inequality

$$\langle \nabla_{\vartheta^{(i)}} J_n(\theta), \nabla_{\vartheta^{(i)}} J^{Light}(\theta) \rangle > \epsilon_t > 0 \quad \text{for } i = 1, \dots, n.$$

holds. We know already, that for $i = n$ condition is satisfied, since $\nabla_{\vartheta^{(n)}} J_n(\theta_t) = \nabla_{\vartheta^{(n)}} J_n^{(n)} = \nabla_{\vartheta^{(n)}} J_n^{Light}(\theta_t)$.In particularly, under some additional conditions, like horizon-dependent learning rates (mentioned earlier in the chapter),this approach appears promising.

Proving any of these conjectures would significantly enhance our theoretical understanding, paving the way towards a comprehensive theoretical framework. Such achievements would not only validate the existing results but also fill the current gaps, offering a more robust and complete picture of our algorithm's theoretical underpinnings. The pursuit of these conjectures represents an exciting frontier in our research, promising to contribute significantly to the field.

3 Neural MPG

In this section, we delve into the Neural Network implementation of the Matryoshka algorithm. Our focus will be on testing various Matryoshka realizations (see A.2) on some standard benchmarks environments (see A.1). This analysis aims to understand how the algorithm performs under different conditions and to identify its strengths and limitations.

We will also discuss the differences between the log-linear parametrization (one we used until know) with a feature map, as introduced in Chapter 2, and the Neural Network scenario. This part of our exploration seeks to pinpoint where the theoretical constructs from earlier chapters align or diverge when applied to Neural Networks. The aim is to assess the extent to which our existing theoretical framework is adaptable to these more complex models.

Lastly, we will present arguments supporting the superiority of the "Shared case" design for larger scale horizons in comparison to the standard Matryoshka design. This analysis will highlight why and how the shared architecture model offers advantages, particularly when dealing with extensive and complex horizon scenarios.

3.1 Neural MPG

This section explores the main aspects of Neural MPG, as presented in the work of Francois (refer to [3] Ch. 3.5 and A.2). Our focus will be on applying these principles to the "Shared parameters case." We begin by establishing the notation that will be used throughout this section.

Set-up

As in our previous discussion, the algorithm aims to estimate preference functions $h_\theta^{(i)}$ for each horizon step, $i = 1, \dots, n$. In this context, however, the preference function is parameterized by a Neural Network, which could be any architecture, including RNNs, LSTMs, or CNNs. We have n distinct Neural Networks, referred to as "blocks," for each horizon step. The set of parameters for the Neural Network corresponding to the ith step horizon is denoted by $\theta^{(i)}$.

To conceptualize the neural network blocks, consider $\sigma : \mathbb{R} \to \mathbb{R}$ as an activation function or any form of measurable non-linearity. We recursively define each block i of depth L_i as follows:

$$
\begin{aligned}
\alpha_0^{(i)}(x_{\text{input}}) &:= x_{\text{input}} \\
&\vdots \\
\tilde{\alpha}_{l+1}^{(i)}(x_{\text{input}}) &:= W_l^{(i)} \cdot \alpha_l^{(i)}(x_{\text{input}}) \\
\alpha_{l+1}^{(i)}(x_{\text{input}}) &:= \sigma\left(W_l^{(i)} \cdot \alpha_l^{(i)}(x_{\text{input}})\right) \\
&\vdots \\
h_\theta^{(i)}(x_{\text{input}}) &:= W_{L_i}^{(i)} \cdot \alpha_{L_i-1}^{(i)}(x_{\text{input}}),
\end{aligned}
$$

where $W_l^{(i)}$ is the set of trainable parameters, and $x_{\text{input}} = (a, s)$.

Similarly to the original linear algorithm, we can define a feature map for the Neural MPG. The concept is straightforward: recognizing that the output layer is linear in relation to $\alpha^{(i)}_{L_i-1}$, we define our feature map as $\psi^{(i)} := \alpha^{(i)}_{L_i-1}$, the last hidden layer. Specifically, after a certain point in training, it is possible to fix all trainable parameters of the neural network except for $W^{(i)}_{L_i}$ and continue training. In this scenario, we align with the settings of Chapter 2, effectively using the Neural Network to generate a feature map. Once the hidden layers are fixed, we can utilize the convergence and optimality theorems. However, this approach represents merely one method to generate the feature map and should not be confused with the core concept of "Neural MPG." In this section, we maintain all parameters as trainable throughout the training process, for reasons that will be elucidated in the next chapter. First, let us define another essential tool: the kernel related to our feature map.

Definition 8 (Conjugate Kernel)**.** *When a policy $\pi \in \mathcal{P}_n$ is parameterized by a neural network, we define the symmetric positive definite (s.p.d) conjugate kernel (CK) for the ith step horizon as:*

$$\Sigma^{(i)}(x, x') := \alpha^{(i)}_{L_i-1}(x) \cdot \alpha^{(i)}_{L_i-1}(x') \qquad x, x' \in \mathcal{A} \times \mathcal{S}.$$

We denote Σ to indicate the set of CKs for each step horizon. The corresponding Reproducing Kernel Hilbert Space (RKHS) is denoted as $\mathcal{H}_\Sigma$, and we refer to $\mathcal{P}^\Sigma_n$ for all policies whose preferences belong to $\mathcal{H}_\Sigma$.

To further our analysis, we also define the *Neural Tangent Kernel* (NTK), which is instrumental in describing the training dynamics.

Definition 9 (NTK)**.** *For the i-step preference function, we define*

$$\Theta^{(i)}(x, x') := \nabla_\theta h^{(i)}(x) \cdot \nabla_\theta h^{(i)}(x') \qquad x, x' \in \mathcal{A} \times \mathcal{S}.$$

We denote Θ to indicate the set of NTKs for each step horizon. The corresponding RKHS is denoted as $\mathcal{H}_\Theta$, and we refer to $\mathcal{P}^\Theta_n$ for all policies whose preferences belong to $\mathcal{H}_\Theta$.

Next, we will explore the principal differences between Neural Networks and linear-feature design, particularly focusing on the potential advantages offered by Neural MPG.

Key Difference

The key distinction in our approach lies in the dynamic nature of the kernel Σ, which evolves over time during the training process (denoted as Σ_t to indicate the time step). This dynamic aspect introduces non-convexity to the problem, precluding guarantees of global convergence. Nevertheless, at the critical point $\pi_\infty = \lim_{t\to\infty} \pi_t$, where the kernel becomes static, we achieve optimality within the given feature space as per the optimal theorem (see 3):

$$\pi_\infty = \underset{\pi \in \mathcal{P}^{\Sigma_\infty}_n}{\operatorname{argmax}} J_n(\pi)$$

Ideally, if the optimal preferences reside within the corresponding RKHS $\mathcal{H}_{\Sigma_\infty}$, then π_∞ aligns with the optimal policy π_*.

Conversely, considering the landscape of Reinforcement Learning and Deep Learning, the appropriate feature map is often not explicitly determinable. Current practices indicate that allowing Neural Networks to autonomously discover effective features often yields superior results, particularly in over-parameterized regimes (for example modern embedding techniques in LLMs). Thus, in most practical scenarios, it is advisable to maintain all parameters as trainable until the conclusion of the training process.

NTK Regime

As discussed, in over-parameterized settings, we have a higher likelihood of reaching the optimal solution. Generally, the more parameters added to a Neural Network, the greater the probability of obtaining an appropriate RKHS. This is simply because an increased number of neurons expands $\mathcal{P}_n^{\Sigma}$, our policy space. However, a pertinent question arises: Is it feasible to guarantee global convergence to the optimal solution π_*? The answer is affirmative. Initially, even in environments with continuous states/actions, discretization to a finite state/action environment is inevitable for training. Consequently, achieving an over-parameterized state where the Neural Network can represent the optimal solution is always plausible.

To ensure global convergence, the problem needs to be convex, implying a static kernel. We refer to such regimes as "lazy" or "NTK" regimes. In these settings, the NTK kernel remains unchanged throughout the training. Formally, for any $t \in \mathbb{N}$:

$$\Theta_t^{(i)}(x, x') \simeq \Theta_0^{(i)}(x, x') \qquad x, x' \in \mathcal{A} \times \mathcal{S}.$$

These regimes are characteristic of Wide Neural Networks, where the number of neurons at each depth is sufficiently large. As this number approaches infinity, we can guarantee convergence to the global optimum. Therefore, the theoretical framework discussed can be aptly applied to Neural Networks operating in the NTK regime.

