Non-Stationary Approximate Modified Policy Iteration

Boris Lesner BORIS.LESNER.DATEXIM@GMAIL.COM
Bruno Scherrer BRUNO.SCHERRER@INRIA.FR
Inria, Villers-lès-Nancy, F-54600, France
Université de Lorraine, LORIA, UMR 7503, Vandœuvre-lès-Nancy, F-54506, France

Abstract

We consider the infinite-horizon γ-discounted optimal control problem formalized by Markov Decision Processes. Running any instance of Modified Policy Iteration—a family of algorithms that can interpolate between Value and Policy Iteration—with an error ϵ at each iteration is known to lead to stationary policies that are at least $\frac{2\gamma\epsilon}{(1-\gamma)^2}$-optimal. Variations of Value and Policy Iteration, that build ℓ-periodic non-stationary policies, have recently been shown to display a better $\frac{2\gamma\epsilon}{(1-\gamma)(1-\gamma^\ell)}$-optimality guarantee. We describe a new algorithmic scheme, Non-Stationary Modified Policy Iteration, a family of algorithms parameterized by two integers $m \geq 0$ and $\ell \geq 1$ that generalizes all the above mentionned algorithms. While m allows one to interpolate between Value-Iteration-style and Policy-Iteration-style updates, ℓ specifies the period of the non-stationary policy that is output. We show that this new family of algorithms also enjoys the improved $\frac{2\gamma\epsilon}{(1-\gamma)(1-\gamma^\ell)}$-optimality guarantee. Perhaps more importantly, we show, by exhibiting an original problem instance, that this guarantee is tight for all m and ℓ; this tightness was to our knowledge only known in two specific cases, Value Iteration $(m = 0, \ell = 1)$ and Policy Iteration $(m = \infty, \ell = 1)$.

1. Introduction

Dynamic Programming (DP) is an elegant approach for addressing γ-discounted infinite-horizon optimal control problems formalized as Markov Decision Processes (MDP) (Puterman, 1994). The two most well-known DP algorithms in this framework are Value Iteration (VI) and Policy Iteration (PI). While the former has typically lighter iterations, the latter usually converges much faster. Modified Policy Iteration (MPI), that interporlates between the two, was introduced to improve the convergence rate of VI while remaining lighter than PI (Puterman & Shin, 1978).

When the optimal control problem one considers is large, an option is to consider approximate versions of these DP algorithms, where each iteration may be corrupted with some noise ϵ. An important question is the sensitivity of such an approach to the noise. Bertsekas & Tsitsiklis (1996) gather several results regarding approximate versions of VI and PI (thereater named AVI and API). It is known that the policy output by such procedures is guaranteed to be $\frac{2\gamma\epsilon}{(1-\gamma)^2}$-optimal. In particular, when the perturbation ϵ tends to 0, one recovers an optimal solution. This analysis was recently generalized to an approximate implementation of MPI (AMPI) independently by Canbolat & Rothblum (2012) and Scherrer et al. (2012). The better guarantee, obtained by the latter—$\frac{2\gamma\epsilon}{(1-\gamma)^2}$-optimality—exactly matches that of AVI and API. The algorithmic scheme AMPI can be implemented in various ways, reducing the original control problem to a series of (more standard) regression and classification problems (Scherrer et al., 2012), and lead to state-of-the-art results on large benchmark problems, in particular on the Tetris domain (Gabillon et al., 2013).

An apparent weakness of these sensitivity analyses is that the dependence with respect to the discount factor γ is bad: since γ is typically close to 1, the denominator of the constant $\frac{2\gamma}{(1-\gamma)^2}$ often makes the guarantee uninformative in practice. Unfortunately, it turns out that it is not so much a weakness of the analyses but a weakness of the very algorithmic approach since Bertsekas & Tsitsiklis (1996) and Scherrer & Lesner (2012) showed that the bound $\frac{2\gamma\epsilon}{(1-\gamma)^2}$ is tight respectively for API and AVI and thus cannot be improved in general. Interestingly, the authors of the latter article described a trick for modifying AVI and API so as to improve the guarantee: even though one knows that there exists a stationary policy that is optimal, Scherrer & Lesner (2012) showed that variations of AVI and API

Proceedings of the 32^{nd} International Conference on Machine Learning, Lille, France, 2015. JMLR: W&CP volume 37.

that compute ℓ-periodic non-stationary policies (thereafter named NS-AVI and NS-API) lead to an improved bound of $\frac{2\gamma\epsilon}{(1-\gamma)(1-\gamma^\ell)}$. For values of ℓ of the order of $\frac{1}{\log\frac{1}{\gamma}}$—that is equivalent to $\frac{1}{1-\gamma}$ when γ is close to 1—the guarantee is improved by a significant factor (of order $\frac{1}{1-\gamma}$). With respect to the standard AVI and API schemes, the only extra algorithmic price to pay is memory that is then $O(\ell)$ instead of $O(1)$. As often in computer science, one gets a clear trade-off between quality and memory.

To the best of our knowledge, it is not known whether the non-stationary trick also applies to a modified algorithm that would interpolate between NS-AVI and NS-API. Perhaps more importantly, it is not known whether the improved bound $\frac{2\gamma\epsilon}{(1-\gamma)(1-\gamma^\ell)}$ is tight for NS-AVI or NS-API, and even whether the standard $\frac{2\gamma\epsilon}{(1-\gamma)^2}$ bound is tight for AMPI. In this article, we fill the missing parts of this topic in the literature. We shall describe NS-AMPI, a new non-stationary MPI algorithm that generalizes all previously mentioned algorithms—AVI, API, AMPI, NS-AVI and NS-API—and prove that it returns a policy that is $\frac{2\gamma\epsilon}{(1-\gamma)(1-\gamma^\ell)}$-optimal. Furthermore, we will show that for any value of the period ℓ and any degree of interpolation between NS-AVI and NS-API, such a bound is tight. Thus, our analysis not only unifies all previous works, but it provides a complete picture of the sensitivity analysis for this large class of algorithms.

The paper is organized as follows. In Section 2 we describe the optimal control problem. Section 3 describes the state-of-the-art algorithms AMPI, NS-AVI and NS-API along with their known sensitivity analysis. In Section 4, we describe the new algorithm, NS-AMPI, and our main results: a performance guarantee (Theorem 3) and a matching lower bound (Theorem 4). Section 5 follows by providing the proof skteches of both results. Section 6 describes a small numerical illustration of our new algorithm, which gives some insight on the choice of its parameters. Section 7 concludes and mentions potential future research directions.

2. Problem Setting

We consider a discrete-time dynamic system whose state transition depends on a control. Let X be a state space. When at some state, an action is chosen from a finite action space A. The current state $x \in X$ and action $a \in A$ characterize through a homogeneous probability kernel $P(dx|x,a)$ the distribution of the next state x'. At each transition, the system is given a reward $r(x,a,x') \in \mathbb{R}$ where $r : X \times A \times X \to \mathbb{R}$ is the instantaneous reward function. In this context, the goal is to determine a sequence of actions (a_t) adapted to the past of the process until time t that maximizes the expected discounted sum of rewards from any starting state x:

$$\mathbb{E}\left[\sum_{k=0}^{\infty}\gamma^k r(x_k,a_k,x_{k+1}) \;\middle|\; x_0 = x,\ x_{t+1}\sim P(\cdot|x_t,a_t)\right],$$

where $0 < \gamma < 1$ is a discount factor. The tuple $\langle X, A, P, r, \gamma\rangle$ is called a *Markov Decision Process* (MDP) and the associated optimization problem *infinite-horizon stationary discounted optimal control* (Puterman, 1994; Bertsekas & Tsitsiklis, 1996) .

An important result of this setting is that there exists at least one stationary deterministic policy, that is a function $\pi : X \to A$ that maps states into actions, that is optimal (Puterman, 1994). As a consequence, the problem is usually recast as looking for the stationary deterministic policy π that maximizes for every state x the quantity

$$v_\pi(x) := \mathbb{E}_\pi\left[\sum_{k=0}^{\infty}\gamma^k r(x_k,\pi(x_k),x_{k+1}) \;\middle|\; x_0 = x\right], \tag{1}$$

called the value of policy π at state x. The notation E_π means that we condition on trajectories such that $x_{t+1} \sim P_\pi(\cdot|x_t)$, where $P_\pi(dx|x)$ is the stochastic kernel $P(dx|x,\pi(x))$ that chooses actions according to policy π. We shall similarly write $r_\pi : X \to \mathbb{R}$ for the function giving the immediate reward while following policy π:

$$\forall x,\ r_\pi(x) = \mathbb{E}\left[r(x_0,\pi(x_0),x_1) \mid x_0 = x,\ x_1 \sim P_\pi(\cdot|x_0)\right].$$

Two linear operators are associated with the stochastic kernel P_π: a left operator on functions $f \in \mathbb{R}^X$

$$\begin{aligned}\forall x,\ (P_\pi f)(x) &= \int f(y)P_\pi(dy|x)\\ &= \mathbb{E}\left[f(x_1) \mid x_0 = x,\ x_1 \sim P_\pi(\cdot|x_0)\right],\end{aligned}$$

and a right operator on distributions μ

$$(\mu P_\pi)(dy) = \int P_\pi(dy|x)\mu(dx).$$

In words, $(P_\pi f)(x)$ is the expected value of f after following policy π for a single time-step starting from x, and μP_π is the distribution of states after a single time-step starting from μ.

Given a policy π, it is well known that the value v_π is the unique solution of the following Bellman equation:

$$v_\pi = r_\pi + \gamma P_\pi v_\pi.$$

In other words, v_π is the fixed point of the affine operator $T_\pi v := r_\pi + \gamma P_\pi v$. The optimal value starting from state x is defined as

$$v_*(x) := \max_\pi v_\pi(x).$$

It is also well known that v_* is characterized by the following Bellman equation:

$$v_* = \max_\pi (r_\pi + \gamma P_\pi v_*) = \max_\pi T_\pi v_*,$$

where the max operator is componentwise. In other words, v_* is the fixed point of the nonlinear operator $Tv := \max_\pi T_\pi v$. Finally, for any function $v \in \mathbb{R}^X$, we say that a policy π is greedy with respect to v if it satisfies:

$$\pi \in \arg\max_{\pi'} T_{\pi'} v$$

or equivalently $T_\pi v = Tv$. We write, with some abuse of notation[1] $\mathcal{G}(v)$ any policy that is greedy with respect to v. The notions of optimal value function and greedy policies are fundamental to optimal control because of the following standard property: any policy π_* that is greedy with respect to the optimal value is an optimal policy and its value v_{π_*} is equal to v_*. Thus, the main problem amounts to computing the optimal value function v_*. The next section descibes algorithmic approaches from the literature.

3. State-of-the-Art Algorithms

We begin by describing the Approximate Modified Policy Iteration (AMPI) algorithmic scheme (Scherrer et al., 2012). Starting from an arbitrary value function v_0, AMPI generates a sequence of value-policy pairs

$$\begin{aligned} \pi_{k+1} &= \mathcal{G}(v_k) && \text{(greedy step)} \\ v_{k+1} &= (T_{\pi_{k+1}})^{m+1} v_k + \epsilon_k && \text{(evaluation step)} \end{aligned}$$

where $m \geq 0$ is a free parameter. At each iteration k, the term ϵ_k accounts for a possible approximation in the evaluation step. AMPI generalizes the well-known approximate DP algorithms Value Iteration (AVI) and Policy Iteration (API) for values $m = 0$ and $m = \infty$, respectively. In the exact case ($\epsilon_k = 0$), MPI requires less computation per iteration than PI (in a way similar to VI) and enjoys the faster convergence (in terms of number of iterations) of PI (Puterman & Shin, 1978; Puterman, 1994).

It was recently shown that controlling the errors ϵ_k when running AMPI is sufficient to ensure some performance guarantee (Scherrer et al., 2012; Canbolat & Rothblum, 2012). For instance, we have the following performance bound, that is remarkably independent of the parameter m.[2]

[1] There might be several policies that are greedy with respect to v.

[2] Note that in practice, the term ϵ_k will generally depend on m. The exact dependence may strongly depend on the precise implementation and we refer the reader to (Scherrer et al., 2012) for examples of such analyses. In this paper, we only consider the situation of a uniform error bound on the errors, all the more that extensions to more complicated errors is straightforward.

Theorem 1 (Scherrer et al. (2012, Remark 2)). *Consider AMPI with any parameter $m \geq 0$. Assume there exists an $\epsilon > 0$ such that the errors satisfy $\|\epsilon_k\|_\infty < \epsilon$ for all k. Then, the* loss *due to running policy π_k instead of the optimal policy π_* satisfies*

$$\|v_* - v_{\pi_k}\|_\infty \leq \frac{2(\gamma - \gamma^k)}{(1-\gamma)^2}\epsilon + \frac{2\gamma^k}{1-\gamma}\|v_* - v_0\|_\infty.$$

In the specific case corresponding to AVI ($m = 0$) and API ($m = \infty$), this bound matches performance guarantees that have been known for a long time (Singh & Yee, 1994; Bertsekas & Tsitsiklis, 1996). The asymptotic constant $\frac{2\gamma}{(1-\gamma)^2}$ can be very big, in particular when γ is close to 1. Unfortunately, it cannot be improved: Bertsekas & Tsitsiklis (1996, Example 6.4) showed that the bound is tight for PI, Scherrer & Lesner (2012) proved that it is tight for VI,[3] and we will prove in this article[4] the—to our knowledge unknown—fact that it is also tight for AMPI. In other words, improving the performance bound requires to change the algorithms.