Shared NN

Now, let's explore how the "shared" Matryoshka model is represented with Neural Network (NN) parametrization. Fundamentally, the structure remains largely similar. Formally, for $i = 1, \ldots, n$, we define a block as follows:

$$\begin{aligned}
\alpha_0^{(i)}(x_{\text{input}}) &:= x_{\text{input}} \\
&\vdots \\
\tilde{\alpha}_{l+1}^{(i)}(x_{\text{input}}) &:= W_l^{(i)} \cdot \alpha_l^{(i)}(x_{\text{input}}) \\
\alpha_{l+1}^{(i)}(x_{\text{input}}) &:= \sigma\left(W_l^{(i)} \cdot \alpha_l^{(i)}(x_{\text{input}})\right) \\
&\vdots \\
\text{block}^{(i)}(x_{\text{input}}) &:= W_{L_i}^{(i)} \cdot \alpha_{L_i-1}^{(i)}(x_{\text{input}}),
\end{aligned}$$

The preference functions h are then constructed by sequentially combining blocks with the preference functions of previous steps:

$$\begin{aligned}
h_\theta^{(1)}(x_{\text{input}}) &:= \text{block}_{\vartheta^{(1)}}^{(1)}(x_{\text{input}}) \\
&\vdots \\
h_\theta^{(i)}(x_{\text{input}}) &:= \text{block}_{\vartheta^{(i)}}^{(i)}(x_{\text{input}}) + h_\theta^{(i-1)}(x_{\text{input}}) \\
&\vdots \\
h_\theta^{(n)}(x_{\text{input}}) &:= \text{block}_{\vartheta^{(n)}}^{(n)}(x_{\text{input}}) + h_\theta^{(n-1)}(x_{\text{input}}),
\end{aligned}$$

To maintain consistency with Chapter 2, $\vartheta^{(i)}$ denotes the set of trainable variables for the i-th block, and $\theta^{(i)}$ represents the set of trainable parameters $\{\vartheta^{(j)}\}_{j=1}^{i}$.

In the context of the Light-MPG update, we can establish similar results for the optimal solution at the critical point, using Theorem 7 7.

$$\pi_\infty = \underset{\pi \in \mathcal{P}_n^{\Sigma\infty}}{\operatorname{argmax}} J_n(\pi)$$

However, there is a slight distinction in the terms of the policy set $\mathcal{P}_n^{\Sigma\infty}$ compared to classical design. For each step horizon, our conjugate kernel $\Sigma^{(i)}$ is not directly related to the preference function $h_\theta^{(i)}$ itself, but rather to the difference $(h_\theta^{(i)} - h_\theta^{(i-1)})$. This gap is precisely what our trainable parameters $\vartheta^{(i)}$ are designed to address. Consequently, we define a policy $\pi^{(i)} \in \mathcal{P}_n^{\Sigma\infty}$ as having a preference function in the form $h^{(i-1)}(x) + u(x)$, where $u(x)$ belongs to $\mathcal{H}_{\Sigma^{(i)}}$.

In a manner similar to the classical approach, if at the end of training $\pi_* \in \mathcal{P}_n^{\Sigma\infty}$, we are fortunate, and $\pi_\infty = \pi_*$. This raises an important question: Is it more probable for π_* to be represented by the RKHS in the non-shared case as compared to the shared one? In the absence of additional information about the environment, both approaches are essentially equivalent in this regard. Bearing this in mind, let us proceed to the next section to discuss why the "Shared" case might be more suitable and desirable compared to the classical Matryoshka design.

3.2 Motivation for "Shared" design

The optimality theorem illuminates our optimization objective within each layer of the S-MPG algorithm. Specifically, we aim to optimize the "gap" between two successive preference functions, $(Q_\theta^{(i)} - Q_\theta^{(i-1)})$, by adjusting the parameters $\vartheta^{(i)}$. This approach contrasts with the original algorithm, where the target was $Q_\theta^{(i)}$ itself, using an independent set of parameters $\theta^{(i)}$. And we believe, that this inherent structure in the algorithm is advantageous for a wide range of environments, effectively reducing the degrees of freedom required to express the optimal solution. This implies that while $(Q_\theta^{(i)} - Q_\theta^{(i-1)})$ can be represented in the corresponding RKHS $\mathcal{H}_{\Theta^{(i)}}$, the term $Q_\theta^{(i)}$ alone may not be representable in this RKHS.

Another motivation stems from the observation that, especially for sufficiently large step horizons, adjacent policies tend to exhibit similar behavior, with the similarity increasing with the horizon size. We encapsulate this observation in the following assumption:

Assumption 4. *For any $k \in \mathbb{N}$, there exists a fixed horizon n and $n_k \in \mathbb{N}$ such that for $n > m > n_k$, it holds that:*

$$\left| \frac{\pi_*^m}{\pi_*^{m-1}}(a,s) \right| \leq (1 + \frac{1}{k}) \quad \textit{for all } (a,s) \in \mathcal{A} \times \mathcal{S}.$$

Under this scenario, our methodologies not only reduce the memory footprint of the algorithm but also maintain the accuracy.

Let us examine the target action value and its difference "gap":

$$Q_*^{(i+1)}(a,s) = r(a,s) + \int_S p(a,s,ds') V_*^{(i)}(s')$$

$$= r(a,s) + \int_S p(a,s,ds')V_*^{(i-1)}(s') + \int_S p(a,s,ds')\left(V_*^{(i)}(s') - V_*^{(i-1)}(s')\right)$$

$$\Rightarrow Q_*^{(i+1)}(a,s) - Q_*^{(i)}(a,s) = \int_S p(a,s,ds')\left(V_*^{(i)}(s') - V_*^{(i-1)}(s')\right)$$

Thus, the difference "gap" can be quantified by the one-step shift in the optimal value function

$$\int_S p(a,s,ds')\left(V_*^{(i)}(s') - V_*^{(i-1)}(s')\right)$$

.

Remark. It is important to highlight that Q_θ, as per Theorem 7, is the function we strive to approximate. Q_*, on the other hand, represents our target, defined by the environment itself.

Proof of concept

Now, let us consider the *Toy* game (see 1.1), and see how our reasoning is suitable for this environment. In this experiment the *Toy* environment characterized by a state space comprising nine distinct states. This environment is defined by randomly initialized reward functions $r(a,s)$ and deterministic transition probabilities $p(s,a,s') = \delta_{\hat{s},s'}$. The decision-making process is constrained by a fixed horizon, denoted as $n = 200$.

We focus on the analysis of the value difference across these horizons, quantified by the gap in the target value function:

$$\Delta V_*^{(i)}(s) = V_*^{(i)}(s) - V_*^{(i-1)}(s),$$

where $\Delta V_*^{(i)}(s)$ represents the difference in value functions between successive horizons at state s.

The analysis is further augmented by examining the standard deviation of these value differences over a span of 15 consecutive horizons, formally expressed as:

$$\text{std}\left(\{\Delta V_*^{(J)}(s)\}_{j=i}^{i+15}\right), \quad \text{for } i = 1,\ldots,175.$$

This statistical measure provides insights into the variability of the value function over the horizon.

By looking on the bottom plot on the figure 2, we see that for each state $s \in \mathcal{S}$, the standard deviation of $\Delta V_*^{(i)}(s)$ is decreasing with step horizon. It indicates that no matter the horizon, the difference of value function associated with one step shift is the same for big enough step horizons: there exist $m \leq n$, such that $\Delta V_*^{(i_1)}(s) \simeq \Delta V_*^{(i_2)}(s)$ for any $i_1, i_2 > m$ and any $s \in \mathcal{S}$.