Even though the theory of optimal control states that there exists a stationary policy that is optimal, Scherrer & Lesner (2012) recently showed that the performance bound of Theorem 1 could be improved in the specific cases $m = 0$ and $m = \infty$ by considering variations of AVI and API that build *periodic non-stationary policies* (instead of stationary policies). Surprisingly, the Non-Stationary AVI (NS-AVI) algorithm proposed there works almost exactly like AVI: it builds the exact same sequence of value-policy pairs from any initialization v_0 (compare with AMPI with $m = 0$):

$$\begin{aligned} \pi_{k+1} &= \mathcal{G}(v_k) && \text{(greedy step)} \\ v_{k+1} &= T_{\pi_{k+1}} v_k + \epsilon_k && \text{(evaluation step)} \end{aligned}$$

The only difference is in what is output: while AVI would return the last policy, say π_k after k iterations, NS-AVI returns the *periodic non-stationary policy* $\pi_{k,\ell}$ that loops in reverse order on the last ℓ generated policies:

$$\pi_{k,\ell} = \underbrace{\pi_k\ \pi_{k-1}\ \cdots\ \pi_{k-\ell+1}}_{\text{last } \ell \text{ policies}}\ \underbrace{\pi_k\ \pi_{k-1}\ \cdots\ \pi_{k-\ell+1}}_{\text{last } \ell \text{ policies}} \cdots$$

Following the policy $\pi_{k,\ell}$ means that the first action is selected by π_k, the second one by π_{k-1}, until the ℓ^{th} one by $\pi_{k-\ell+1}$, then the policy loops and the next actions are selected by π_k, π_{k-1}, so on and so forth. Note that when $\ell = 1$, we recover the output of AVI: the last policy π_k that is used for all actions.

[3] Though the MDP instance used to show the tightness of the bound for VI is the same as that for PI (Bertsekas & Tsitsiklis, 1996, Example 6.4), Scherrer & Lesner (2012) seem to be the first to argue about it in the literature.

[4] Theorem 4 page 4 with $\ell = 1$.

To describe the other algorithm proposed by Scherrer & Lesner (2012), Non-Stationary API (NS-API), we shall introduce the linear Bellman operator $T_{\pi_{k,\ell}}$ associated with $\pi_{k,\ell}$:

$$\forall v \in \mathbb{R}^X, \;\; T_{\pi_{k,\ell}} v = T_{\pi_k} T_{\pi_{k-1}} \dots T_{\pi_{k-\ell+1}} v.$$

It is indeed straightforward to show that the value $v_{\pi_{k,\ell}}$ obtained by following $\pi_{k,\ell}$ is the unique fixed point of $T_{\pi_{k,\ell}}$. Then, from any initial set of ℓ policies $(\pi_0, \pi_{-1}, \dots, \pi_{-\ell+1})$, NS-API generates the following sequence of value-policy pairs:

$$\begin{aligned} v_k &= v_{\pi_{k,\ell}} + \epsilon_k && \text{(evaluation step)} \\ \pi_{k+1} &= \mathcal{G}(v_k) && \text{(greedy step)} \end{aligned}$$

While computing the value v_k requires (approximately) solving the fixed point equation $v_{\pi_{k,\ell}} = T_{\pi_{k,\ell}} v_{\pi_{k,\ell}}$ of the non-stationary policy $\pi_{k,\ell}$ made of the last ℓ computed policies, the new policy π_{k+1} that is computed in the greedy step is (as usual) a simple stationary policy. After k iterations, similarly to NS-AVI, the algorithm returns the periodic non-stationary policy $\pi_{k,\ell}$. Here again, setting $\ell = 1$ provides the standard API algorithm.

On the one hand, using these non-stationary variants may require more memory since one must store ℓ policies instead of one. On the other hand, the following result shows that this extra memory allows us to improve the performance guarantee.

Theorem 2 (Scherrer & Lesner (2012, Theorems 2 and 4))**.** *Consider NS-AVI or NS-API with any parameter $l \geq 0$. Assume there exists an $\epsilon > 0$ such that the errors satisfy $\|\epsilon_k\|_\infty < \epsilon$ for all k. Then, the* loss *due to running the non-stationary policy $\pi_{k,\ell}$ instead of the optimal policy π_* satisfies*

$$\left\|v_* - v_{\pi_{k,\ell}}\right\|_\infty \leq \frac{2(\gamma - \gamma^k)}{(1-\gamma)(1-\gamma^\ell)} \epsilon + \gamma^k g_0.$$

where $g_0 = \frac{2}{1-\gamma^\ell} \|v_* - v_0\|_\infty$ for NS-AVI or $g_0 = \left\|v_* - v_{\pi_{0,\ell}}\right\|_\infty$ for NS-API.

For any $\ell \geq 1$, it is a factor $\frac{1-\gamma}{1-\gamma^\ell}$ better than in Theorem 1. Using $\ell = \left\lceil \frac{1}{1-\gamma} \right\rceil$ yields[5] an asymptotic performance bound of $\frac{3.164\gamma}{1-\gamma}\epsilon$. which constitutes an improvement of order $O(\frac{1}{1-\gamma})$, which is significant in typical situations where γ is close to 1.

[5] Using the facts that $1 - \gamma \leq -\log\gamma$ and $\log\gamma \leq 0$, we have $\log\gamma^\ell \leq \log\gamma^{\frac{1}{1-\gamma}} \leq \frac{1}{-\log\gamma}\log\gamma = -1$ hence $\gamma^\ell \leq e^{-1}$. Therefore $\frac{2}{1-\gamma^\ell} \leq \frac{2}{1-e^{-1}} < 3.164$.

4. Main results

We are now ready to present the first contribution of this paper. We shall introduce a new algorithm, Non-Stationary AMPI (NS-AMPI), that generalizes NS-AVI and NS-API (in the same way the standard AMPI algorithm generalizes standard AVI and API) and AMPI (in the same way NS-VI and NS-PI respectively generalize AVI and API). Given some free parameters $m \geq 0$ and $\ell \geq 1$, an arbitrary value function v_0 and an arbitrary set of $\ell - 1$ policies $\pi_0, \pi_{-1}, \pi_{-\ell+2}$, consider the algorithm that builds a sequence of value-policy pairs as follows:

$$\begin{aligned} \pi_{k+1} &= \mathcal{G}(v_k) && \text{(greedy step)} \\ v_{k+1} &= (T_{\pi_{k+1,\ell}})^m T_{\pi_{k+1}} v_k + \epsilon_k. && \text{(evaluation step)} \end{aligned}$$

While the greedy step is identical to that of all algorithms, the evaluation step involves the non-stationary Bellman operator $T_{\pi_{k+1,\ell}}$ (composed with itself m times) that we introduced in the previous section, composed with the standard Bellman operator $T_{\pi_{k+1}}$. As in NS-AVI and NS-API, after k iterations, the output of the algorithm is the periodic non-stationary policy $\pi_{k,\ell}$. For the values $m = 0$ and $m = \infty$, it is easy to see that one respectively recovers NS-AVI and NS-API. When $\ell = 1$, one recovers AMPI (that itself generalizes the standard AVI and API algorithms, obtained if we further set respectively $m = 0$ and $m = \infty$).

At this point, a natural question is whether the previous sensitivity results extend to this more general setting. As the following original result states, the answer is yes.

Theorem 3. *Consider NS-AMPI with any parameters $m \geq 0$ and $\ell \geq 1$. Assume there exists an $\epsilon > 0$ such that the errors satisfy $\|\epsilon_k\|_\infty < \epsilon$ for all k. Then, the* loss *due to running policy $\pi_{k,\ell}$ instead of the optimal policy π_* satisfies*

$$\left\|v_* - v_{\pi_{k,\ell}}\right\|_\infty \leq \frac{2(\gamma - \gamma^k)}{(1-\gamma)(1-\gamma^\ell)} \epsilon + \frac{2\gamma^k}{1-\gamma} \|v_* - v_0\|_\infty.$$

Theorem 3 asymptotically generalizes both Theorem 1 for $\ell > 1$ (the bounds match when $\ell = 1$) and Theorem 2 for $m > 0$ (the bounds are very close when $m = 0$ or $m = \infty$). As already observed for AMPI, it is remarkable that this performance bound is independent of m.

The second main result of this article is that the bound of Theorem 3 is tight, in the precise sense formalized by the following theorem.

Theorem 4. *For all parameter values $m \geq 0$ and $\ell \geq 1$, for all discount factor γ, for all $\epsilon > 0$, there exists an MDP instance, an initial value function v_0, a set of initial policies $\pi_0, \pi_{-1}, \dots, \pi_{-\ell+2}$ and a sequence of error terms $(\epsilon_k)_{k \geq 1}$ satisfying $\|\epsilon_k\|_\infty \leq \epsilon$, such that for all iterations k, the bound of Theorem 3 is satisfied with equality.*

This theorem generalizes the (separate) tightness results for PI ($m = \infty$, $\ell = 1$) (Bertsekas & Tsitsiklis, 1996) and for VI ($m = 0$, $\ell = 1$) (Scherrer & Lesner, 2012), which are the only results we are aware of. To our knowledge, this result is new even for the standard AMPI algorithm (m arbitrary but $\ell = 1$), and for the non-trivial instances of NS-VI ($m = 0$, $\ell > 1$) and NS-PI ($m = \infty$, $\ell > 1$) proposed by Scherrer & Lesner (2012).

Since it is well known that there exists an optimal policy that is stationary, our result—as well as those of Scherrer & Lesner (2012)—suggesting to consider non-stationary policies may appear strange. There exists, however, a very simple approximation scheme of discounted infinite-horizon control problems—that has to our knowledge never been documented in the literature—that sheds some light on the deep reason why non-stationary policies may be an interesting option. Given an infinite-horizon problem, consider approximating it by a finite-horizon discounted control problem by "cutting the horizon" after some sufficiently big instant T (that is assume there is no reward after time T). Contrary to the original infinite-horizon problem, the resulting finite-horizon problem is non-stationary, and has therefore *naturally* a non-stationary solution that is built by dynamic programming in reverse order. Moreover, it can be shown (Kakade, 2003, by adapting the proof of Theorem 2.5.1) that solving this finite-horizon with VI with a potential error of ϵ at each iteration, will induce at most a performance error of $2\sum_{i=0}^{T-1}\gamma^t\epsilon = \frac{2(1-\gamma^T)}{1-\gamma}\epsilon$. If we add the error due to truncating the horizon ($\gamma^T\frac{\max_{s,a}|r(s,a)|}{1-\gamma}$), we get an overall error of order $O\left(\frac{1}{1-\gamma}\epsilon\right)$ for a memory T of the order of[6] $\tilde{O}\left(\frac{1}{1-\gamma}\right)$. Though this approximation scheme may require a significant amount of memory (when γ is close to 1), it achieves the same $O(\frac{1}{1-\gamma})$ improvement over the performance bound of AVI/API/AMPI as NS-AVI/NS-API/NS-AMPI do. In comparison, the non-stationary algorithms with a fixed period ℓ can be seen as a more flexible way to make the trade-off between the memory and the quality.

5. Proof sketches

We begin by considering Theorem 3. While the performance guarantee was obtained through three independent proofs for NS-VI, NS-PI and AMPI, the more general setting that we consider here involves a totally unified proof, which we describe in the remaining of this section.

We write P_k (resp. P_*) for the transition kernel P_{π_k} (resp. P_{π_*}) induced by the stationary policy π_k (resp. π_*). We will write T_k (resp. T_*) for the associated Bellman operator. Similarly, we will write $P_{k,\ell}$ for the transition kernel associated with the non-stationary policy $\pi_{k,\ell}$ and $T_{k,\ell}$ for its associated Bellman operator. For $k \geq 0$ we define the following quantities: $b_k = T_{k+1}v_k - T_{k+1,\ell}T_{k+1}v_k$, $s_k = v_k - v_{\pi_{k,\ell}} - \epsilon_k$, $d_k = v_* - v_k + \epsilon_k$, and $l_k = v_* - v_{\pi_{k,\ell}}$. The last quantity, the loss l_k of using policy $\pi_{k,\ell}$ instead of π_* is the quantitiy we want to ultimately upper bound.

The core of the proof consists in deriving the following recursive relations.

Lemma 1. *The quantities b_k, s_k and d_k satisfy:*

$$b_k \leq \gamma P_{k+1}\left((\gamma^\ell P_{k,\ell})^m b_{k-1} + (I - \gamma^\ell P_{k,\ell})\epsilon_k\right),$$

$$d_k = \gamma P_* d_{k-1} - \gamma P_* \epsilon_{k-1} + \sum_{i=0}^{m-1}(\gamma^\ell P_{k,\ell})^i b_{k-1},$$

$$s_k = (\gamma^\ell P_{k,\ell})^m \sum_{j=0}^{\infty}(\gamma^\ell P_{k,\ell})^j \left(T_k v_{k-1} - T_{k,\ell}T_k v_{k-1}\right).$$

Since ϵ is a uniform upper-bound on the pointwise absolute value of the errors $|\epsilon_k|$, the first inequality implies that $b_k \leq O(\epsilon)$, and as a result, the second and third inequalities gives us $d_k \leq O(\epsilon)$ and $s_k \leq O(\epsilon)$. This means that the loss $l_k = d_k + s_k$ will also satisfy $l_k \leq O(\epsilon)$ and the result is obtained by taking the norm $\|\cdot\|_\infty$. The actual bound given in the theorem requires a careful expansion of these three inequalities where we make precise what we have just hidden in the O-notations. The details of this expansion are tedious and deferred to Appendix B of the Supplementary Material. We thus now concentrate on the proof of these relations.