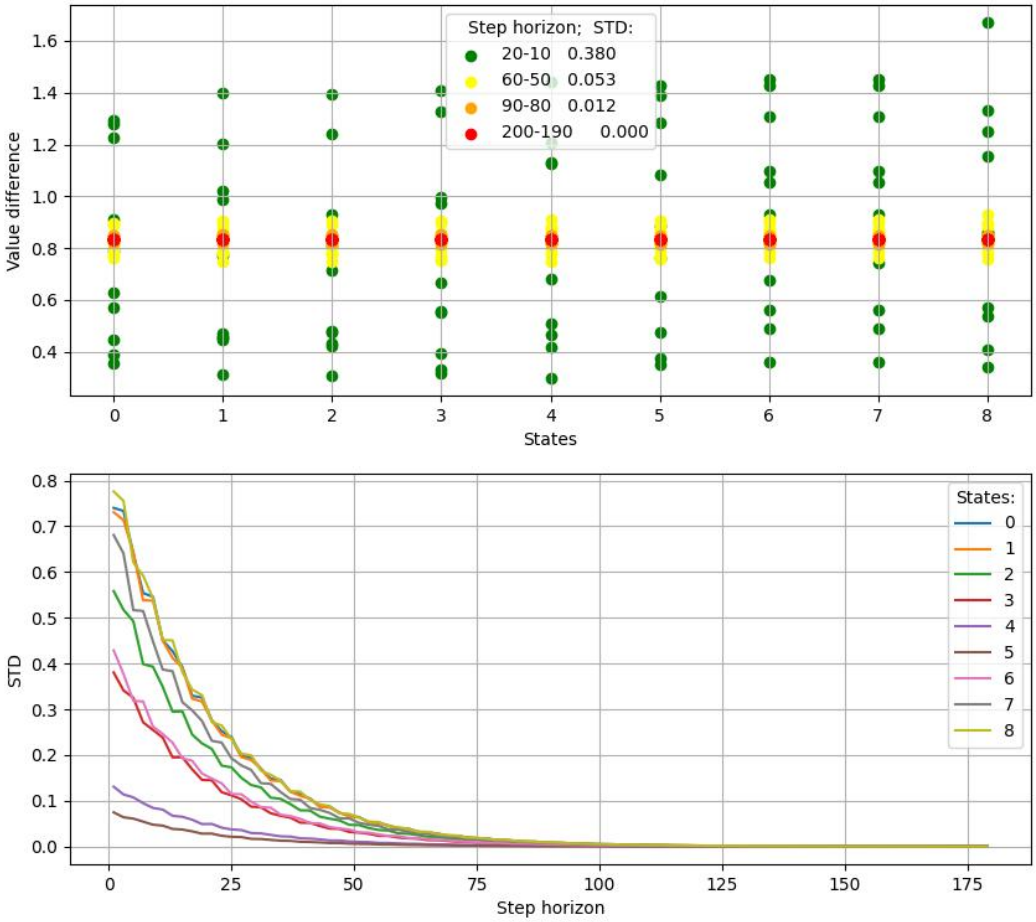


Figure 2: The first plot illustrates sample observations of the value difference for each state. Additionally, we estimate the standard deviation across each state, as depicted by different colors in the plot (refer to the legend for details). The second plot presents the standard deviation of the next 15 value differences with respect to each step of the horizon.

A particularly interesting observation emerges from the analysis of the upper plot. We note that there is not only a consistent shift in the value function for each state, but this shift remains uniform across all states. Notably, for higher horizons (illustrated by red samples), the difference in value functions across all states converges to a similar value, with a standard deviation (std) approaching zero, and is approximately equal to 0.8. This observation indicates that not only does the assumption 4 hold, but it also leads to an even stronger result: there exists a horizon $m < n$ such that for all $i > m$, it holds that:

$$\left|\Delta V_*^{(i)}(s)\right| \simeq C,$$

where $C > 0$ is a constant. Consequently, this implies:

$$\left|Q_*^{(i)}(a,s) - Q_*^{(i-1)}(a,s)\right| \simeq C \qquad \text{for all } (a,s) \in \mathcal{A} \times \mathcal{S}.$$

Intuitively, this result suggests that for higher horizons, the policies not only exhibit similar behaviors, but at the preference function level, we essentially need to account for a constant value C.

This empirical observation can be encapsulated in a theoretical asymptotic result (see 9), demonstrating that for sufficiently high step horizons l, the difference $Q_*^{(l)}(a,s) - Q_*^{(l-1)}(a,s)$ becomes a constant for any action/state pair.

Proposition 9. *Let $s \in \mathcal{S}$ and $|\mathcal{S}| < \infty$, admitting assumption (4). For all $(a,s) \in \mathcal{A} \times \mathcal{S}$ and $n \to \infty$ we have*

$$Q_*^{(n)}(a,s) - Q_*^{(n-1)}(a,s) \to C,$$

where $C \in \mathbb{R}$ is a constant.

Proof. As we have seen before, we can write $Q_*^{(n)}$ as follow:

$$Q_*^{(n)}(a,s) = Q_*^{(n-1)}(a,s) + \int_S p(a,s,ds') \left(V_*^{(n-1)}(s') - V_*^{(n-2)}(s')\right)$$

Let us, focus on the difference: $V_*^{(n-1)}(s') - V_*^{(n-2)}(s')$. For $k \in \mathbb{N}$, we consider n_k from the assumption (see 4) and we note

$$V_*^{(n-1)}(s') = \mathbb{E}_{\pi_*}\left[\sum_{k=0}^{n-n_k-1} \left(R_k - D_{KL}(\pi_*^{(n-1-k)} \| \hat{\pi})(S_k)\right) \mid S_0 = s'\right] + \mathbb{E}_{\pi_*}\left[\sum_{k=n-n_k}^{n-2} (\dots) \mid S_0 = s'\right]$$

$$V_*^{(n-2)}(s') = \mathbb{E}_{\pi_*}\left[\sum_{k=0}^{n-n_k-1} \left(R_k - D_{KL}(\pi_*^{(n-2-k)} \| \hat{\pi})(S_k)\right) \mid S_0 = s'\right] + \mathbb{E}_{\pi_*}\left[\sum_{k=n-n_k}^{n-3} (\dots) \mid S_0 = s'\right]$$

We will refer to first terms as $I_1^{(n-1)}$, $I_1^{(n-2)}$ and $I_2^{(n-1)}$, $I_2^{(n-2)}$ for second ones. Then we have:

$$\Delta V_*^{(n-1)}(s') = V_*^{(n-1)}(s') - V_*^{(n-2)}(s') = \left(I_1^{(n-1)}(s') - I_1^{(n-2)}(s')\right) + \left(I_2^{(n-1)}(s') - I_2^{(n-2)}(s')\right) \quad (20)$$

We begin, by showing that difference of the first term converge to zero, as $n \to \infty$:

$$\left(I_1^{(n-1)}(s') - I_1^{(n-2)}(s')\right) \to 0$$

Consider $I_1^{(n-1)}(s')$

$$I_1^{(n-1)}(s') = \int_A \pi_*^{(n-1)}(da|s') \left(r(a,s) - \tau \log \frac{\pi^{(n-1)}}{\hat{\pi}}\right) + \int_A \pi_*^{(n-1)}(da|s') \int_S p(a,s',ds'') I_1^{(n-2)}(s'').$$

Similarly for $I_1^{(n-2)}(s')$, we get

$$I_1^{(n-2)}(s') = \int_A \pi_*^{(n-2)}(da|s') \left(r(a,s) - \tau \log \frac{\pi^{(n-2)}}{\hat{\pi}}\right) + \int_A \pi_*^{(n-2)}(da|s') \int_S p(a,s',ds'') I_1^{(n-3)}(s'').$$

Without loss of generality, suppose that $I_1^{(n-2)}(s') \leq I_1^{(n-1)}(s')$, then we multiply and divide $I_1^{(n-1)}(s')$ by $\pi_*^{(n-2)}$

$$I_1^{(n-1)}(s') = \int_A \pi_*^{(n-2)}(da|s') \frac{\pi_*^{(n-1)}}{\pi_*^{(n-2)}} \left(r(a,s) - \tau \log \frac{\pi^{(n-2)}}{\hat{\pi}} \right) + \int_A \pi_*^{(n-1)}(da|s') \int_S p(a,s',ds'') I_1^{(n-3)}(s'')$$

Then, using assumption 4 and bound B' (see 5) , one can derive:

$$\left| I_1^{(n-1)}(s') - I_1^{(n-2)}(s') \right| \leq \log(1+\frac{1}{k}) + \frac{(U + \tau\frac{\mu(\mathcal{A})}{e} + \tau\frac{\mu(\mathcal{A}^2)}{e})}{k}$$

$$+ \int_A \pi_*^{(n-1)} \int_S p(a,s',ds'')(1+\frac{1}{k}) \left| I_1^{(n-2)}(s'') - I_1^{(n-2)}(s'') \right|$$

We notice, that the same argument is repeating. We can deduce

$$\Rightarrow \left| I_1^{(n-1)}(s') - I_1^{(n-2)}(s') \right| \leq \log(1+\frac{1}{k}) + \frac{(U + \tau\frac{\mu(\mathcal{A})}{e} + \tau\frac{\mu(\mathcal{A}^2)}{e})}{k}$$

$$+(1+\frac{1}{k}) \left(\log(1+\frac{1}{k}) + \frac{(U + \tau\frac{\mu(\mathcal{A})}{e} + \tau\frac{\mu(\mathcal{A}^2)}{e})}{k} \right) + (1+\frac{1}{k})^2 \left(\log(1+\frac{1}{k}) + \frac{(U + \tau\frac{\mu(\mathcal{A})}{e} + \tau\frac{\mu(\mathcal{A}^2)}{e})}{k} \right) + \dots$$

We notice, that each term individual converge to zero, and number of terms is fixed and equal to $n - n_k$, then we conclude

$$\Rightarrow \left| I_1^{(n-1)}(s') - I_1^{(n-2)}(s') \right| \to 0, \text{ as } k \to \infty.$$

We comeback to the equation (20), we are left to show:

$$\left(I_2^{(n-1)}(s') - I_2^{(n-2)}(s') \right) \to Const \qquad \text{as } k \to \infty$$