Proof of Lemma 1. We will repeatedly use the fact that since policy π_{k+1} is greedy with respect to v_k, we have

$$\forall \pi', \;\; T_{k+1}v_k \geq T_{\pi'}v_k. \tag{2}$$

For a non-stationary policy $\pi_{k,\ell}$, the induced ℓ-step transition kernel is $P_{k,\ell} = P_k P_{k-1}\cdots P_{k-\ell+1}$. As a consequence, for any function $f : \mathcal{S} \to \mathbb{R}$, the operator $T_{k,\ell}$ may be expressed as: $T_{k,\ell}f = r_k + \gamma P_{k,1}r_{k-1} + \gamma^2 P_{k,2}r_{k-2} + \cdots + \gamma^\ell P_{k,\ell}f$ and, for any function $g : \mathcal{S} \to \mathbb{R}$, we have

$$T_{k,\ell}f - T_{k,\ell}g = \gamma^\ell P_{k,\ell}(f - g) \tag{3}$$

and

$$T_{k,\ell}(f + g) = T_{k,\ell}f + \gamma^\ell P_{k,\ell}(g). \tag{4}$$

Let us now bound b_k. We have

$$\begin{aligned} b_k &= T_{k+1}v_k - T_{k+1,\ell}T_{k+1}v_k \\ &\overset{Eq.(2)}{\leq} T_{k+1}v_k - T_{k+1,\ell}T_{k-\ell+1}v_k \\ &= T_{k+1}v_k - T_{k+1}T_{k,\ell}v_k \\ &= \gamma P_{k+1}\left(v_k - T_{k,\ell}v_k\right) \end{aligned}$$

[6] We use the fact that $\gamma^T K < \frac{\epsilon}{1-\gamma} \Leftrightarrow T > \frac{\log\frac{(1-\gamma)K}{\epsilon}}{\log\frac{1}{\gamma}} \simeq \frac{\log\frac{(1-\gamma)K}{\epsilon}}{1-\gamma}$ with $K = \frac{\max_{s,a}|r(s,a)|}{1-\gamma}$.

$$
\begin{aligned}
&= \gamma P_{k+1} \left((T_{k,\ell})^m T_k v_{k-1} \right. \\
&\qquad \left. + \epsilon_k - T_{k,\ell} \left((T_{k,\ell})^m T_k v_{k-1} + \epsilon_k \right) \right) \\
&\overset{Eq.(4)}{=} \gamma P_{k+1} \left((T_{k,\ell})^m T_k v_{k-1} \right. \\
&\qquad \left. - (T_{k,\ell})^{m+1} T_k v_{k-1} + (I - \gamma^\ell P_{k,\ell}) \epsilon_k \right) \\
&\overset{Eq.(3)}{=} \gamma P_{k+1} \left((\gamma^\ell P_{k,\ell})^m \left(T_k v_{k-1} \right. \right. \\
&\qquad \left. \left. - T_{k,\ell} T_k v_{k-1} \right) + (I - \gamma^\ell P_{k,\ell}) \epsilon_k \right) \\
&= \gamma P_{k+1} \left((\gamma^\ell P_{k,\ell})^m b_{k-1} + (I - \gamma^\ell P_{k,\ell}) \epsilon_k \right).
\end{aligned}
$$

We now turn to the bound of d_k:

$$
\begin{aligned}
d_k &= v_* - v_k + \epsilon_k \\
&= v_* - (T_{k,\ell})^m T_k v_{k-1} \\
&= v_* - T_k v_{k-1} \\
&\quad + \sum_{i=0}^{m-1} (T_{k,\ell})^i T_k v_{k-1} - (T_{k,\ell})^{i+1} T_k v_{k-1} \\
&\overset{Eq.(3)}{=} T_* v_* - T_k v_{k-1} \\
&\quad + \sum_{i=0}^{m-1} (\gamma^\ell P_{k,\ell})^i \left(T_k v_{k-1} - T_{k,\ell} T_k v_{k-1} \right) \\
&\overset{Eq.(2)}{\leq} T_* v_* - T_* v_{k-1} + \sum_{i=0}^{m-1} (\gamma^\ell P_{k,\ell})^i b_{k-1} \\
&\overset{Eq.(3)}{=} \gamma P_* (v_* - v_{k-1}) + \sum_{i=0}^{m-1} (\gamma^\ell P_{k,\ell})^i b_{k-1} \\
&= \gamma P_* d_{k-1} - \gamma P_* \epsilon_{k-1} + \sum_{i=0}^{m-1} (\gamma^\ell P_{k,\ell})^i b_{k-1}.
\end{aligned}
$$

Finally, we prove the relation for s_k:

$$
\begin{aligned}
s_k &= v_k - v_{\pi_{k,\ell}} - \epsilon_k \\
&= (T_{k,\ell})^m T_k v_{k-1} - v_{\pi_{k,\ell}} \\
&= (T_{k,\ell})^m T_k v_{k-1} - (T_{k,\ell})^\infty T_{k,\ell} T_k v_{k-1} \\
&= (\gamma^\ell P_{k,\ell})^m \sum_{j=0}^{\infty} (\gamma^\ell P_{k,\ell})^j \left(T_k v_{k-1} - T_{k,\ell} T_k v_{k-1} \right). \square
\end{aligned}
$$

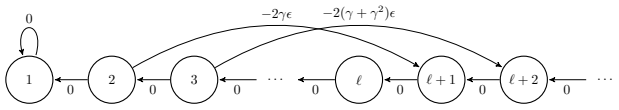

Figure 1. The deterministic MDP matching the bound of Theorem 3.

We now turn to the tightness results given in Theorem 4. The proof considers a generalization of the MDP instance used to prove the tightness of the bound for VI (Scherrer & Lesner, 2012) and PI (Bertsekas & Tsitsiklis, 1996, Example 6.4). Precisely, this MDP consists of states $\{1, 2, \dots\}$, two actions: left ($\leftarrow$) and right ($\rightarrow$); the reward function r and transition kernel P are characterized as follows for any state $i \geq 2$:

$$
\begin{aligned}
r(i, \leftarrow) &= 0, & r(i, \rightarrow) &= -2 \frac{\gamma - \gamma^i}{1 - \gamma} \epsilon, \\
P(i|i+1, \leftarrow) &= 1, & P(i+\ell-1|i, \rightarrow) &= 1,
\end{aligned}
$$

and $r(1) = 0$ and $P(1|1)1$ for state 1 (all the other transitions having zero probability mass). As a shortcut, we will use the notation r_i for the non-zero reward $r(i, \rightarrow)$ in state i. Figure 1 depicts the general structure of this MDP. It is easily seen that the optimal policy π_* is to take $\leftarrow$ in all states $i \geq 2$, as doing otherwise would incur a negative reward. Therefore, the optimal value $v_*(i)$ is 0 in all states i. The proof of the above theorem considers that we run AMPI with $v_0 = v_* = 0$, $\pi_0 = \pi_{-1} = \dots = \pi_{\ell+2} = \pi_*$, and the following sequence of error terms:

$$
\forall i, \quad \epsilon_k(i) = \begin{cases} -\epsilon & \text{if } i = k, \\ \epsilon & \text{if } i = k + \ell, \\ 0 & \text{otherwise.} \end{cases}
$$

In such a case, one can prove that the sequence of policies $\pi_1, \pi_2, \dots, \pi_k$ that are generated up to iteration k is such that for all $i \leq k$, the policy π_i takes $\leftarrow$ in all states but i, where it takes $\rightarrow$. As a consequence, a non-stationary policy $\pi_{k,\ell}$ built from this sequence takes $\rightarrow$ in k (as dictated by π_k), which transfers the system into state $k + \ell - 1$ incurring a reward of r_k. Then the policies $\pi_{k-1}, \pi_{k-2}, \dots, \pi_{k-\ell+1}$ are followed, each indicating to take $\leftarrow$ with 0 reward. After ℓ steps, the system is again is state k and, by the periodicity of the policy, must again use the action $\pi_k(k) = \rightarrow$. The system is thus stuck in a loop, where every ℓ steps a negative reward of r_k is received. Consequently, the value of this policy from state k is:

$$
v_{\pi_{k,\ell}}(k) = \frac{r_k}{1 - \gamma^\ell} = -\frac{\gamma - \gamma^k}{(1 - \gamma)(1 - \gamma^\ell)} 2\epsilon.
$$

As a consequence, we get the following lower bound,

$$
\left\| v_* - v_{\pi_{k,\ell}} \right\|_\infty \geq |v_{\pi_{k,\ell}}(k)| = \frac{\gamma - \gamma^k}{(1 - \gamma)(1 - \gamma^\ell)} 2\epsilon
$$

which *exactly* matches the upper bound of Theorem 3 (since $v_0 = v_* = 0$). The proof of this result involves computing the values $v_k(i)$ for all states i, steps k of the algorithm, and values m and ℓ of the parameters, and proving that the policies π_{k+1} that are greedy with respect to these values satisfy what we have described above. Due to lack of space, the complete proof is deferred to Appendix B of the Supplementary Material; in Lemma 7 and the associated Figures 4 and 5 there, note the quite complex shape of the value function that is induced by the cyclic nature of the MDP and the NS-AMPI algorithm.

6. Empirical Illustration

In this section, we describe an empirical illustration of the new algorithm NS-AMPI. Note that the goal here is not to convince the reader that the new degrees of freedom for approximate dynamic programming may be interesting in difficult real control problems—we leave this important question to future work—but rather to give some insight, on small and artificial well-controlled problems, on the effect of the main parameters m and ℓ.

The problem we consider, the dynamic location problem from Bertsekas & Yu (2012), involves a repairman moving between n sites according to some transition probabilities. As to allow him do his work, a trailer containing supplies for the repair jobs can be relocated to any of the sites at each decision epoch. The problem consists in finding a relocation policy for the trailer according the repairman's and trailer's positions which maximizes the discounted expectation of a reward function.

Given n sites, the state space has n^2 states comprising the locations of both the repairman and the trailer. There are n actions, each one corresponds to a possible destination of the trailer. Given an action $a = 1, \ldots, n$, and a state $s = (s_r, s_t)$, where the repairman and the trailer are at locations s_r and s_t, respectively, we define the reward as $r(s, a) = -|s_r - s_t| - |s_t - a|/2$. At any time-step the repairman moves from its location $s_r < n$ with uniform probability to any location $s_r \leq s'_r \leq n$; when $s_r = n$, he moves to site 1 with probability 0.75 or otherwise stays. Since the trailer moves are deterministic, the transition kernel is

$$P((s'_r, a)|(s_r, s_t), a) = \begin{cases} \frac{1}{n - s_r + 1} & \text{if } s_r < n \\ 0.75 & \text{if } s_r = n \wedge s'_r = 1 \\ 0.25 & \text{if } s_r = n \wedge s'_r = n \end{cases}$$

and 0 everywhere else.

We evaluated the empirical performance gain of using non-stationary policies by implementing the algorithm using random error vectors ϵ_k, with each component being uniformly random between 0 and some user-supplied value ϵ. The adjustable size (with n) of the state and actions spaces allowed to compute an optimal policy to compare with the approximate ones generated by NS-AMPI for all combinations of parameters $\ell \in \{1, 2, 5, 10\}$ and $m \in \{1, 2, 5, 10, 25, \infty\}$. Recall that the cases $m = 1$ and $m = \infty$ correspond respectively to the NS-VI and NS-PI, while the case $\ell = 1$ corresponds to AMPI. We used $n = 8$ locations, $\gamma = 0.98$ and $\epsilon = 4$ in all experiments.

Figure 2 shows the average value of the error $v_* - v_{\pi_{k,\ell}}$ per iteration for the different values of parameters m and ℓ. For each parameter combination, the results are obtained by averaging over 250 runs. While higher values of ℓ impacts computational efficiency (by a factor $O(\ell)$) it always results with better asymptotic performance. Especially with the lower values of m, a higher ℓ allows for faster convergence. While increasing m, this trend fades to be finally reversed in favor of faster convergence for small ℓ. However, while small ℓ converges faster, it is with greater error than with higher ℓ after convergence. It can be seen that convergence is attained shortly after the ℓ^{th} iteration which can be explained by the fact that the first policies (involving $\pi_0, \pi_{-1}, \ldots, \pi_{-\ell+2}$), are of poor quality and the algorithm must perform at least ℓ iterations to "push them out" of $\pi_{k,\ell}$.

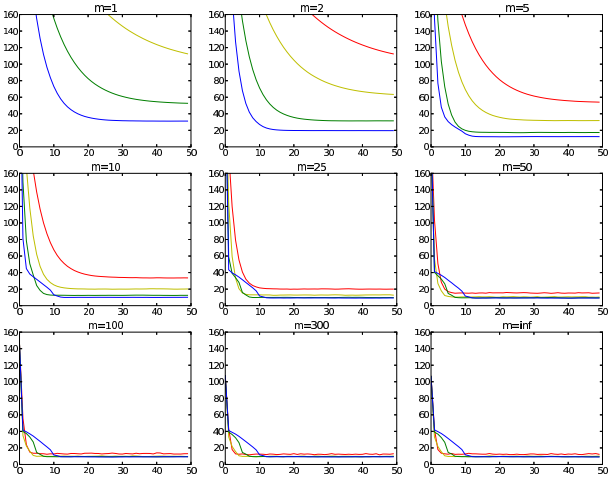

Figure 2. Average error of policy $\pi_{k,\ell}$ per iteration k of NS-AMPI. Red lines for $\ell = 1$, yellow for $\ell = 2$, green for $\ell = 5$ and blue for $\ell = 10$.

We conducted a second experiment to study the relative influence of the parameters ℓ and m. From the observation that, in the very setting we are considering, the time complexity of an iteration of NS-AMPI can be roughly summarized by the number $\ell m + 1$ of applications of a stationary policy's Bellman operator, we ran the algorithm for fixed values of the product ℓm and measured the asymptotic policy error for varying values of ℓ after 150 iterations. These results are depicted on Figure 3. This setting gives insight on how to set both parameters for a given "time budget" ℓm. While runs with a lower ℓ are slightly faster to converge, higher values always give the best policies after a sufficient number of iterations, and greatly reduces the variance across all runs, showing that non-stationarity adds robustness to the approximation noise. Regarding asymptotic quality, it thus appears that the best setting is to favor ℓ instead of m.

Overall, both experiments confirm our theoretical analysis that the main parameter for asymptotic quality is ℓ. Regarding the rate of convergence, the first experiments sug-

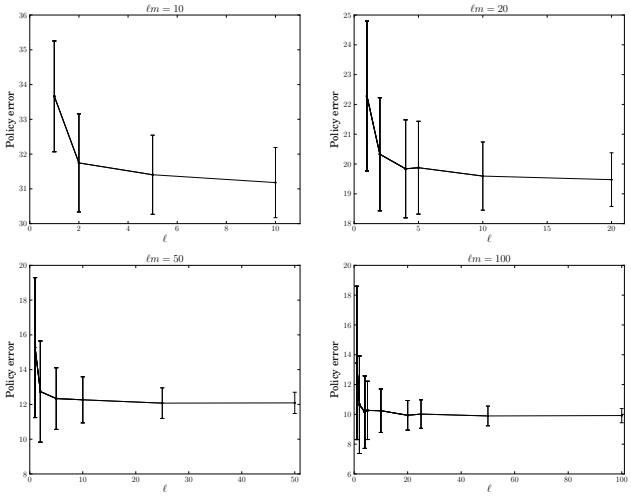

Figure 3. Policy error and standard deviation after 150 iterations for different different values of ℓ. Each plot represents a fixed value of the product ℓm. Data is collected over 250 runs with $n = 8$.

gests that too big values of ℓ may be harmful. In practice, a schedule where ℓ progressively grows while m decreases may provide the best compromise. Confirming this, as well as studying approximate implementations designed for real problems constitutes a matter for future investigation.

7. Conclusion

We have described a new dynamic-programming scheme, NS-AMPI, that extends and unifies several state-of-the-art algorithms of the literature: AVI, API, AMPI, NS-VI, and NS-PI. NS-AMPI has two integer parameters: $m \geq 0$ that allows to move from a VI-style update to a PI-style update, and $\ell \geq 1$ that characterizes the period of the non-stationary policy that it builds. In Theorem 3, we have provided a performance guarantee for this algorithm that is independent of m and that improves when ℓ increases; since ℓ directly controls the memory of the process, this allows to make a trade-off between memory and quality. In the literature, similar upper bounds were only known for AMPI (Scherrer et al., 2012)—$\ell = 1$ and m arbitrary—and NS-AVI/NS-API (Scherrer & Lesner, 2012)—ℓ arbitrary but $m \in \{0, \infty\}$. For most settings—$\ell > 1$ and $1 \leq m < \infty$—the result is new. By exhibiting a specially designed MDP, we argued (Theorem 4) that our analysis is tight. Similar lower bounds were only known for AVI and API—$\ell = 1$ and $m \in \{0, \infty\}$. In other words, we have generalized the scarce existing bounds in a unified setting and closed the gap between the upper and lower bounds for all values of $m \geq 0$ and $\ell \geq 1$.

A practical limitation of Theorem 3 is that it assumes that the errors ϵ_k are controlled in max norm. In practice, the evaluation step of dynamic programming algorithm is usually done through some regression scheme—see for instance (Bertsekas & Tsitsiklis, 1996; Antos et al., 2007a;b; Scherrer et al., 2012)—and thus controlled through the $L_{2,\mu}$ norm, defined as $\|f\|_{2,\mu} = \sqrt{\int f(x)\mu(dx)}$. Munos (2003; 2007) originally developed such analyzes for AVI and API. Farahmand et al. (2010) and Scherrer et al. (2012) later improved them. Using a technical lemma due to Scherrer et al. (2012, Lemma 3), one can easily deduce[7] from our analysis (developed in Appendix A of the Supplementary Material) the following performance bound.

Corollary 1. *Consider AMPI with any parameters $m \geq 0$ and $\ell \geq 1$. Assume there exists an $\epsilon > 0$ such that the errors satisfy $\|\epsilon_k\|_{2,\mu} < \epsilon$ for all k. Then, the expected (with respect to some initial measure ρ)* loss *due to running policy $\pi_{k,\ell}$ instead of the optimal policy π_* satisfies*

$$\mathbb{E}_{s\sim\rho}[v_*(s) - v_{\pi_{k,\ell}}(s)] \leq \frac{2(\gamma - \gamma^k)C_{1,k,\ell}}{(1-\gamma)(1-\gamma^\ell)}\epsilon + O\left(\frac{\gamma^k}{1-\gamma}\right),$$

$$\text{where } C_{j,k,l} = \frac{(1-\gamma)(1-\gamma^l)}{\gamma^j - \gamma^k}\sum_{i=j}^{k-1}\sum_{n=i}^{\infty}\gamma^{i+ln}c(i+ln)$$

is a convex combination of concentrability coefficients based on Radon-Nikodym derivatives $c(j) = \max_{\pi_1,\cdots,\pi_j}\left\|\frac{d(\rho P_{\pi_1}P_{\pi_2}\cdots P_{\pi_j})}{d\mu}\right\|_{2,\mu}$.

With respect to the previous bound in norm $\|\cdots\|_\infty$, this bound involves extra constants $C_{j,k,l} \geq 1$. Each such coefficient $C_{j,k,l}$ is a convex combination of terms $c(i)$, that each quantifies the difference between 1) the distribution μ used to control the errors and 2) the distribution obtained by starting from ρ and making k steps with arbitrary sequences of policies. Overall, this extra constant can be seen as a measure of stochastic smoothness of the MDP (the smoother, the smaller). Further details on these coefficients can be found in (Munos, 2003; 2007; Farahmand et al., 2010).

We have shown on a small numerical study the significant influence of the parameter ℓ on the asymptotic quality of approximately optimal controllers, and suggested that optimizing the speed of convergence may require a fine schedule between ℓ and m. Instantiating and analyzing specific implementations of NS-AMPI as was done recently for AMPI (Scherrer et al., 2012), and applying them on large domains constitutes interesting future work.

[7] Precisely, Lemma 3 of (Scherrer et al., 2012) should be applied to Equation (8) page 15 in Appendix A of the Supplementary Material.

References

Antos, A., Munos, R., and Szepesvári, C. Fitted Q-iteration in continuous action-space MDPs. In *NIPS*, 2007a.

Antos, A., Szepesvári, C., and Munos, R. Value-iteration based fitted policy iteration: learning with a single trajectory. In *Approximate Dynamic Programming and Reinforcement Learning, 2007. ADPRL 2007*, pp. 330–337. IEEE, 2007b.

Bertsekas, D.P. and Tsitsiklis, J.N. *Neuro-dynamic programming*. Athena Scientific, 1996.

Bertsekas, D.P. and Yu, H. Q-learning and enhanced policy iteration in discounted dynamic programming. *Mathematics of Operations Research*, 37(1):66–94, 2012.

Canbolat, P. and Rothblum, U. (Approximate) iterated successive approximations algorithm for sequential decision processes. *Annals of Operations Research*, pp. 1–12, 2012. ISSN 0254-5330.

Farahmand, A.M., Munos, R., and Szepesvári, Cs. Error propagation for approximate policy and value iteration (extended version). In *NIPS*, 2010.

Gabillon, Victor, Ghavamzadeh, Mohammad, and Scherrer, Bruno. Approximate Dynamic Programming Finally Performs Well in the Game of Tetris. In *Neural Information Processing Systems (NIPS) 2013*, South Lake Tahoe, United States, December 2013.

Kakade, S.M. *On the Sample Complexity of Reinforcement Learning*. PhD thesis, University College London, 2003.

Munos, R. Error bounds for approximate policy iteration. In *International Conference on Machine Learning (ICML)*, pp. 560–567, 2003.

Munos, R. Performance bounds in L_p-norm for approximate value iteration. *SIAM Journal on Control and Optimization*, 46(2):541–561, 2007.

Puterman, M.L. *Markov decision processes: Discrete stochastic dynamic programming*. John Wiley & Sons, Inc., 1994.

Puterman, M.L. and Shin, M.C. Modified policy iteration algorithms for discounted Markov decision problems. *Management Science*, 24(11):1127–1137, 1978.

Scherrer, B. and Lesner, B. On the Use of Non-Stationary Policies for Stationary Infinite-Horizon Markov Decision Processes. In *Advances in Neural Information Processing Systems 25*, pp. 1835–1843, 2012.

Scherrer, B., Ghavamzadeh, M., Gabillon, V., and Geist, M. Approximate Modified Policy Iteration. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pp. 1207–1214, July 2012.

Singh, S. and Yee, R. An upper bound on the loss from approximate optimal-value functions. *Machine Learning*, 16-3:227–233, 1994.

Supplementary Material for Non-Stationary Approximate Modified Policy Iteration

A. Proof of Theorem 3

For clarity, we here provide a detailed and complete proof. Throughout this proof we will write P_k (resp. P_*) for the transition kernel P_{π_k} (resp. P_{π_*}) induced by the stationary policy π_k (resp. π_*). We will write T_k (resp. T_*) for the associated Bellman operator. Similarly, we will write $P_{k,\ell}$ for the transition kernel associated with the non-stationary policy $\pi_{k,\ell}$ and $T_{k,\ell}$ for its associated Bellman operator.

For $k \geq 0$ we define the following quantities:

- $b_k = T_{k+1}v_k - T_{k+1,\ell}T_{k+1}v_k$. This quantity which we will call the *residual* may be viewed as a non-stationary analogue of the Bellman residual $v_k - T_{k+1}v_k$.
- $s_k = v_k - v_{\pi_{k,\ell}} - \epsilon_k$. We will call it *shift*, as it measures the shift between the value $v_{\pi_{k,\ell}}$ and the estimate v_k before incurring the error.
- $d_k = v_* - v_k + \epsilon_k$. This quantity, called *distance* thereafter, provides the distance between the k^{th} value function (before the error is added) and the optimal value function.
- $l_k = v_* - v_{\pi_{k,\ell}}$. This is the *loss* of the policy $v_{\pi_{k,\ell}}$. The loss is always non-negative since no policy can have a value greater than or equal to v_*.

The proof is outlined as follows. We first provide a bound on b_k which will be used to express both the bounds on s_k and d_k. Then, observing that $l_k = s_k + d_k$ will allow to express the bound of $\|l_k\|_\infty$ stated by Theorem 3. Our arguments extend those made by Scherrer et al. (2012) in the specific case $\ell = 1$.

We will repeatedly use the fact that since policy π_{k+1} is greedy with respect to v_k, we have

$$\forall \pi', \;\; T_{k+1}v_k \geq T_{\pi'}v_k. \tag{5}$$

For a non-stationary policy $\pi_{k,\ell}$, the induced ℓ-step transition kernel is

$$P_{k,\ell} = P_k P_{k-1} \cdots P_{k-\ell+1}.$$

As a consequence, for any function $f : \mathcal{S} \to \mathbb{R}$, the operator $T_{k,\ell}$ may be expressed as:

$$T_{k,\ell}f = r_k + \gamma P_{k,1} r_{k-1} + \gamma^2 P_{k,2} r_{k-2} + \cdots + \gamma^{\ell-1} P_{k,\ell-1} r_{k-\ell+1} + \gamma^\ell P_{k,\ell} f$$

then, for any function $g : \mathcal{S} \to \mathbb{R}$, we have

$$T_{k,\ell}f - T_{k,\ell}g = \gamma^\ell P_{k,\ell}(f - g) \tag{6}$$

and

$$T_{k,\ell}(f + g) = T_{k,\ell}f + \gamma^\ell P_{k,\ell}(g). \tag{7}$$

The following notation will be useful.

Definition 1 (Scherrer et al. (2012))**.** *For a positive integer n, we define $\mathbb{P}_n$ as the set of discounted transition kernels that are defined as follows:*

1. *for any set of n policies $\{\pi_1, \ldots, \pi_n\}$, $(\gamma P_{\pi_1})(\gamma P_{\pi_2}) \cdots (\gamma P_{\pi_n}) \in \mathbb{P}_n$,*
2. *for any $\alpha \in (0, 1)$ and $P_1, P_2 \in \mathbb{P}_n$, $\alpha P_1 + (1 - \alpha)P_2 \in \mathbb{P}_n$*

With some abuse of notation, we write Γ^n for denoting any element of $\mathbb{P}_n$.

Example 1 (Γ^n notation)**.** *If we write a transition kernel P as $P = \alpha_1\Gamma^i + \alpha_2\Gamma^j\Gamma^k = \alpha_1\Gamma^i + \alpha_2\Gamma^{j+k}$, it should be read as: "There exists $P_1 \in \mathbb{P}_i$,$P_2 \in \mathbb{P}_j$,$P_3 \in \mathbb{P}_k$ and $P_4 \in \mathbb{P}_{j+k}$ such that $P = \alpha_1 P_1 + \alpha_2 P_2 P_3 = \alpha_1 P_1 + \alpha_2 P_4$.".*

We first provide three lemmas bounding the residual, the shift and the distance, respectively.

Lemma 2 (residual bound)**.** *The residual b_k satisfies the following bound:*

$$b_k \leq \sum_{i=1}^{k} \Gamma^{(\ell m+1)(k-i)} x_i + \Gamma^{(\ell m+1)k} b_0$$

where

$$x_k = (I - \Gamma^\ell)\Gamma\epsilon_k.$$

Proof. We have:

$$\begin{aligned}
b_k &= T_{k+1}v_k - T_{k+1,\ell}T_{k+1}v_k \\
&\leq T_{k+1}v_k - T_{k+1,\ell}T_{k-\ell+1}v_k && \{T_{k+1}v_k \geq T_{k-\ell+1}v_k \text{ (5)}\} \\
&= T_{k+1}v_k - T_{k+1}T_{k,\ell}v_k \\
&= \gamma P_{k+1}\left(v_k - T_{k,\ell}v_k\right) \\
&= \gamma P_{k+1}\left((T_{k,\ell})^m T_k v_{k-1} + \epsilon_k - T_{k,\ell}\left((T_{k,\ell})^m T_k v_{k-1} + \epsilon_k\right)\right) \\
&= \gamma P_{k+1}\left((T_{k,\ell})^m T_k v_{k-1} - (T_{k,\ell})^{m+1} T_k v_{k-1} + (I - \gamma^\ell P_{k,\ell})\epsilon_k\right) && \{(7)\} \\
&= \gamma P_{k+1}\left((\gamma^\ell P_{k,\ell})^m \left(T_k v_{k-1} - T_{k,\ell}T_k v_{k-1}\right) + (I - \gamma^\ell P_{k,\ell})\epsilon_k\right) && \{(6)\} \\
&= \gamma P_{k+1}\left((\gamma^\ell P_{k,\ell})^m b_{k-1} + (I - \gamma^\ell P_{k,\ell})\epsilon_k\right).
\end{aligned}$$

Which can be written as

$$b_k \leq \Gamma(\Gamma^{\ell m} b_{k-1} + (I - \Gamma^\ell)\epsilon_k) = \Gamma^{\ell m+1} b_{k-1} + x_k.$$

Then, by induction:

$$b_k \leq \sum_{i=0}^{k-1} \Gamma^{(\ell m+1)i} x_{k-i} + \Gamma^{(\ell m+1)k} b_0 = \sum_{i=1}^{k} \Gamma^{(\ell m+1)(k-i)} x_i + \Gamma^{(\ell m+1)k} b_0.$$