Since, the space is assumed to be finite and MDP is irreducible, by ergodicity there exist $m \in \mathbb{N}$, such that for all $l > m$, the state S_l follows stationary law $S_l \sim m^\infty(\cdot)$. With m^∞, we note stationary measure. Without loss of generality, we can say $n - n_k > m$ and write:

$$I_2^{(n-2)}(s') = \mathbb{E}_{\pi_*} \left[\sum_{k=n-n_k}^{n-3} \left(R_k - \tau D_{KL}(\pi^{(n-2-k)} \| \hat{\pi})(S_k) \right) \mid S_0 = s' \right]$$

$$= \mathbb{E}_{\pi_*} \left[\sum_{k=0}^{n_k-3} \left(R_k - \tau D_{KL}(\pi^{(n_k-2-k)} \| \hat{\pi})(S_k) \right) \mid S_0 \sim m^{(n-n_k)} \right] = \int_S m^{(n-n_k)}(ds) V_*^{(n_k-2)}(s)$$

Similarly, we do for $I_2^{(n-1)}$:

$$I_2^{(n-1)}(s') = \int_S m^{(n-n_k)}(ds) V_*^{(n_k-1)}(s) = \int_S m^{(n-n_k)}(ds) \int_A \pi_*^{(n_k-1)}(da|s) \left(r(a,s) - \tau \log \frac{\pi_*^{(n_k-1)}}{\hat{\pi}} \right)$$
$$+ \int_S m^{(n-n_k+1)}(ds) V_*^{(n_k-2)}(s').$$

Then, by ergodicity, as $k \to \infty$ we have

$$\left| I_2^{(n-1)}(s') - I_2^{(n-2)}(s') \right| \to \int_S m^\infty(ds) \left(\mathbb{E}_{\pi_*^{(n)}}[r(A,s)] - \tau D_{KL}(\pi_*^{(n)} \| \hat{\pi})(s) \right).$$

Finally, we can deduce the proposition. For $n \to \infty$

$$\begin{aligned} Q_*^{(n)}(a,s) - Q_*^{(n-1)}(a,s) &= \int_S p(a,s,ds') \left(\left(I_1^{(n-1)}(s') - I_1^{(n-2)}(s') \right) + \left(I_2^{(n-1)}(s') - I_2^{(n-2)}(s') \right) \right) \\ &= \int_S p(a,s,ds') \left(\int_S m^\infty(ds) \left(\mathbb{E}_{\pi_*^{(n)}}[r(A,s)] - \tau D_{KL}(\pi_*^{(n)} \| \hat{\pi})(s) \right) \right) \\ \Rightarrow Q_*^{(n)}(a,s) - Q_*^{(n-1)}(a,s) &\to \int_S m^\infty(ds) \left(\mathbb{E}_{\pi_*^{(n)}}[r(A,s)] - \tau D_{KL}(\pi_*^{(n)} \| \hat{\pi})(s) \right) \end{aligned}$$

With that we conclude the proof.

□

An analysis of the observations presented in Figure (2) and Proposition 9 reveals a notable tendency: the longer the step-horizon, the more it resembles the structure of the preceding step. Essentially, with each subsequent horizon level, the need for freedom in expressing the optimal policy diminishes, thereby reducing the requirement for parameters. This observation is crucial and leads us to propose a "dynamical" parameters structure: $\#\vartheta^{(n)} \leq \#\vartheta^{(n-1)} \leq \ldots \leq \#\vartheta^{(1)}$, where we systematically reduce the number of trainable parameters. This approach significantly reduces the memory footprint of the algorithm.

In practical applications, gathering statistics similar to those depicted in Figure 2 can provide valuable insights into the optimal distribution of parameters. It might also be beneficial to adopt a semi-"shared" structure. This would mean using independent parameter sets for certain critical horizon-steps, while employing "shared" parametrization for others.

Lastly, it is intriguing to consider that the "shared" version of the algorithm can be seen as an intermediary between the REINFORCE and Matryoshka algorithms. By adjusting the number of parameters $\vartheta^{(i)}$, we can effectively strike a balance between the two designs. Setting $\#\vartheta^{(i)} = 0$ for $i > 2$ would, in essence, yield a REINFORCE algorithm augmented with entropy-regularized rewards.

3.3 Experiments

This section is devoted to evaluating various algorithms with Neural Network (NN) parametrizations in different environments. We focus on the following algorithms:

- *REIN* (Benchmark) - The standard REINFORCE algorithm, incorporating an ϵ-greedy policy. This serves as a benchmark algorithm representing stationary policies without the concept of step horizon.
- *MTR* (Old) - Similar to *REIN*, this algorithm is built around a singular policy. However, an extra input dimension is introduced to incorporate current horizon step information into the policy network $h : (a, s, i) \in \mathcal{A} \times \mathcal{S} \times \mathbb{N} \to (0, 1)$, making the policy adapt to step horizon dynamics. This can be viewed as an initial realization of the Matryoshka algorithm, as detailed in Francois's work (see page 15 [3]).
- *Original* (Old) - A direct implementation of the Matryoshka principle, employing a set of independent Neural Networks for each step horizon.
- *MtrNet* (New) - A novel implementation of the Matryoshka principle, utilizing shared parametrization with the S-MPG update (see 2.1).
- *MtrNet Light* (New) - Similar in architecture to *MtrNet*, but employs the Light MPG update during training (see 2.3).
- *MtrNet Light Dyn.* (New) - Implements the concept of dynamically decreasing the number of trainable parameters, using the Light MPG update.

These algorithms are graded based on their ability to perform non-stationary tasks, as follows:

$$\text{REIN} < \text{MTR} < \text{MtrNet} < \text{MtrNet Light Dyn.} < \text{MtrNet Light} < \text{Original}$$

From left to right, this gradation indicates the increasing capability of the algorithm to adapt to non-stationary environments. (To get more info on the tested models, see Appendix A.2)

To validate our hypotheses, we have selected four distinct environments for testing:

- *Cart Pole*: Suitable for stationary policies, where the optimal action is not dependent on the horizon step. We anticipate good performance from *MTR* and *REIN*. However, due to the high maximal horizon, Matryoshka-type algorithms might face challenges due to restricted training epochs.
- *Toy*: Specifically designed to test the efficacy of Matryoshka algorithms. Hence, we expect the Original and *MtrNet Light* versions to excel in this setting, in contrast to the *REIN* algorithm.
- *Maze and Lake*: These grid scenario environments are expected to be compatible with all proposed algorithms. The primary challenge lies in avoiding local optima, such as oscillating movements to evade pitfalls. Where for our test, we choose the base-line policy $\bar{\pi}$ to be an uniform distribution. Therefore algorithms with a higher temperature parameter τ are predicted to perform better in these scenarios.

For detailed descriptions of the environments used in our experiments, readers are referred to A.1.

In the subsequent sections, we present tables displaying the performance metrics of different algorithms in these environments. As metric, we simply use the average of cumulative non-regularized reward, and maximum cumulative reward obtained by algorithms. The tables include additional details about the tests, such as the fixed number of epochs used for all algorithms.

Prior to delving into the results analysis, it is essential to highlight a key aspect of our testing methodology. In these experiments, we refrained from selecting or tuning hyperparameters. The same learning rates and τ values were employed across all tested algorithms, as elaborated in Appendix B. While further tuning might improve the outcomes in each scenario, the primary objective of these tests was to evaluate the intrinsic performance of the algorithms without reliance on hyperparameter optimization. This approach provides a more authentic assessment of each algorithm's capability in addressing specific tasks. Intriguingly, the results obtained align closely with our initial hypotheses and expectations.

Beginning with table 1, we observe the superiority of the *Original* and *MtrNet Light* algorithms in achieving the highest scores, demonstrating commendable stability across both 4 and 7 maximal horizons. As predicted, the stationary algorithms *MTR* and *REIN* exhibited inferior performance, underscoring the effectiveness of fixed horizon algorithms in these contexts.

Subsequently, we examine the Cart Pole game, which is particularly suited for *REIN*, *MTR*, and likely *MtrNet* (S-MPG update). These three algorithms exhibited the best performance, corroborating our belief that the S-MPG update can adeptly handle highly stationary strategies, akin to those required in the Cart Pole game. On the other hand, Light-MPG updates may necessitate a greater number of epochs to achieve comparable results.

Regarding the last two environments, *Lake* and *Maze* (refer to table 2), they are suitable for all algorithms, yet they demand an effective exploration phase to converge on the optimal solution. Hence, algorithms with a higher temperature factor τ are expected to excel, a hypothesis that holds true except for the Original algorithm in the Maze environment. This outcome also illustrates that entropy regularization is an efficient method to govern the exploration process.

In summary, the comprehensive analysis of the aggregated results from our series of experiments highlights a notable outcome: the newly proposed algorithm, *Mtr Light Dyn.*, consistently achieves the highest performance scores across a variety of test environments. It demonstrates remarkable proficiency in addressing both stationary and non-stationary games. This underscores the practical efficacy of our algorithm, outperforming traditional approaches such as *REIN*, the *Original* Matryoshka algorithm, and even the *MTR* proposed by Francois, which should adapt well to non-stationary solutions. Its ability to seamlessly adapt to the dynamic requirements of different gaming scenarios underscores its practical efficacy and marks a significant advancement in the field of reinforcement learning.