□

Lemma 3 (distance bound)**.** *The distance d_k satisfies the following bound:*

$$d_k \leq \sum_{i=1}^{k} \sum_{j=0}^{mi-1} \Gamma^{\ell j+i-1} x_{k-i} + \sum_{i=1}^{k} \Gamma^{i-1} y_{k-i} + z_k,$$

where

$$y_k = -\Gamma\epsilon_k$$

and

$$z_k = \sum_{i=0}^{mk-1} \Gamma^{k-1+\ell i} b_0 + \Gamma^k d_0.$$

Proof. First expand d_k:

$$\begin{aligned}
d_k &= v_* - v_k + \epsilon_k \\
&= v_* - (T_{k,\ell})^m T_k v_{k-1} \\
&= v_* - T_k v_{k-1} + T_k v_{k-1} - T_{k,\ell} T_k v_{k-1} + T_{k,\ell} T_k v_{k-1} - (T_{k,\ell})^2 T_k v_{k-1} \\
&\quad + (T_{k,\ell})^2 T_k v_{k-1} - \cdots - (T_{k,\ell})^{m-1} T_k v_{k-1} + (T_{k,\ell})^{m-1} T_k v_{k-1} - (T_{k,\ell})^m T_k v_{k-1} \\
&= v_* - T_k v_{k-1} + \sum_{i=0}^{m-1} (T_{k,\ell})^i T_k v_{k-1} - (T_{k,\ell})^{i+1} T_k v_{k-1} \\
&= T_* v_* - T_k v_{k-1} + \sum_{i=0}^{m-1} (\gamma^\ell P_{k,\ell})^i \left(T_k v_{k-1} - T_{k,\ell} T_k v_{k-1}\right) && \{(6)\} \\
&\le T_* v_* - T_* v_{k-1} + \sum_{i=0}^{m-1} (\gamma^\ell P_{k,\ell})^i b_{k-1} && \{T_k v_{k-1} \ge T_* v_{k-1}\ (5)\} \\
&= \gamma P_* (v_* - v_{k-1}) + \sum_{i=0}^{m-1} (\gamma^\ell P_{k,\ell})^i b_{k-1} && \{(6)\} \\
&= \gamma P_* d_{k-1} - \gamma P_* \epsilon_{k-1} + \sum_{i=0}^{m-1} (\gamma^\ell P_{k,\ell})^i b_{k-1} && \{d_k = v_* - v_k + \epsilon_k\} \\
&= \Gamma d_{k-1} + y_{k-1} + \sum_{i=0}^{m-1} \Gamma^{\ell i} b_{k-1}.
\end{aligned}$$

Then, by induction

$$d_k \le \sum_{j=0}^{k-1} \Gamma^{k-1-j} \left(y_j + \sum_{p=0}^{m-1} \Gamma^{\ell p} b_j \right) + \Gamma^k d_0.$$

Using the bound on b_k from Lemma 2 we get:

$$\begin{aligned}
d_k &\le \sum_{j=0}^{k-1} \Gamma^{k-1-j} \left(y_j + \sum_{p=0}^{m-1} \Gamma^{\ell p} \left(\sum_{i=1}^{j} \Gamma^{(\ell m+1)(j-i)} x_i + \Gamma^{(\ell m+1)j} b_0 \right) \right) + \Gamma^k d_0 \\
&= \sum_{j=0}^{k-1} \sum_{p=0}^{m-1} \sum_{i=1}^{j} \Gamma^{k-1-j+\ell p+(\ell m+1)(j-i)} x_i + \sum_{j=0}^{k-1} \sum_{p=0}^{m-1} \Gamma^{k-1-j+\ell p+(\ell m+1)j} b_0 + \Gamma^k d_0 + \sum_{i=1}^{k} \Gamma^{i-1} y_{k-i}.
\end{aligned}$$

First we have:

$$\begin{aligned}
\sum_{j=0}^{k-1} \sum_{p=0}^{m-1} \sum_{i=1}^{j} \Gamma^{k-1-j+\ell p+(\ell m+1)(j-i)} x_i &= \sum_{i=1}^{k-1} \sum_{j=i}^{k-1} \sum_{p=0}^{m-1} \Gamma^{k-1+\ell(p+mj)-i(\ell m+1)} x_i \\
&= \sum_{i=1}^{k-1} \sum_{j=0}^{m(k-i)-1} \Gamma^{k-1+\ell(j+mi)-i(\ell m+1)} x_i \\
&= \sum_{i=1}^{k-1} \sum_{j=0}^{m(k-i)-1} \Gamma^{\ell j+k-i-1} x_i \\
&= \sum_{i=1}^{k-1} \sum_{j=0}^{mi-1} \Gamma^{\ell j+i-1} x_{k-i}.
\end{aligned}$$

Second we have:

$$\sum_{j=0}^{k-1}\sum_{p=0}^{m-1}\Gamma^{k-1-j+\ell p+(\ell m+1)j}b_0=\sum_{j=0}^{k-1}\sum_{p=0}^{m-1}\Gamma^{k-1+\ell(p+mj)}b_0=\sum_{i=0}^{mk-1}\Gamma^{k-1+\ell i}b_0=z_k-\Gamma^k d_0.$$

Hence

$$d_k\le\sum_{i=1}^{k}\sum_{j=0}^{mi-1}\Gamma^{\ell j+i-1}x_{k-i}+\sum_{i=1}^{k}\Gamma^{i-1}y_{k-i}+z_k.$$

□

Lemma 4 (shift bound)**.** *The shift s_k is bounded by:*

$$s_k\le\sum_{i=1}^{k-1}\sum_{j=mi}^{\infty}\Gamma^{\ell j+i-1}x_{k-i}+w_k,$$

where

$$w_k=\sum_{j=mk}^{\infty}\Gamma^{\ell j+k-1}b_0.$$

Proof. Expanding s_k we obtain:

$$\begin{aligned}
s_k&=v_k-v_{\pi_{k,\ell}}-\epsilon_k\\
&=(T_{k,\ell})^mT_kv_{k-1}-v_{\pi_{k,\ell}}\\
&=(T_{k,\ell})^mT_kv_{k-1}-(T_{k,\ell})^\infty T_{k,\ell}T_kv_{k-1} && \{\forall f:\ v_{\pi_{k,\ell}}=(T_{k,\ell})^\infty f\}\\
&=(\gamma^\ell P_{k,\ell})^m\sum_{j=0}^{\infty}(\gamma^\ell P_{k,\ell})^j\left(T_kv_{k-1}-T_{k,\ell}T_kv_{k-1}\right)\\
&=\Gamma^{\ell m}\sum_{j=0}^{\infty}\Gamma^{\ell j}b_{k-1}\\
&=\sum_{j=0}^{\infty}\Gamma^{\ell m+\ell j}b_{k-1}.
\end{aligned}$$

Plugging the bound on b_k of Lemma 2 we get:

$$\begin{aligned}
s_k&\le\sum_{j=0}^{\infty}\Gamma^{\ell m+\ell j}\left(\sum_{i=1}^{k-1}\Gamma^{(\ell m+1)(k-1-i)}x_i+\Gamma^{(\ell m+1)(k-1)}b_0\right)\\
&=\sum_{j=0}^{\infty}\sum_{i=1}^{k-1}\Gamma^{\ell m+\ell j+(\ell m+1)(k-1-i)}x_i+\sum_{j=0}^{\infty}\Gamma^{\ell m+\ell j+(\ell m+1)(k-1)}b_0\\
&=\sum_{j=0}^{\infty}\sum_{i=1}^{k-1}\Gamma^{\ell(j+mi)+i-1}x_{k-i}+\sum_{j=0}^{\infty}\Gamma^{\ell(j+mk)+k-1}b_0\\
&=\sum_{i=1}^{k-1}\sum_{j=mi}^{\infty}\Gamma^{\ell j+i-1}x_{k-i}+\sum_{j=mk}^{\infty}\Gamma^{\ell j+k-1}b_0\\
&=\sum_{i=1}^{k-1}\sum_{j=mi}^{\infty}\Gamma^{\ell j+i-1}x_{k-i}+w_k.
\end{aligned}$$

□

Lemma 5 (loss bound)**.** *The loss l_k is bounded by:*

$$l_k \leq \sum_{i=1}^{k-1} \Gamma^i \left(\sum_{j=0}^{\infty} \Gamma^{\ell j} (I - \Gamma^\ell) - I \right) \epsilon_{k-i} + \eta_k,$$

where

$$\eta_k = z_k + w_k = \sum_{i=0}^{mk-1} \Gamma^{k-1+\ell i} b_0 + \Gamma^k d_0 + \sum_{j=mk}^{\infty} \Gamma^{\ell j + k - 1} b_0 = \sum_{i=0}^{\infty} \Gamma^{\ell i + k - 1} b_0 + \Gamma^k d_0.$$

Proof. Using Lemmas 3 and 4, we have:

$$\begin{aligned}
l_k &= s_k + d_k \\
&\leq \sum_{i=1}^{k-1} \sum_{j=mi}^{\infty} \Gamma^{\ell j + i - 1} x_{k-i} + \sum_{i=1}^{k-1} \sum_{j=0}^{mi-1} \Gamma^{\ell j + i - 1} x_{k-i} + \sum_{i=1}^{k} \Gamma^{i-1} y_{k-i} + z_k + w_k \\
&= \sum_{i=1}^{k-1} \sum_{j=0}^{\infty} \Gamma^{\ell j + i - 1} x_{k-i} + \sum_{i=1}^{k} \Gamma^{i-1} y_{k-i} + \eta_k.
\end{aligned}$$

Plugging back the values of x_k and y_k and using the fact that $\epsilon_0 = 0$ we obtain:

$$\begin{aligned}
l_k &\leq \sum_{i=1}^{k-1} \sum_{j=0}^{\infty} \Gamma^{\ell j + i - 1} (I - \Gamma^\ell) \Gamma \epsilon_{k-i} + \sum_{i=1}^{k-1} \Gamma^{i-1} (-\Gamma) \epsilon_{k-i} - \Gamma^k \epsilon_0 + \eta_k \\
&= \sum_{i=1}^{k-1} \left(\sum_{j=0}^{\infty} \Gamma^{\ell j + i} (I - \Gamma^\ell) \epsilon_{k-i} - \Gamma^i \epsilon_{k-i} \right) + \eta_k \\
&= \sum_{i=1}^{k-1} \Gamma^i \left(\sum_{j=0}^{\infty} \Gamma^{\ell j} (I - \Gamma^\ell) - I \right) \epsilon_{k-i} + \eta_k.
\end{aligned}$$

□

We now provide a bound of η_k in terms of d_0:

Lemma 6.

$$\eta_k \leq \Gamma^k \left(\sum_{i=0}^{\infty} \Gamma^i (\Gamma - I) + I \right) d_0.$$

Proof. First recall that

$$\eta_k = \sum_{i=0}^{\infty} \Gamma^{\ell i + k - 1} b_0 + \Gamma^k d_0.$$

In order to bound η_k in terms of d_0 only, we express b_0 in terms of d_0:

$$\begin{aligned}
b_0 &= T_1 v_0 - (T_1)^\ell T_1 v_0 \\
&= T_1 v_0 - (T_1)^2 v_0 + (T_1)^2 v_0 - \cdots - (T_1)^\ell v_0 + (T_1)^\ell v_0 - (T_1)^{\ell+1} v_0 \\
&= \sum_{i=1}^{\ell} (\gamma P_1)^i (v_0 - T_1 v_0) \\
&= \sum_{i=1}^{\ell} (\gamma P_1)^i (v_0 - v_* + T_* v_* - T_* v_0 + T_* v_0 - T_1 v_0) \\
&\leq \sum_{i=1}^{\ell} (\gamma P_1)^i (v_0 - v_* + T_* v_* - T_* v_0) && \{T_1 v_0 \geq T_* v_0 \text{ (5)}\} \\
&= \sum_{i=1}^{\ell} (\gamma P_1)^i (\gamma P_* - I) d_0.
\end{aligned}$$

Consequently, we have:

$$\begin{aligned}
\eta_k &\leq \sum_{i=0}^{\infty} \Gamma^{\ell i + k - 1} \sum_{j=1}^{\ell} (\gamma P_1)^j (\gamma P_* - I) d_0 + \Gamma^k d_0 \\
&= \sum_{i=0}^{\infty} \Gamma^{\ell i + k} \sum_{j=0}^{\ell - 1} (\gamma P_1)^j (\gamma P_* - I) d_0 + \Gamma^k d_0 \\
&= \Gamma^k \left(\sum_{i=0}^{\infty} \Gamma^{\ell i} \sum_{j=0}^{\ell - 1} \Gamma^j (\Gamma - I) + I \right) d_0 \\
&= \Gamma^k \left(\sum_{i=0}^{\infty} \Gamma^i (\Gamma - I) + I \right) d_0.
\end{aligned}$$

□

We now conclude the proof of Theorem 3. Taking the absolute value in Lemma 6 we obtain:

$$|\eta_k| \leq \Gamma^k \left(\sum_{i=0}^{\infty} \Gamma^i (\Gamma + I) + I \right) |d_0| = 2 \sum_{i=k}^{\infty} \Gamma^i |d_0|$$

Since l_k is non-negative, from Lemma 5 we have:

$$|l_k| \leq \sum_{i=1}^{k-1} \Gamma^i \left(\sum_{j=0}^{\infty} \Gamma^{\ell j} (I + \Gamma^\ell) + I \right) |\epsilon_{k-i}| + |\eta_k| = 2 \sum_{i=1}^{k-1} \Gamma^i \sum_{j=0}^{\infty} \Gamma^{\ell j} |\epsilon_{k-i}| + 2 \sum_{i=k}^{\infty} \Gamma^i |d_0|. \tag{8}$$

Since $\|v\|_\infty = \max |v|$, $d_0 = v_* - v_0$ and $l_k = v_* - v_{\pi_{k,\ell}}$, we can take the maximum in (8) and conclude that:

$$\left\| v_* - v_{\pi_{k,\ell}} \right\|_\infty \leq \frac{2(\gamma - \gamma^k)}{(1-\gamma)(1-\gamma^\ell)} 2\epsilon + \frac{\gamma^k}{1-\gamma} \|v_* - v_0\|_\infty.$$

B. Proof of Theorem 4

We shall prove the following result.

Lemma 7. *Consider NS-AMPI with parameters $m \geq 0$ and $\ell \geq 1$ applied on the problem of Figure 1, starting from $v_0 = 0$ and all initial policies $\pi_0, \pi_{-1}, \ldots, \pi_{-\ell+2}$ equal to π_*. Assume that at each iteration k, the following error terms are applied, for some $\epsilon \geq 0$:*

$$\forall i, \;\; \epsilon_k(i) = \begin{cases} -\epsilon & \text{if } i = k \\ \epsilon & \text{if } i = k + \ell \\ 0 & \text{otherwise} \end{cases}.$$

Then NS-AMPI can[8] generate a sequence of value-policy pairs that is described below.