Environment	Toy, horizon 4					
Learning rate	0.001				0.01	
τ	0.6		1.2		0.6	
Agents/Stats	mean	max	mean	max	mean	max
Rein	0.47	0.52	0.47	0.52	0.47	0.53
MTR	0.42	0.45	0.44	0.48	0.44	0.50
MtrNet	0.53	0.59	0.53	0.57	0.63	0.79
Original	0.56	0.67	0.52	0.65	0.64	0.96
MtrNet-Light	0.69	0.77	0.50	0.57	0.59	0.78
MtrNet-Light-Dyn.	0.60	0.72	0.63	0.65	0.59	0.71

Environment	Toy, horizon 7			
Learning rate	0.001			
τ	0.6		0.9	
Agents/Stats	mean	max	mean	max
Rein	0.116	0.205	0.116	0.205
MTR	0.121	0.180	0.132	0.155
MtrNet	0.384	0.528	0.349	0.404
Original	0.381	0.398	0.378	0.405
MtrNet-Light	0.478	0.574	0.430	0.504
MtrNet-Light-Dyn.	0.449	0.516	0.504	0.604

Environment	Cart, horizon 50							
Learning rate	0.0001				0.001			
τ	0.7		0.9		0.7		0.9	
Agents/Stats	mean	max	mean	max	mean	max	mean	max
Rein	49.02	49.43	49.02	49.43	43.86	50.00	43.86	50.00
MTR	49.60	49.96	48.50	50.00	49.95	50.00	45.96	50.00
MtrNet	46.05	50.00	49.56	50.00	27.00	30.66	36.12	49.97
Original	27.83	33.94	29.44	36.87	39.32	47.11	37.61	42.86
MtrNet-Light-Dyn.	18.77	50.00	37.13	47.51	37.27	49.80	35.24	49.93

Environment	Cart, horizon 100			
Learning rate	0.0001			
τ	0.7		0.9	
Agents/Stats	mean	max	mean	max
Rein	89.35	99.66	89.35	99.66
MTR	79.34	100	100	100
MtrNet	87.75	99.4	95.58	99.4
MtrNet-Light-Dyn.	30.10	61.56	31.36	71.7

Table 1: Toy. Training: number of Epoch = 1000. Number of games per epoch = 25. Repeated 4-7 times for each learning rate and τ. Cart Pole. Training: number of Epoch = 1000. Number of games per epoch = 3. Repeated 3-5 times for each learning rate and τ. Max reward = 50 and 100 respectively.

Environment	Maze, grid 5x5, horizon 7				Lake, grid 4x4, horizon Horizon: 10			
Learning rate	0.001				0.001			
τ	0.5		1		0.9		1.5	
Agents/Stats	mean	max	mean	max	mean	max	mean	max
Rein	2.87	5.75	2.87	5.75	24.95	24.99	24.95	24.99
MTR	-1.72	2.80	4.62.	5.76	21.80	25.00	25.00	25.00
MtrNet	-3.28	-2.55	5,60	5,64	25.00	25.00	18.60	25.00
Original	3.32	4.83	2.56	3.94	9.25	25.00	15.71	24.00
MtrNet-Light	4.94	5.68	4.65	5.57	17.25	25.00	25.00	25.00
MtrNet-Light-Dyn.	5.55	5.77	5.18	5.8	25.00	25.00	25.00	25.00

Table 2: Maze. Training: number of Epoch = 300. Number of games per epoch = 25. Lake. Training: number of Epoch = 600. Number of games per epoch = 25. Repeated 4-7 times for each learning rate and τ. Maze max reward = 25.

Optimal Solution Test

Mirroring the experiments conducted in the work "Matryoshka Policy Gradient for Entropy-Regularized RL: Convergence and Global Optimality" by Francois (see Figure 1), our study aims to validate our theoretical findings in a "shared" settings context. We focus on testing the optimality solution theorem, which is expected to be applicable for Light MPG as per Theorem 7. Unlike the original experiments that were based on log-linear parametrization, our approach extends to Neural MPG, thus incorporating Neural Network parametrization. This endeavor also serves as a practical verification of the discussions on optimal solutions in Neural MPG, as delineated in Section 3.1 (Key Difference), and encapsulated in Corollary 1 of Francois's work (see page 10 in [3]). The corollary posits that for a policy $\pi_t \in \mathcal{P}_n$ is parametrized by neural networks and updated ideally as $\theta_{t+1} - \theta_t = \eta \nabla_\theta J_n(\pi_t)$ with a sufficiently small learning rate $\eta > 0$, then at the policy's critical point π_∞, it holds that:

$$\pi_\infty = \underset{\pi \in P_n^{\Theta_\infty}}{\operatorname{argmax}} J_n(\pi) = \underset{\pi \in P_n^{\Sigma_\infty}}{\operatorname{argmax}} J_n(\pi). \tag{21}$$

Specifically, when a policy $\pi_* \in \mathcal{P}_n^{\Sigma_\infty}$, it is expected that $\pi_\infty = \pi_*$. This implies that if the optimal solution is expressible through the conjugate kernel (or equivalently, the ntk kernel), i.e., for all $i \in [n], Q_*^{(i)} \in \mathcal{H}_{\Sigma_\infty}$ (or in a shared scenario: $Q_*^{(i)} - h_\theta^{(i-1)} \in \mathcal{H}_{\Sigma_\infty}$), we are guaranteed to reach the optimal solution at the critical point. Our objective is to ascertain if, in a realizable case, we can achieve the optimal regularized solution for a specified τ.

For this proof of concept, we revisit our *Toy* environment with an action space comprising two actions $\{+1, +2\}$ and a maximal horizon of $n = 7$. Rewards for each action/state pair $r_{a,s}$ are assigned accordingly. Utilizing Proposition 1, we estimate V_* by recursively calculating for each horizon:

$$V_*^{(i)}(s) = \log \mathbb{E}_{\bar{\pi}} \left[\exp \left(Q_*^{(i)}(A, s)/\tau \right) \right]$$
$$Q_*^{(i+1)}(a, s) = r_{a,s} + \int_S p(a, s, ds') V_*^{(i)}(s),$$

where $Q_*^{(1)}(a, s) = r_{a,s}$ and the transition probability is deterministic.

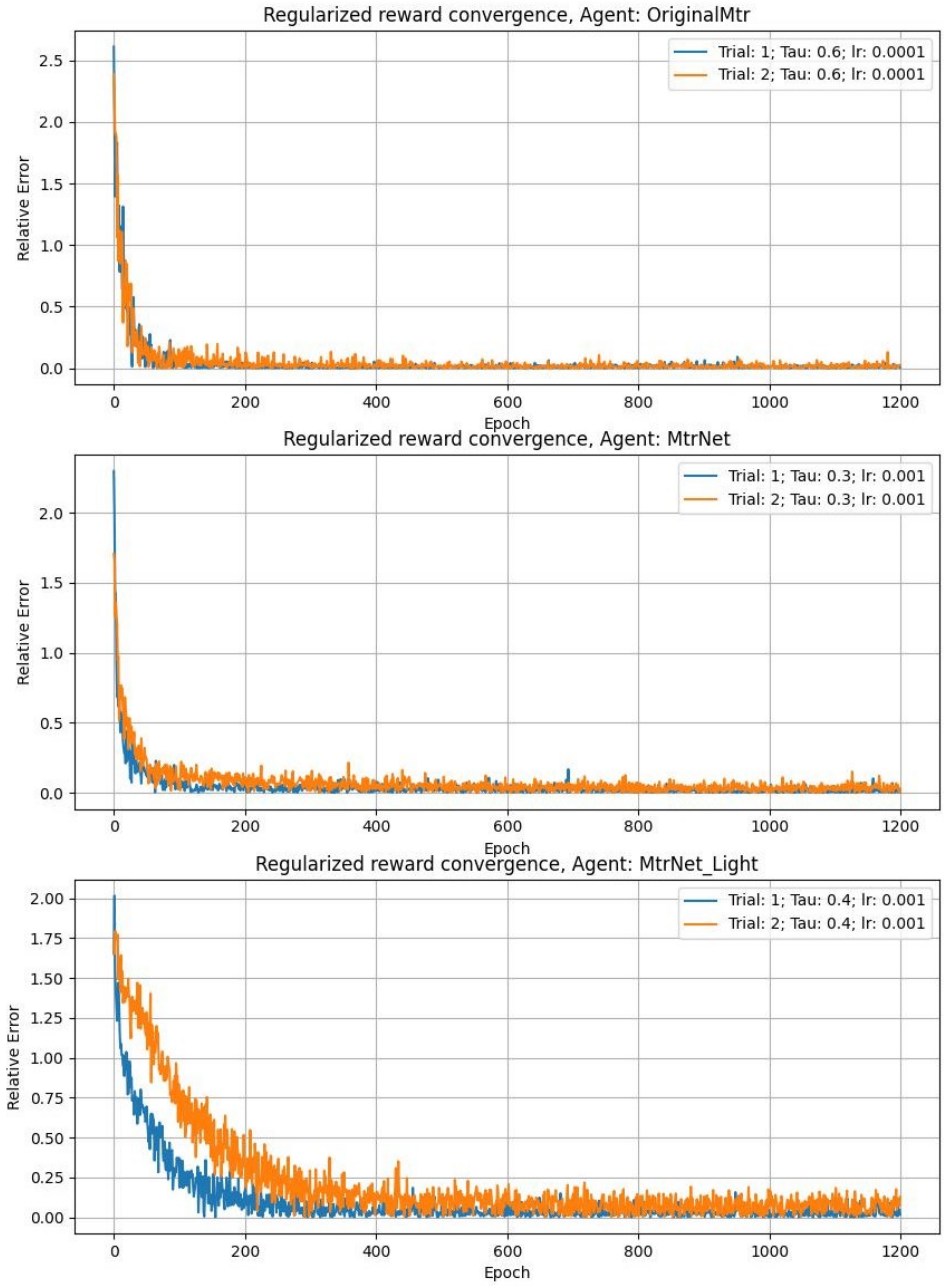


Figure 3: Testing convergence to the optimal solution in the 'Toy' environment with various architectures.

The optimal objective is calculated as $J_*(\tau) = \int_S \vartheta(ds) V_*^{(7)}(s)$, with the initial state distribution ϑ being uniformly distributed between states [0] and [2]. Thus, the optimal regularized reward for τ is $J_* = (V_*^{(7)}(0) + V_*^{(7)}(2))/2$. To verify that the CK (Conjugate Kernel) contains the optimal solution, we leverage the fact that our action space has only two actions, necessitating just one degree of freedom per state to encapsulate the optimal policy. If the CK (respectively NTK) achieves full rank (equal to the number of actions times the number of states) by the end of training, we are assured of being in a realizable case, thereby enabling the attainment of the

optimal solution.

In our test presented in Figure 3, we evaluated the *Original*, *MtrNet*, and *MtrNet Light* architectures for their convergence to the optimal solution. Each trial in these tests achieved full rank for NTK and CK kernels. The error was measured as the difference between the average cumulative regularized error over 80 trials per epoch achieved by the algorithm and the optimal solution $J_*(\tau)$. Notably, for all three architectures and each trial at full rank, we observed the desired convergence, validating our approach and supporting Corollary 1 in the realizable case.

4 Conclusion

Remark: This section adopts a unique structure, revisiting each significant point and topic covered in this work. It includes discussions on these topics, emphasizing potential areas for further investigation based on our findings.

Initially, we introduced the novel concept of "Shared" Matryoshka, characterized by the sharing of certain parameters between two adjacent policies. Subsequently, we developed two distinct parameter updates: S-MPG (Shared Matryoshka Gradient Update) and Light-MPG (Light Matryoshka Gradient Update). We demonstrated (see Proposition 8) that under the Realizable assumption 2.4, the critical point of the Light objective also serves as a critical point for the S-MPG algorithm, indicating that they optimize the same objective function. An important point, not previously discussed, concerns the applicability of these results in the absence of the Realizable assumption. It turns out that there is a lot of chances that would not hold. As per Proposition 7, the critical point of the Light objective satisfies the consistency property. Specifically, the preference function $h_\theta^{(i)}$ converges to the orthogonal projection of $Q_\theta^{(i)}$ into the RKHS $\mathcal{H}_{\Theta^{(i)}}$. Whereas, the kernel $\Theta^{(i)}$ is different for two update, and in particular, one can easily show that $\mathcal{H}^{Light}_{\Theta^{(i)}} \subset \mathcal{H}^{S-MPG}_{\Theta^{(i)}}$. However, the RKHS for S-MPG is generally larger for each step horizon, though it utilizes the same resources for optimizing two adjacent policies. Consequently, if the Realizable assumption does not hold for a particular horizon, S-MPG might offer better decisions for that horizon, but this does not guarantee superior overall performance due to potential loss of expressibility for preceding step horizons. Interestingly, there might be a scenario where Proposition 8 holds without the Realizable assumption, especially in highly stationary games like Cart Pole, where the optimal decision remains consistent across step horizons. This would imply that RKHS of different horizons are the same $\mathcal{H}_{\Theta^{(i)}} = \mathcal{H}_{\Theta^{(i-1)}}$, therefore the both updates share the same RKHS $\mathcal{H}^{Light}_{\Theta^{(i)}} = \mathcal{H}^{S-MPG}_{\Theta^{(i)}}$, leading to the same projection. However, a theoretical underpinning for this scenario requires a consistency property for S-MPG, which is currently lacking in our framework. This notion presents an intriguing direction for future research and practical testing. Unfortunately, due to time constraints towards the project's end, this idea remains unexplored.

Let us now continue our recapitulation. In the theoretical framework, we determined the learning rate η that ensures the convergence of S-MPG (see Theorem 5). Additionally, we mirrored the work of Francois to establish that Light MPG possesses a unique critical point that adheres to the consistency property (see Theorem 7). Particularly, under the Realizable assumption, this critical point aligns with the optimal policy. Despite these advancements, our theoretical framework concludes with some unresolved gaps, as discussed in the section "Complete Framework" (see 2.7). We proposed potential solutions to bridge these gaps in the subsection "Roadmap for a Complete Framework", leaving them as open questions for future study.

Moving forward to the topic of "Neural MPG", we delve into the integration of the shared Matryoshka concept with Neural Network parametrization, alongside discussions about the conjugate kernel and its role in representing the optimal solution. An area ripe for further exploration is determining when and why the RKHS associated with the Conjugate kernel can or cannot encapsulate the optimal solution. Notably, the RKHS pertaining to the Conjugate Kernel ($\mathcal{H}_{\Sigma^{(i)}}$) is a subspace of the RKHS linked to the Neural Tangent Kernel ($\mathcal{H}_{\Theta^{(i)}}$), as highlighted in Corollary 1

of Francois's work (refer to [3]). It is established that

$$\pi_\infty = \underset{\pi \in P_n^{\Theta_\infty}}{\text{argmax}} J_n(\pi),$$

indicating that h_{θ_∞} resides in $\mathcal{H}_{\Theta_\infty^{(i)}}$. It is interesting to see, when and why the RKHS related to the Conjugate kernel, can or can not express the optimal solution. A promising approach is to analyze the NTK kernel, given its influence over the CK kernel. Focusing on the effective rank[5] of the NTK kernel could reveal why certain directions collapse (eigenvalues approach zero) during training. This investigation could extend to examining how the effective rank is affected by different initialization processes and nonlinear adjustments like batch normalization. For instance, exploring the impact of various initialization regimes—ordered, chaotic, or edge of chaos—could yield significant insights (see Chapter 5 in [5] for a detailed discussion on the effects of these regimes). It is important to note that this type of analysis was initially planned. However, due to the intricate nature and depth of exploration required for the "Shared" Matryoshka concept, it became the focal point of the project, precluding the exploration of these additional aspects.

Finlay we arrive to the section 3.2, where we highlight the distinct advantages of the "Shared" Matryoshka approach in comparison to the original design. This section emphasized that even in non-step stationary environments like "Toy," the environment becomes effectively stationary for sufficiently large horizons. This characteristic allows us to leverage the inherent structure of the "Shared" Matryoshka model. Subsequently, in the final section of the project, we explored various practical implementations of the Matryoshka algorithm, along with the standard REINFORCE model. We introduced a conceptual framework with two poles: REINFORCE, representing non-fixed horizon designs, and the Original Matryoshka algorithm. Between these poles, we positioned different "Shared" designs based on their performance and expected scores in the tested environments. Our empirical evaluations confirmed our theoretical predictions, particularly highlighting that our primary model, *MtrNet Light Dyn.*, consistently outperformed other algorithms in average scores. This achievement aligns with one of our project's objectives: to develop a practical implementation of Neural MPG.

In the "Optimal Solution Test" subsection, we validated the ability of our proposed designs to converge to the optimal policy, essentially proving the concepts of Theorem 6 and Corollary 1 (see eq. 21) for both Light MPG and Original designs. Originally, we intended to include additional results related to the consistency projection property of both Light MPG and Original Matryoshka in Neural Network parametrization. We aimed to demonstrate that in non-realizable cases (where the Conjugate kernel is not full rank), the obtained policy π_θ converges to a policy $\pi_\cdot$ (see 2), driven by the projection of the action-value function $Q_\theta^{(i)}$ onto the $\mathcal{H}_{\Sigma_\infty^{(i)}}$ RKHS. This RKHS is constructed as a span of the eigenvectors of the CK kernel associated with its effective rank, represented as $Q_\cdot^{(i)}$. However, due to technical challenges, this experiment did not yield the anticipated results and is currently ongoing.