For all iterations $k \geq 1$, the policy π_k takes the optimal action in all states but k, that is

$$\forall i \geq 2, \;\; \pi_k(i) = \begin{cases} \rightarrow & \text{if } i = k \\ \leftarrow & \text{otherwise} \end{cases} \tag{9}$$

For all iterations $k \geq 1$, the value function v_k satisfies the following equations:

- *For all $i < k$:*
$$v_k(i) = -\gamma^{(k-1)(\ell m+1)}\epsilon \tag{10.a}$$
- *For all i such that $k \leq i \leq k + ((k-1)m+1)\ell$:*
  - *For $i = k + (qm + p + 1)\ell$ with $q \geq 0$ and $0 \leq p < m$ (i.e. $i = k + n\ell$, $n \geq 1$):*
$$v_k(i) = \gamma^{q(\ell m+1)}\left(\frac{\gamma^{\ell(p+1)} - \gamma^{\ell(m+1)}}{1-\gamma^\ell} r_{k-q} + \mathbb{1}_{[p=0]}\epsilon + \sum_{j=1}^{k-q-1} \gamma^{j(\ell m+1)}\left(\frac{\gamma^\ell - \gamma^{\ell(m+1)}}{1-\gamma^\ell} r_{k-q-j} + \epsilon\right)\right) \tag{10.b}$$
  - *For $i = k$:*
$$v_k(k) = v_k(k+\ell) + r_k - 2\epsilon \tag{10.c}$$
  - *For $i = k + q\ell + p$ with $0 \leq q \leq (k-1)m - 1$ and $1 \leq p < \ell$:*
$$v_k(i) = -\gamma^{(k-1)(\ell m+1)}\epsilon \tag{10.d}$$
  - *Otherwise, i.e. when $i = k + (k-1)m\ell + p$ with $1 \leq p < \ell$:*
$$v_k(i) = 0 \tag{10.e}$$
- *For all $i > k + ((k-1)m+1)\ell$*
$$v_k(i) = 0 \tag{10.f}$$

The relative complexity of the different expressions of v_k in Lemma 7 is due to the presence of nested periodic patterns in the shape of the value function along the state space and the horizon. Figures 4 and 5 give the shape of the value function for different values of ℓ and m, exhibiting the periodic patterns. The proof of Lemma 7 is done by recurrence on k.

B.1. Base case $k = 1$

Since $v_0 = 0$, π_1 is the optimal policy that takes $\leftarrow$ in all states as desired. Hence, $(T_{1,\ell})^m T_1 v_0 = 0$ in all states. Accounting for the errors ϵ_1 we have $v_1 = (T_{1,\ell})^m T_1 v_0 + \epsilon_1 = \epsilon_1$. As can be seen on Figures 4 and 5, when $k = 1$ we only need to consider equations (10.b), (10.c), (10.e) and (10.f) since the others apply to an empty set of states.

First, we have

$$v_1(1+\ell) = \epsilon_1(1+\ell) = \epsilon$$

[8] We write here "can" since at each iteration, several policies will be greedy with respect to the current value.

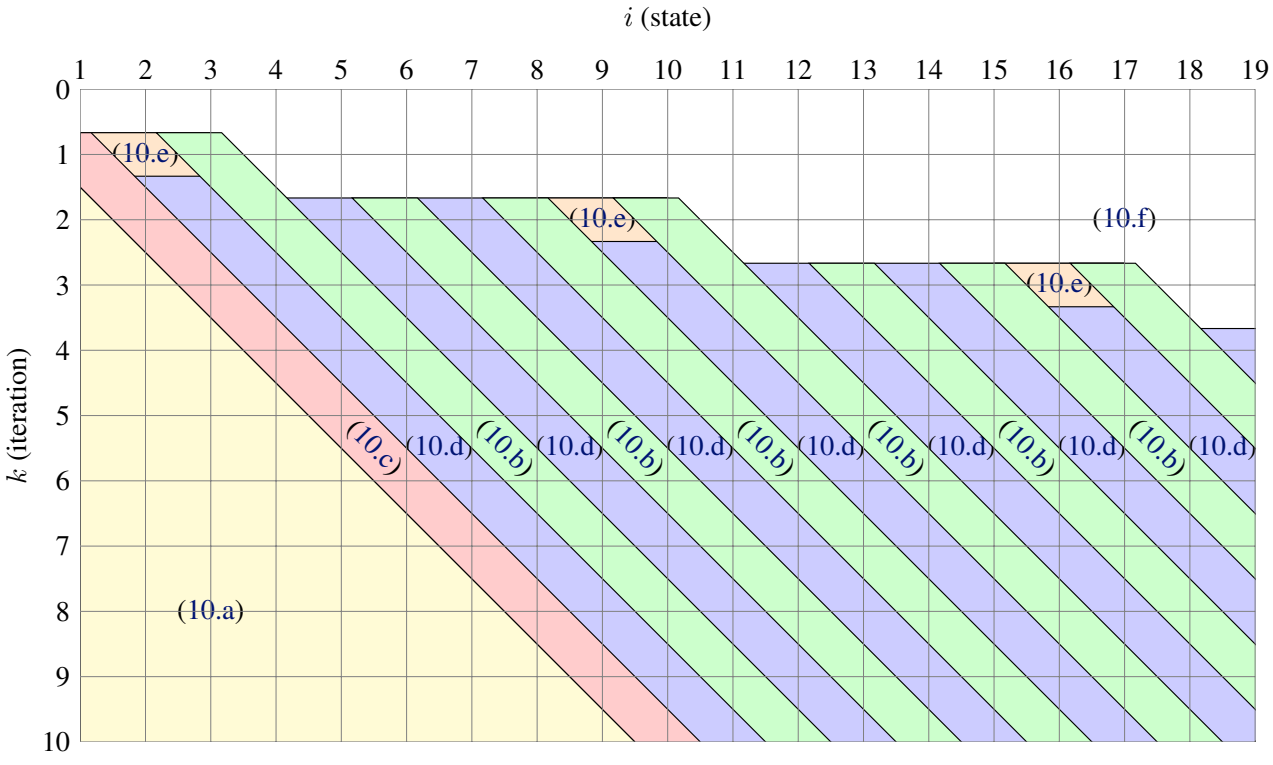

Figure 4. Shape of the value function with $\ell = 2$ and $m = 3$.

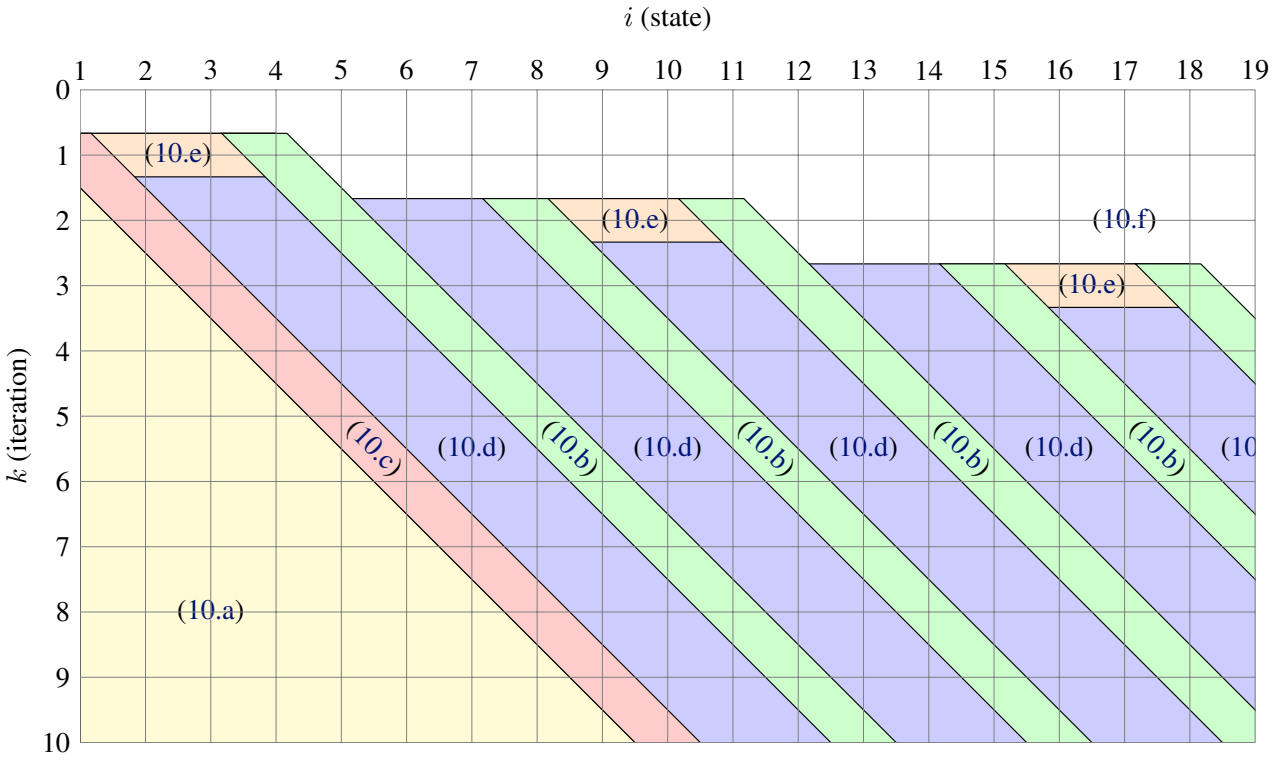

Figure 5. Shape of the value function with $\ell = 3$ and $m = 2$.

which is (10.b) when $q = (k-1) = 0$ and $p = 0$.

Second, we have

$$v_1(1) = \epsilon_1(1) = -\epsilon = \epsilon + 0 - 2\epsilon = v_1(1+\ell) + r_1 - 2\epsilon$$

which corresponds to (10.c).

Third, for $1 \leq p < \ell$ we have

$$v_1(1+p) = \epsilon_1(1+p) = 0$$

corresponding to (10.e).

Finally, for all the remaining states $i > 1 + \ell$, we have

$$v_1(i) = \epsilon_1(i) = 0$$

corresponding to (10.f).

The base case is now proved.

B.2. Induction Step

We assume that Lemma 7 holds for some *fixed* $k \geq 1$, we now show that it also holds for $k+1$.

B.2.1. The policy π_{k+1}

We begin by showing that the policy π_{k+1} is greedy with respect to v_k. Since there is no choice in state 1 is $\rightarrow$, we turn our attention to the other states. There are many cases to consider, each one of them corresponding to one or more states. These cases, labelled from A through F, are summarized as follows, depending on the state i:

(A) $1 < i < k+1$

(B) $i = k+1$

(C) $i = k+1+q\ell+p$ with $1 \leq p < \ell$ and $0 \leq q \leq (k-1)m$

(D) $i = k+1+(qm+p+1)\ell$ with $0 \leq p < m$ and $0 \leq q < k-1$

(E) $i = k+1+((k-1)m+1)\ell$

(F) $i > k+1+((k-1)m+1)\ell$

Figure 6 depicts how those cases cover the whole state space.

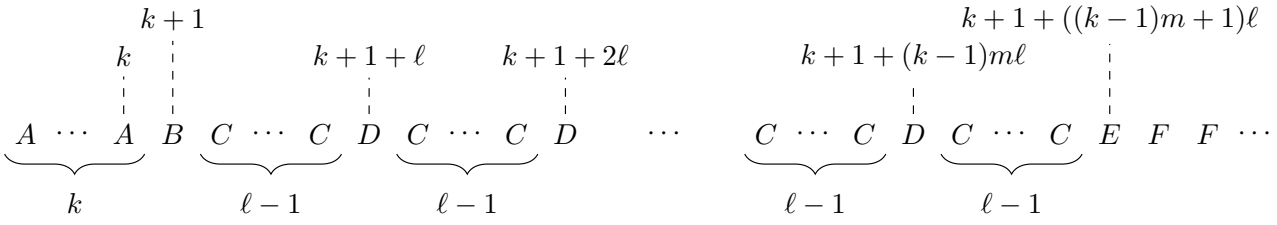

Figure 6. Policy cases, each state is represented by a letter corresponding to a case of the policy π_{k+1}. Starting from 1, state number increase from left to right.

For all states $i > 1$ in each of the above cases, we consider the *action-value functions* $q^{\rightarrow}_{k+1}(i)$ (resp. $q^{\leftarrow}_{k+1}(i)$) of action $\rightarrow$ (resp. $\leftarrow$) defined as:

$$q^{\rightarrow}_{k+1}(i) = r_i + \gamma v_k(i-1) \qquad \text{and} \qquad q^{\leftarrow}_{k+1}(i) = \gamma v_k(i+\ell-1).$$

In case $i = k+1$ (B) we will show that $q^{\rightarrow}_{k+1}(i) = q^{\leftarrow}_{k+1}(i)$ meaning that a policy π_{k+1} greedy for v_k may be either $\pi_{k+1}(k+1) = \rightarrow$ or $\pi_{k+1}(k+1) = \leftarrow$. In all other cases we show that $q^{\rightarrow}_{k+1}(i) < q^{\leftarrow}_{k+1}(i)$ which implies that for those $i \neq k+1$, $\pi_{k+1}(i) = \leftarrow$, as required by Lemma 7.

A: In states $1 < i < k+1$ We have $q^{\rightarrow}_{k+1}(i) = r_i + \gamma v_k(i+\ell-1)$ and $q^{\leftarrow}_{k+1}(i) = \gamma v_k(i-1)$, depending on the value of $i+\ell-1$, which is reached by taking the $\rightarrow$ action, we need to consider two cases:

- Case 1: $i+\ell-1 \neq k$. In this case $v_k(i+\ell-1)$ is described by either (10.a) or (10.d) when $i+\ell-1$ is less than, or greater than k, respectively. In either case we have $v_k(i+\ell-1) = -\gamma^{(k-1)(\ell m+1)}\epsilon = v_k(i-1)$ and hence:

$$q^{\rightarrow}_{k+1}(i) = r_i + \gamma v_k(i+\ell-1) = r_i + \gamma v_k(i-1) < \gamma v_k(i-1) = q^{\leftarrow}_{k+1}(i)$$

  which gives $\pi_{k+1}(i) = \leftarrow$ as desired.

- Case 2: $i+\ell-1 = k$.

$$\begin{aligned}
q^{\rightarrow}_{k+1}(i) &= r_i + \gamma v_k(k) = r_i + \gamma\left(v_k(k+\ell) + r_k - 2\epsilon\right) && \{(10.c)\}\\
&= \gamma\left(\sum_{j=0}^{k-1}\gamma^{j(\ell m+1)}\left(\frac{\gamma^\ell - \gamma^{\ell(m+1)}}{1-\gamma^\ell}r_{k-j} + \epsilon\right) + r_k - 2\epsilon\right) && \{(10.b)\}\\
&\leq \gamma\left(\sum_{j=0}^{k-1}\gamma^{j(\ell m+1)}\epsilon + r_k - 2\epsilon\right) && \{r_{k-j} \leq 0\}\\
&= \gamma\left(\sum_{j=1}^{k-1}\left(\gamma^{j(\ell m+1)}\epsilon - 2\gamma^j\epsilon\right) - \epsilon\right) && \{r_k = -2\sum_{j=1}^{k-1}\gamma^j\epsilon\}\\
&< -\gamma\epsilon && \{\gamma^{j(\ell m+1)}\epsilon - 2\gamma^j\epsilon < 0\}\\
&< \gamma v_k(i-1) && \{v_k(i-1) = -\gamma^{(k-1)(\ell m+1)}\epsilon\ (10.a)\}\\
&= q^{\leftarrow}_{k+1}(i)
\end{aligned}$$

  giving $\pi_{k+1}(i) = \leftarrow$ as desired.

B: In state $k+1$ Looking at the action value function $q^{\leftarrow}_{k+1}$ in state $k+1$, we observe that:

$$\begin{aligned}
q^{\leftarrow}_{k+1}(k+1) &= \gamma v_k(k) = \gamma\left(r_k - 2\epsilon + v_k(k+\ell)\right) && \{(10.c)\}\\
&= \gamma r_k - 2\gamma\epsilon + \gamma v_k(k+\ell)\\
&= r_{k+1} + \gamma v_k(k+\ell) && \{r_{i+1} = \gamma r_i - 2\gamma\epsilon\}\\
&= q^{\rightarrow}_{k+1}(k+1)
\end{aligned}$$

This means that the algorithm can take $\pi_{k+1}(k+1) = \rightarrow$ so as to satisfy Lemma 7.

C: In states $i = k+1+q\ell+p$ We restrict ourselves to the cases when $1 \leq p < \ell$ and $0 \leq q \leq (k-1)m$. Three cases for the value of q need to be considered:

- Case 1: $0 \leq q < (k-1)m - 1$. We have:

$$\begin{aligned}
q^{\rightarrow}_{k+1}(i) &= r_i + \gamma v_k(k+(q+1)\ell+p)\\
&= r_i + \gamma v_k(k+q\ell+p) && \{(10.d) \text{ independent of } q\}\\
&< \gamma v_k(k+q\ell+p) && \{r_i < 0\}\\
&= q^{\leftarrow}_{k+1}(i).
\end{aligned}$$

- Case 2: $q = (k-1)m - 1$

$$\begin{aligned}
q^{\rightarrow}_{k+1}(i) &= r_i + \gamma v_k(k + (q+1)\ell + p) \\
&= r_i + \gamma 0 && \{(10.e)\} \\
&= -2\epsilon \frac{\gamma - \gamma^{k+1+q\ell+p}}{1-\gamma} \\
&= -2\epsilon \left(\frac{\gamma - \gamma^{k+q\ell+p}}{1-\gamma} + \gamma^{k+q\ell+p} \right) \\
&< -\gamma^{k+q\ell+p}\epsilon \\
&= -\gamma^{k+(k-1)\ell m - \ell + p}\epsilon && \{q = (k-1)m - 1\} \\
&< -\gamma^{k+(k-1)\ell m}\epsilon = -\gamma^{(k-1)(\ell m+1)+1}\epsilon && \{p - \ell < 0\} \\
&= \gamma v_k(k + q\ell + p) && \{(10.d)\} \\
&= q^{\leftarrow}_{k+1}(i).
\end{aligned}$$

- Case 3: $q = (k-1)m$

$$\begin{aligned}
q^{\rightarrow}_{k+1}(i) &= r_i + \gamma v_k(k + ((k-1)m+1)\ell + p) \\
&= r_i + \gamma 0 && \{(10.f)\} \\
&= r_i + \gamma v_k(k + ((k-1)m)\ell + p) && \{(10.e)\} \\
&= r_i + \gamma v_k(i-1) \\
&< q^{\leftarrow}_{k+1}(i). && \{r_i < 0\}
\end{aligned}$$

D: In states $i = k + 1 + (qm + p + 1)\ell$ In these states, we have:

$$\begin{aligned}
q^{\leftarrow}_{k+1}(i) &= \gamma v_k(k + (qm + p + 1)\ell) \\
\\
q^{\rightarrow}_{k+1}(i) &= r_i + \gamma v_k(k + 1 + (qm + p + 1)\ell + \ell - 1) \\
&= r_i + \gamma v_k(k + (qm + p + 2)\ell). && (11)
\end{aligned}$$

As for the right-hand side of (11) we need to consider two cases:

- Case 1: $p + 1 < m$:

In the following, define

$$x_{k,q} = \sum_{j=1}^{k-q-1} \gamma^{j(\ell m+1)} \left(\frac{\gamma^{\ell} - \gamma^{\ell(m+1)}}{1-\gamma^{\ell}} r_{k-q-j} + \epsilon \right).$$

Then,

$$\begin{aligned}
&q^{\rightarrow}_{k+1}(i) = r_i + \gamma v_k(k + (qm + (p+1) + 1)\ell) \\
&= r_i + \gamma\gamma^{q(\ell m+1)} \left(\frac{\gamma^{\ell(p+2)} - \gamma^{\ell(m+1)}}{1-\gamma^{\ell}} r_{k-q} + \sum_{j=1}^{k-q-1} \gamma^{j(\ell m+1)} \left(\frac{\gamma^{\ell} - \gamma^{\ell(m+1)}}{1-\gamma^{\ell}} r_{k-q-j} + \epsilon \right) \right) && \{(10.b)\} \\
&= r_i + \gamma^{q(\ell m+1)+1} \left(\left(\frac{\gamma^{\ell(p+1)} - \gamma^{\ell(m+1)}}{1-\gamma^{\ell}} - \gamma^{\ell(p+1)} \right) r_{k-q} + x_{k,q} \right) \\
&= r_i - \gamma^{(qm+p+1)\ell+q+1} r_{k-q} + \gamma^{q(\ell m+1)+1} \left(\frac{\gamma^{\ell(p+1)} - \gamma^{\ell(m+1)}}{1-\gamma^{\ell}} r_{k-q} + x_{k,q} \right) \\
&= r_i - \gamma^{i-k+q} r_{k-q} + \gamma v_k(k + (qm + p + 1)\ell) - \mathbb{1}_{[p=0]} \gamma^{q(\ell m+1)+1}\epsilon && \{(10.b)\} \\
&\leq r_i - \gamma^{i-k+q} r_{k-q} + \gamma v_k(k + (qm + p + 1)\ell) \\
&= r_i - \gamma^{i-k+q} r_{k-q} + q^{\leftarrow}_{k+1}(i). && (12)
\end{aligned}$$

Now, observe that

$$\begin{aligned}
\gamma^{i-k+q} r_{k-q} &= -2\gamma^{i-k+q}\frac{\gamma-\gamma^{k-q}}{1-\gamma}\epsilon \\
&= -2\frac{\gamma^{i-k+q+1}-\gamma^{i}}{1-\gamma}\epsilon \\
&= -2\frac{\gamma-\gamma+\gamma^{i-k+q+1}-\gamma^{i}}{1-\gamma}\epsilon \\
&= -2\frac{\gamma-\gamma^{i}}{1-\gamma}\epsilon - 2\frac{-\gamma+\gamma^{i-k+q+1}}{1-\gamma}\epsilon \\
&= r_i - r_{i-k+q+1}.
\end{aligned}$$

Plugging this back into (12), we get:

$$\begin{aligned}
q^{\rightarrow}_{k+1}(i) &\leq r_i - r_i + r_{i-k+q+1} + q^{\leftarrow}_{k+1}(i) \\
&< q^{\leftarrow}_{k+1}(i). && \{r_{i-k+q+1} < 0\}
\end{aligned}$$

- Case 2: $p+1=m$:

Using the fact that $p+1=m$ implies $\frac{\gamma^{\ell(p+1)}-\gamma^{\ell(m+1)}}{1-\gamma^{\ell}} = \gamma^{\ell m}$ we have:

$$\begin{aligned}
&q^{\rightarrow}_{k+1}(i) = r_i + \gamma v_k(k+((q+1)m+1)\ell) \\
&= r_i + \gamma\gamma^{(q+1)(\ell m+1)}\left(\frac{\gamma^{\ell}-\gamma^{\ell(m+1)}}{1-\gamma^{\ell}} r_{k-q-1} + \epsilon + \sum_{j=1}^{k-q-2}\gamma^{j(\ell m+1)}\left(\frac{\gamma^{\ell}-\gamma^{\ell(m+1)}}{1-\gamma^{\ell}} r_{k-q-j-1} + \epsilon\right)\right) && \{(10.b)\} \\
&= r_i + \gamma\gamma^{(q+1)(\ell m+1)}\left(\sum_{j=0}^{k-q-2}\gamma^{j(\ell m+1)}\left(\frac{\gamma^{\ell}-\gamma^{\ell(m+1)}}{1-\gamma^{\ell}} r_{k-q-j-1} + \epsilon\right)\right) \\
&= r_i + \gamma\gamma^{q(\ell m+1)}\left(\sum_{j=1}^{k-q-1}\gamma^{j(\ell m+1)}\left(\frac{\gamma^{\ell}-\gamma^{\ell(m+1)}}{1-\gamma^{\ell}} r_{k-q-j} + \epsilon\right)\right) \\
&= r_i + \gamma\gamma^{q(\ell m+1)}\left(\left(\frac{\gamma^{\ell(p+1)}-\gamma^{\ell(m+1)}}{1-\gamma^{\ell}} - \gamma^{\ell m}\right) r_{k-q} + \sum_{j=1}^{k-q-1}\gamma^{j(\ell m+1)}\left(\frac{\gamma^{\ell}-\gamma^{\ell(m+1)}}{1-\gamma^{\ell}} r_{k-q-j} + \epsilon\right)\right) \\
&= r_i - \gamma^{q(\ell m+1)+1}\gamma^{\ell m} r_{k-q} + \gamma\left(v_k(k+(qm+p+1)\ell) - \mathbb{1}_{[p=0]}\gamma^{q(\ell m+1)}\epsilon\right) && \{(10.b)\} \\
&\leq r_i - \gamma^{i-k+q} r_{k-q} + \gamma v_k(k+(qm+p+1)\ell) \\
&< q^{\leftarrow}_{k+1}(i),
\end{aligned}$$

where we concluded by observing that this is the same result as (12).

E: In state $i = k+((k-1)m+1)\ell+1$

$$\begin{aligned}
q^{\leftarrow}_{k+1}(i) &= \gamma v_k(i-1) = \gamma v_k(k+((k-1)m+1)\ell) \\
&= \gamma^{(k-1)(\ell m+1)+1}\left(\frac{\gamma^{\ell}-\gamma^{\ell(m+1)}}{1-\gamma^{\ell}} r_1 + \epsilon\right) && \{(10.b) \text{ with } q=k-1 \text{ and } p=0\} \\
&= \gamma^{(k-1)(\ell m+1)+1}\epsilon && \{r_1 = 0\} \\
&> r_i && \{r_i < 0\} \\
&= r_i + \gamma v_k(i+\ell-1) && \{v_k(i+\ell+1) = 0 \ \ (10.f)\} \\
&= q^{\rightarrow}_{k+1}(i).
\end{aligned}$$

F: In states $i > k + ((k-1)m+1)\ell + 1$ Following (10.f) we have $v_k(i-1) = v_k(i+\ell-1) = 0$ and hence

$$q^{\leftarrow}_{k+1}(i) = 0 > r_i = q^{\rightarrow}_{k+1}(i).$$

B.2.2. THE VALUE FUNCTION v_{k+1}

In the following we will show that the value function v_{k+1} satisfies Lemma 7. To that end we consider the value of $((T_{k+1,\ell})^m T_{k+1} v_k)(s_0)$ by analysing the trajectories obtained by first following m times $\pi_{k,\ell}$ then π_{k+1} from various starting states s_0.

Given a starting state s_0 and a non stationary policy $\pi_{k+1,\ell}$, we will represent the trajectories as a sequence of triples $(s_i, a_i, r(s_i, a_i))_{i=0,\dots,\ell m}$ arranged in a "trajectory matrix" of ℓ columns and m rows. Each column corresponds to one of the policies $\pi_{k+1}, \pi_k, \dots, \pi_{k+2-\ell}$. In a column labeled by policy π_j the entries are of the form $(s_i, \pi_j(s_i), r(s_i, \pi_j(s_i))$; this layout makes clear which stationary policy is used to select the action in any particular step in the trajectory. Indeed, in column π_j, we have $(s_i, \rightarrow, r_j)$ if and only if $s_i = j$, otherwise each entry is of the form $(s_i, \leftarrow, 0)$. Such a matrix accounts for the first m applications of the operator $T_{k+1,\ell}$. One addional row of only one triple $(s_i, \pi_{k+1}(s_i), r_{\pi_{k+1}}(s_i))$ represents the final application of T_{k+1}. After this triple comes the end state of the trajectory $s_{\ell m+1}$.

$$\begin{array}{r} \\ m = 4 \text{ times} \end{array} \overbrace{\left\{\begin{array}{ccc} \pi_4 & \pi_3 & \pi_2 \\ (10, \leftarrow, 0) & (9, \leftarrow, 0) & (8, \leftarrow, 0) \\ (7, \leftarrow, 0) & (6, \leftarrow, 0) & (5, \leftarrow, 0) \\ (4, \rightarrow, r_4) & (6, \leftarrow, 0) & (5, \leftarrow, 0) \\ (4, \rightarrow, r_4) & (6, \leftarrow, 0) & (5, \leftarrow, 0) \\ (4, \rightarrow, r_4) & \boxed{6} & \end{array}\right.}^{\ell = 3 \text{ steps}}$$

Figure 7. The trajectory matrix of policy $\pi_{4,\ell}$ starting from state 10 with $m = 4$ and $\ell = 3$.