With this said, we conclude the final chapter of the Master's project.

□

[5]The effective rank involves performing a spectral decomposition of the NTK kernel and discarding vectors associated with small eigenvalues, as these directions are not learned and do not affect the CK. The remaining vectors determine the effective rank.

A Appendix

A.1 Description of Environments for Reinforcement Learning Experiments

In this appendix, we provide detailed descriptions of the four environments used for testing in the context of Reinforcement Learning experiments. Each environment offers unique challenges and characteristics that make them suitable for evaluating different RL algorithms.

Cart-Pole Environment

The Cart-Pole environment is a classic benchmark in Reinforcement Learning. It involves balancing a pole on a moving cart.

Description:

- **State Space:** The state space consists of four continuous variables representing the position and velocity of the cart and pole.
- **Action Space:** The agent can take two discrete actions: push the cart left or right.
- **Rewards:** The standard reward function provides a reward of +1 for each time step the pole remains upright. If the pole falls or the cart moves too far from the center, the episode terminates with a reward of 0.

Maze Environment

The Maze environment is a grid-world scenario with the goal of navigating through a maze to reach a reward. It offers a discrete state and action space and is commonly used for testing RL algorithms in a discrete setting.

Description:

- **Grid Size:** The maze is a 5x5 grid with walls blocking certain paths.
- **Initial States:** The agent can start from one of three initial positions: (0,0), (0,4), or (4,0).
- **Goal:** The reward is positioned at (4,4) with a reward of +10. All other states provide a reward of -1.
- **Obstacle:** A wall is present at position (2,2), creating a barrier that the agent must navigate around.

Lake Environment

The Lake environment is another grid-world scenario with a focus on navigation and risk. The agent must avoid falling into holes while aiming to reach a rewarding destination. Additionally, the environment is slippery, introducing stochasticity into the agent's actions.

Description:

- **Grid Size:** The Lake is a 4x4 grid.

- **Initial State:** The agent starts at position (0,0).
- **Holes:** There are four holes at positions (1,1), (0,3), (3,1), and (3,2). Falling into a hole terminates the episode with a reward of -5.
- **Reward:** A high-reward state is located at (3,3) with a reward of +30. All other states provide a reward of -1 per time step.
- **Slippery:** The environment is slippery, meaning that actions may not always lead to the intended outcome, introducing randomness into the agent's movements.

Toy Environment

We have already introduced the general mechanics of this game (See 1.1). Then, we directly, start with description phase.
Description:

- **State Space:** $\mathcal{S} = \{0, 1, 2 \dots 6\}$.
- **Action Space:** $\mathcal{A} = \{+1, +2\}$.
- **Initial State:** The agent starts at random position on the state space. For the "Optimal Solution Test", we go to only two random state $\mathcal{S}_0 = \{0, 2\}$, with 50% chance for each.
- **Reward:** We contact rewards similarly to the way it is done in Francois's work (see Appendix F.1 in [3]). We define an orthonormal basis (in $\ell^2(\mathcal{A} \times \mathcal{S})$) of the space of functions $\{f : \mathcal{A} \times \mathcal{S} \to \mathbb{R} : f(1, s) + f(2, s) = 0, \forall s \in \mathcal{S}\}$. Then we set a scalar vector $\alpha = [0.1, 0.2, -0.1, -0.2, 0.1, -0.3, 0.8]$ and we define reward, as $r_{a,s} = \left(\sum_{i=1}^{7} \alpha_i e_i\right)(a, s)$. In such settings, we are sure that the Conjugate kernel requires to be the full rank, in order to express the optimal solution, for the first step horizon.

A.2 Neural models

This section details the implementation of Neural Network parametrization for various algorithms, with a focus on the technical aspects specific to the PyTorch framework.

As previously introduced in Section 3.3 , we have evaluated several design architectures: *REIN*, *MTR*, *Original*, *MtrNet*, *MtrNet Light*, and *MtrNet Light Dyn*. It is noteworthy that the last three differ only in their update mechanisms and in the number of trainable parameters, leading to essentially four distinct architectural types. We will first discuss their design and subsequently delve into the initialization process for our experiments.

Design

Our Neural Networks are designed to parametrize preference functions $h_\theta^{(i)}(a, s)$ for $i \in [n]$, which are essential in estimating the action value $Q^{(i)}(a, s)$ for a given input pair $(a, s) \in \mathcal{A} \times \mathcal{S}$. A common feature across all four designs is that the input layer $\alpha_0^{(i)}(x_{in})$ only incorporates state data s, excluding action data a. For the i-th step horizon, the output layer $h_\theta^{(i)}(x_{in})$ has a dimensionality

equal to the number of actions, thereby $h_\theta^{(i)}(x_{in}) \in \mathbb{R}^{|A|}$, with each dimension representing the preference function for a specific action: $h_\theta^{(i)}(s) = [h_\theta^{(i)}(0, s), h_\theta^{(i)}(1, s), \ldots]$. This design approach is typical in Reinforcement Learning tasks with discrete and small action spaces, as is the case in our testing environments. However, for continuous action spaces, both state and action variables would need to be included in the input.

Upon obtaining the output layer $h_\theta(s)$, we apply a softmax function with a given temperature factor τ and a baseline policy $\hat{\pi}$, resulting in a probability vector $[\pi_\theta(0|s), \pi_\theta(1|s), \ldots]$. Let's examine how we arrive at the preference vectors $h_\theta^{(i)}$ using our four distinct concepts.

REIN and *MTR*. Both these algorithms employ a single Neural Network to estimate preferences for each step horizon, as illustrated in the diagram below. They are stationary regarding step horizons, but for *MTR*, we introduce an additional input dimension to incorporate information about the current horizon step, normalized as $(n - step)/n$.

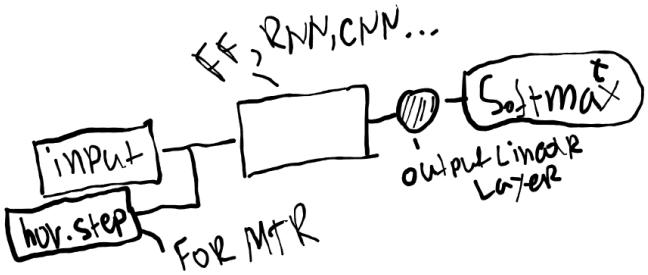


Figure 4: Neural network design for *REIN* and *MTR* algorithms.

The rectangle in the middle of the NN scheme (see 4) represents the hidden layers, which can embody various architectures such as Feed Forward (FF), Residual (RNN), or Convolutional (CNN) neural networks. For our experiments, we opted for a two-layer FF neural network with ReLU activations. The small black disk symbolizes the output layer h_θ, responsible for estimating preferences.

MtrNet. In this design, each step horizon has a separate rectangular block and disk, indicating individual hidden layers and output layers, respectively. The output layer of one step horizon is aggregated with that of the next, as depicted in the following diagram

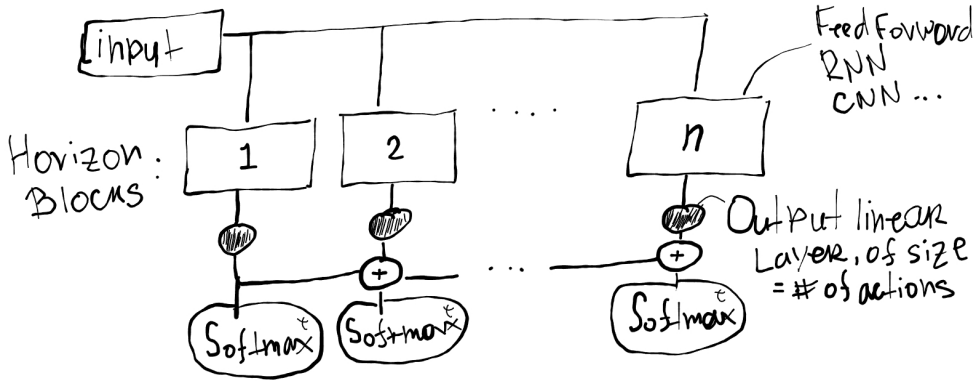


Figure 5: Design of the *MtrNet* architecture showcasing the summation of adjacent output layers.

Each rectangular block represents an independent hidden layer with its own set of trainable parameters. The adjacent output layers are combined through summation, effectively emulating the shared Neural Matryoshka algorithm as outlined in Section 3.1, "Neural MPG", sub-section "Shared NN" (see 3.1). In the case of *MtrNet Light Dyn.*, the number of trainable parameters in the rectangular blocks is progressively reduced with increasing step horizon index.

Original. The design of the *Original* algorithm closely mirrors that of *MtrNet*, with one key distinction: the interconnections between output layers are removed. This results in a structure comprising n independent neural networks.

Initialization Methodology

In our study, we adopt a cutting-edge approach for initializing Neural Networks, known as "ntk parametrization." This method has garnered acclaim for enhancing the learning efficiency and ensuring stable backpropagation across various models.

For each i-th step horizon in our network, the initialization of the l-th hidden layer is structured as follows:

$$\alpha_{l+1}^{(i)}(x_{\text{input}}) = \sigma\left(W_l^{(i)} \cdot \alpha_l^{(i-1)}(x_{\text{input}}) + b_l^{(i)}\right) \text{ and } \begin{cases} W_{l,i,j}^{(i)} & = \frac{\sigma_\omega}{\sqrt{n_l}}\omega_{l,ij}^{(i)} \\ b_{l,j}^{(i)} & = \sigma_b \beta_{l,j}^{(i)} \end{cases},$$

with n_l denoting the number of neurons in the layer. The weights $\omega_{l,i,j}^{(i)}$ and biases $\beta_{l,j}^{(i)}$ are trainable parameters, each independently drawn from a Gaussian distribution $\omega_{l,ij}^{(i)}, \beta_{l,j}^{(i)} \sim \mathcal{N}(0,1)$.

The selection of hyperparameters, especially in ntk parametrization, plays a crucial role in optimizing the learning process and maintaining backpropagation stability. It has been observed that a specific set of parameters, which lie on the 'Edge of Chaos' curve, can significantly enhance the network's performance (see [4] for an in-depth analysis of EOC curves). The 'Edge of Chaos' curve varies with different architectures. For our configuration, which involves a Feed Forward (FF) network with ReLU activation functions, the optimal hyperparameters are typically set as $(\sigma_\omega, \sigma_\beta) = (\sqrt{2}, 0)$. In our experiments, all hidden layers of the networks are initialized according to this ntk parametrization, precisely calibrated to operate at the edge of chaos, thereby optimizing network performance and stability.

Stochastic Gradient Update in Model-Free RL

In our model-free RL framework, we adopt a standard training procedure involving batch gameplay. During these sessions, the agent's actions and state transitions are recorded. The optimization of the algorithm is then carried out using a Gradient Policy approach, specifically focusing on the data collected during gameplay. Notably, our method does not attempt to estimate state distribution or transition probabilities. While this approach does not yield the ideal update represented by $\theta_{t+1} - \theta_t = \eta \nabla_\theta J_n(\pi_t)$, it effectively implements Stochastic Gradient Ascent (SGA). Crucially, the design of our Matryoshka Policy Gradient update ensures that, in expectation, the desired update is achieved (see Theorem 2, Proposition 4, Proposition 5). Consequently, with a sufficiently large batch size, our method serves as an effective approximation of the ideal update, aligning closely with the theoretical framework.

B Reproducibility of Results

Ensuring reproducibility of results is paramount in scientific research. To facilitate this, we have made our experiment procedures transparent and reproducible. An open-source GitHub repository is available, containing the Python codebase that forms a comprehensive framework with a wide range of APIs, simplifying and expediting work with Matryoshka algorithms. The repository includes a README file detailing framework usage. This section will focus on the technical setup for experiments within the framework.

Setting Up Hyperparameters for Training

All training and testing processes are conducted in the main.ipynb Jupyter notebook. Initially, we establish the global parameters for our experiments, applying them uniformly across all agents.

```
"""
        SETUP ENVIRONMENT
"""

GAME_NAME = "Lake"
HORIZON = 10 #fixed horizon

"""REINFORCE"""
HIDDEN_DIM_REIN = 24
EPSILON = 0.15 # for epsilon greedy policy
"""MTR"""
HIDDEN_DIM_MTR = HIDDEN_DIM_REIN
"""MtrNet"""
HIDDEN_DIM_RESNET = 10
#MtrNet with dynamically changing parameters number per layer
INIT_HIDDEN_LAYER = 20

"""Training"""
TAU = 0.9
LEARNING_RATE = 0.001
NUM_EPOCHES = 600
N_EPISODES = 25 # number of episodes per epoch.

"""Testing"""
NUM_TEST_EPISODES = 80

"""DYNAMICAL TAU AND LR"""
PATIENCE = 20 # for dynamical tau and lr
TAU_END = TAU # for dynamical tau
LR_END = LEARNING_RATE / 100 # for dynamical learning rate
```

The selection of the appropriate **GAME-NAME** and **HORIZON** is essential for each tested environment. We then tailor the Neural Network (NN) parametrization for each design. For our tests, all neural network blocks (as shown in 5) are configured as two-layer Feed Forward (FF) networks with ReLU activation. The number of neurons per layer varies across different designs but remains constant for each environment. For *REIN* and *MTR*, we utilized 24 neurons per layer, whereas for *MtrNet*, we employed 10 neurons per layer. The **INIT-HIDDEN-LAYER** variable

determines the neuron count for *MtrNet Light Dyn.*, where the number dynamically decreases for higher step horizons. The initialization process for this configuration is described as follows:

```
int(n_actions + (init_hidden_dim - n_actions) * (horizon - step_hor) /
    horizon)
```

LEARNING-RATE, **TAU**, **NUM-EPOCHES**, and **N-EPISODES** are set in accordance with the details presented in the captions of Tables 1 and 2. It is important to note that the learning rate and τ are dynamically adjusted during training. The learning rate smoothly decreases until it reaches **LR-END**, set to one-hundredth of the initial rate. In contrast, τ remains constant (**TAU-END = TAU**) throughout the training to maintain consistency and highlight the impact of the temperature factor.

Training and Testing loops

This subsection details specific aspects of our training and testing processes, tailored to the needs of different testing environments. The general structure of our training/testing cell is outlined as follows:

```
"""Training}"""
agent.policy.ntk_init()
_,_,data = train_agent(agent, env, num_epoches=NUM_EPOCHES,
    n_episodes=N_EPISODES,game_name=GAME_NAME, tau_end=TAU_END ,
    lr_end=LR_END, patience=PATIENCE, clip_grad=False, testing=False)
"""Testing}"""
_, data_ = test_agent(agent, env, GAME_NAME,
    n_episodes=NUM_TEST_EPISODES, tau = 0.05)
"""PUSH DATA"""
df = push(df, data, data_)
```

Initially, we initialize the trainable parameters of the agent using the **.ntk-init()** method. By default, this implements NTK initialization in the Edge of Chaos regime, but specific parameters σ_ω and σ_β can be manually set (refer to A.2 for details). In the case of the *Toy* environment, for instance, a "Zero" initialization strategy is employed ($\sigma_\omega = 0, \sigma_\beta = 0$).

During the training loop, particular attention is given to the **clip-grad** variable. This parameter was set to **True** exclusively for the Cart Pole environment to stabilize training by truncating gradients exceeding an absolute value of 8. This adjustment is crucial as the reward in the Cart Pole game increases rapidly for higher horizons. In the testing loop, the primary adjustable variable is τ, which is typically set to 0.05 in our tests.

For the "Optimal Solution Test", the setup generally mirrors the standard procedure with a few modifications. After initializing the environment, its testing attribute is set to **True** (**env.testing = True**). This adjustment results in two potential initial states: a 50% chance of starting at $S_0 = 0$ and a 50% chance at $S_1 = 1$, rather than a random selection from all states. This controlled setting aids in accurately estimating the true optimal reward. Additionally, in the training loop, we set the variable **testing = True**. When activated, this mode adds information about whether the CK has full effective rank to our data frame **df**. It's important to note that this testing mode is specifically designed for the *Toy* environment with **env.testing = True** enabled.

References

[1] Alekh Agarwal, Sham M. Kakade, Jason D. Lee, and Gaurav Mahajan. On the theory of policy gradient methods: Optimality, approximation, and distribution shift, 2020.

[2] Bolin Gao and Lacra Pavel. On the properties of the softmax function with application in game theory and reinforcement learning, 2018.

[3] François Ged and Maria Han Veiga. Matryoshka policy gradient for entropy-regularized rl: Convergence and global optimality, 2023.

[4] Soufiane Hayou, Arnaud Doucet, and Judith Rousseau. On the impact of the activation function on deep neural networks training, 2019.

[5] Arthur Ulysse Jacot-Guillarmod. Theory of deep learning: Neural tangent kernel and beyond, 2020.

[6] Frank Nielsen. An elementary introduction to information geometry. *Entropy*, 22(10):1100, September 2020.

[7] Simon Weissmann Sara Klein and Leif Döring. Beyond stationarity: Convergence analysis of stochastic softmax policy gradient methods. In *Submitted to The Twelfth International Conference on Learning Representations*, 2023. under review.

[8] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018.

[9] Kaiqing Zhang, Alec Koppel, Hao Zhu, and Tamer Başar. Global convergence of policy gradient methods to (almost) locally optimal policies, 2020.