Example 2. *Figure 7 depicts the trajectory matrix of policy $\pi_{4,\ell} = \pi_4\pi_3\pi_2$ with $m = 4$ and $\ell = 3$. The trajectory starts from state $s_0 = 10$ and ends in state $s_{\ell m+1} = 6$. The $\leftarrow$ action is always taken with reward 0 except when in state 4 under the policy π_4. From this matrix we can deduce that, for any value function v:*

$$\begin{aligned} ((T_{4,\ell})^m T_4 v)(10) &= \gamma^6 r_4 + \gamma^9 r_4 + \gamma^{12} r_4 + \gamma^{13} v(6) \\ &= \gamma^{2\ell} r_4 + \gamma^{3\ell} r_4 + \gamma^{4\ell} r_4 + \gamma^{4\ell+1} v(6) \\ &= \frac{\gamma^{2\ell} - \gamma^{(m+1)\ell}}{1-\gamma^\ell} r_4 + \gamma^{\ell m+1} v(6). \end{aligned}$$

With this in hand, we are going to prove each case of Lemma 7 for v_{k+1}.

In states $i < k+1$ Following m times $\pi_{k+1,\ell}$ and then π_{k+1} starting from these states consists in taking the $\leftarrow$ action $\ell m + 1$ times to eventually finish either in state 1 if $i \leq \ell m + 2$ with value

$$v_{k+1}(i) = \gamma^{\ell m+1} v_k(1) + \epsilon_{k+1}(i) = -\gamma^{\ell m+1}\gamma^{(k-1)(\ell m+1)}\epsilon = -\gamma^{k(\ell m+1)}\epsilon$$

or otherwise in state $i - \ell m - 1 < k$ with value

$$v_{k+1}(i) = \gamma^{\ell m+1} v_k(i - \ell m - 1) + \epsilon_{k+1}(i) = -\gamma^{\ell m+1}\gamma^{(k-1)(\ell m+1)}\epsilon = -\gamma^{k(\ell m+1)}\epsilon$$

This matches Equation (10.a) in both cases.

In states $i = k + 1 + (qm + p + 1)\ell$ Consider the states $i = k + 1 + (qm + p + 1)\ell$ with $q \geq 0$ and $0 \leq p < m$. Following m times $\pi_{k+1,\ell}$ and then π_{k+1} starting from state i gives the following trajectories:

- when $q = 0$, (*i.e.* $i = k + 1 + (p + 1)\ell$):

$$
\begin{array}{rc}
 & \overbrace{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}^{\ell \text{ steps}} \\
 & \begin{array}{cccc} \pi_{k+1} & \pi_k & \dots & \pi_{k-\ell+2} \end{array} \\
p+1 \text{ times} \left\{ \right. & \begin{array}{cccc} (k+1+(p+1)\ell, \leftarrow, 0) & (k+(p+1)\ell, \leftarrow, 0) & \dots & (k+p\ell+2, \leftarrow, 0) \\ (k+1+p\ell, \leftarrow, 0) & (k+p\ell, \leftarrow, 0) & \dots & (k+(p-1)\ell+2, \leftarrow, 0) \\ \vdots & \vdots & \vdots & \vdots \\ (k+1+\ell, \leftarrow, 0) & (k+\ell, \leftarrow, 0) & \dots & (k+2, \leftarrow, 0) \end{array} \\
m-p-1 \text{ times} \left\{ \right. & \begin{array}{cccc} (k+1, \rightarrow, r_{k+1}) & (k+\ell, \leftarrow, 0) & \dots & (k+2, \leftarrow, 0) \\ \vdots & \vdots & \vdots & \vdots \\ (k+1, \rightarrow, r_{k+1}) & (k+\ell, \leftarrow, 0) & \dots & (k+2, \leftarrow, 0) \end{array} \\
 & \begin{array}{cc} (k+1, \rightarrow, r_{k+1}) & \boxed{k+\ell} \end{array}
\end{array}
$$

Using (10.b) with $q = p = 0$ as our induction hypothesis, this gives

$$
\begin{aligned}
\left((T_{k+1,\ell})^m T_{k+1} v_k\right)(i) &= \sum_{j=p+1}^{m} \gamma^{\ell j} r_{k+1} + \gamma^{\ell m+1} v_k(k+\ell) \\
&= \sum_{j=p+1}^{m} \gamma^{\ell j} r_{k+1} + \gamma^{\ell m+1} \left(\frac{\gamma^\ell - \gamma^{\ell(m+1)}}{1-\gamma^\ell} r_k + \epsilon + \sum_{j=1}^{k-1} \gamma^{j(\ell m+1)} \left(\frac{\gamma^\ell - \gamma^{\ell(m+1)}}{1-\gamma^\ell} r_{k-j} + \epsilon \right) \right) \\
&= \frac{\gamma^{\ell(p+1)} - \gamma^{\ell(m+1)}}{1-\gamma^\ell} r_{k+1} + \sum_{j=1}^{k} \gamma^{j(\ell m+1)} \left(\frac{\gamma^\ell - \gamma^{\ell(m+1)}}{1-\gamma^\ell} r_{k-j} + \epsilon \right)
\end{aligned}
$$

Accounting for the error term and the fact that $i = k + 1 + \ell \iff p = q = 0$, we get

$$
\begin{aligned}
v_{k+1}(i) &= \left((T_{k+1,\ell})^m T_{k+1} v_k\right)(i) + \mathbb{1}_{[i=k+1+\ell]}\epsilon \\
&= \frac{\gamma^{\ell(p+1)} - \gamma^{\ell(m+1)}}{1-\gamma^\ell} r_{k+1} + \mathbb{1}_{[p=0]}\epsilon + \sum_{j=1}^{k} \gamma^{j(\ell m+1)} \left(\frac{\gamma^\ell - \gamma^{\ell(m+1)}}{1-\gamma^\ell} r_{k-j} + \epsilon \right)
\end{aligned}
$$

which is (10.b) for $k + 1$ and $q = 0$ as desired.

- when $1 \leq q \leq k$:

In this case we have $i - (\ell m + 1) \geq k + 1$, meaning that $k + 1$, the first state where the $\rightarrow$ action would be available is unreachable (in the sense that the tractory could end in $k + 1$, but no action will be taken there). Consequently the $\leftarrow$ action is taken $\ell m + 1$ times and the system ends in state $i - \ell m - 1 = k + ((q - 1)m + p + 1)\ell$. Therefore, using (10.b) as induction hypothesis and the fact that $i \notin \{k + 1, k + \ell + 1\} \implies \epsilon_{k+1}(i) = 0$, we have:

$$
\begin{aligned}
v_{k+1}(i) &= \gamma^{\ell m+1} v_k(k + ((q-1)m + p + 1)\ell) + \epsilon_{k+1}(i) \\
&= \gamma^{q(\ell m+1)} \left(\frac{\gamma^{\ell(p+1)} - \gamma^{\ell(m+1)}}{1-\gamma^\ell} r_{k+1-q} + \mathbb{1}_{[p=0]}\epsilon + \sum_{i=1}^{k-q} \gamma^{i(\ell m+1)} \left(\frac{\gamma^\ell - \gamma^{\ell(m+1)}}{1-\gamma^\ell} r_{k+1-q-k} + \epsilon \right) \right),
\end{aligned}
$$

which statisfies (10.b) for $k + 1$.

In state $k+1$ Following m times $\pi_{k+1,\ell}$ and then π_{k+1} starting from $k+1$ gives the following trajectory:

$$
\begin{array}{c}
\overbrace{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}^{\ell \text{ steps}} \\
\begin{array}{cccc}
\pi_{k+1} & \pi_k & \dots & \pi_{k-\ell+2} \\
\end{array} \\
m \text{ times} \left\{ \begin{array}{cccc}
(k+1, \rightarrow, r_{k+1}) & (k+\ell, \leftarrow, 0) & \dots & (k+2, \leftarrow, 0) \\
\vdots & \vdots & \vdots & \vdots \\
(k+1, \rightarrow, r_{k+1}) & (k+\ell, \leftarrow, 0) & \dots & (k+2, \leftarrow, 0) \\
(k+1, \rightarrow, r_{k+1}) & \boxed{k+\ell} & &
\end{array} \right.
\end{array}
$$

As a consequence, with (10.c) as induction hypothesis we have:

$$
\begin{aligned}
&\left((T_{k+1,\ell})^m T_{k+1} v_k\right)(k+1) = \frac{1-\gamma^{\ell(m+1)}}{1-\gamma^\ell} r_{k+1} + \gamma^{\ell m+1} v_k(k+\ell) \\
&= r_{k+1} + \frac{\gamma^\ell - \gamma^{\ell(m+1)}}{1-\gamma^\ell} r_{k+1} + \gamma^{\ell m+1} \left(\frac{\gamma^\ell - \gamma^{\ell(m+1)}}{1-\gamma^\ell} r_k + \epsilon + \sum_{j=1}^{k-1} \gamma^{j(\ell m+1)} \left(\frac{\gamma^\ell - \gamma^{\ell(m+1)}}{1-\gamma^\ell} r_{k-j} + \epsilon \right) \right) \\
&= r_{k+1} + \frac{\gamma^\ell - \gamma^{\ell(m+1)}}{1-\gamma^\ell} r_{k+1} + \sum_{j=1}^{k} \gamma^{j(\ell m+1)} \left(\frac{\gamma^\ell - \gamma^{\ell(m+1)}}{1-\gamma^\ell} r_{k-j+1} + \epsilon \right) \\
&= r_{k+1} + v_{k+1}(k+\ell+1) - \epsilon
\end{aligned}
$$

Hence,

$$
\begin{aligned}
v_{k+1}(k+1) &= \left((T_{k+1,\ell})^m T_{k+1} v_k\right)(k+1) + \epsilon_{k+1}(k+1) \\
&= v_{k+1}(k+\ell+1) + r_{k+1} - 2\epsilon,
\end{aligned}
$$

which matches (10.c).

In states $i = k+1+q\ell+p$ For states $i = k+1+q\ell+p$ with $0 \le q \le km-1$ and $1 \le p < \ell$, the policy $\pi_{k+1,\ell}$ always takes the $\leftarrow$ action with either one of the following trajectories

- when $q \ge m$:

$$
\begin{array}{c}
\overbrace{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}^{\ell \text{ steps}} \\
\begin{array}{cccc}
\pi_{k+1} & \pi_k & \dots & \pi_{k-\ell+2} \\
\end{array} \\
m \text{ times} \left\{ \begin{array}{cccc}
(k+1+q\ell+p, \leftarrow, 0) & (k+q\ell+p, \leftarrow, 0) & \dots & (k+(q-1)\ell+p+2, \leftarrow, 0) \\
\vdots & \vdots & \vdots & \vdots \\
(k+1+(q-m+1)\ell+p, \leftarrow, 0) & (k+q\ell+p, \leftarrow, 0) & \dots & (k+(q-m)\ell+p+2, \leftarrow, 0) \\
(k+1+(q-m)\ell+p, \leftarrow, 0) & \boxed{k+(q-m)\ell+p} & &
\end{array} \right.
\end{array}
$$

As a consequence, with (10.d) as induction hypothesis we have:

$$
v_{k+1}(i) = \left((T_{k+1,\ell})^m T_{k+1} v_k\right)(i) = \gamma^{\ell m+1} v_k(k+(q-m)\ell+p) = -\gamma^{\ell m+1} \gamma^{(k-1)(\ell m+1)} \epsilon = -\gamma^{k(\ell m+1)} \epsilon
$$

which satisfies (10.d) in this case.

- when $q < m$:

Assuming that negative states correspond to state 1, where the action is irrelevant, we have the following trajectory:

$$
\begin{array}{rc}
 & \overbrace{\qquad\qquad\qquad\qquad\qquad\qquad}^{\ell \text{ steps}} \\
 & \begin{array}{ccc} \pi_{k+1} & \dots & \pi_{k-\ell+2} \end{array} \\
q \text{ times} \left\{ \right. & \begin{array}{ccc} (k+1+q\ell+p, \leftarrow, 0) & \dots & (k+(q-1)\ell+p+2, \leftarrow, 0) \\ \vdots & \vdots & \vdots \\ (k+1+\ell+p, \leftarrow, 0) & \dots & (k+p+2, \leftarrow, 0) \end{array} \\
m-q \text{ times} \left\{ \right. & \begin{array}{ccc} (k+1+p, \leftarrow, 0) & \dots & (k-\ell+p+2, \leftarrow, 0) \\ (k+1-\ell+p, \leftarrow, 0) & \dots & (k-2\ell+p+2, \leftarrow, 0) \\ \vdots & \vdots & \vdots \\ (k+1-(m-q-1)\ell+p, \leftarrow, 0) & \dots & (k-(m-q)\ell+p+2, \leftarrow, 0) \end{array} \\
 & (k+1-(m-q)\ell+p, \leftarrow, 0) \qquad \boxed{k+(q-m)\ell+p}
\end{array}
$$

In the above trajectory, one can see that only the $\leftarrow$ action is taken (ignoring state 1). Indeed, since we follow the policies $\pi_{k+1}\pi_k, \dots, \pi_{k-\ell+2}$ the $\rightarrow$ action may only be taken in states $k+1, k, \dots, k-\ell+2$. When state $k+1$ is reached, the selected action is $\pi_{k-p+1}(k+1)$ which is $\leftarrow$ since $p \geq 1$. The same reasonning applies in the next states $k, \dots, k-\ell+1$, where $p \geq 1$ prevents to use a policy that would select the $\rightarrow$ action in those states.

Since $p - \ell < 0$ the trajectory always terminates in a state $j < k$ with value $v_k(j) = -\gamma^{(k-1)(\ell m-1)}\epsilon$ as for the $q \geq m$ case, which allows to conclude that (10.d) also holds in this case.

In states $i = k+1+km\ell+p$ Observe that following m times $\pi_{k+1,\ell}$ and then π_{k+1} once amounts to always take $\leftarrow$ actions. Thus, one eventually finishes in state $k+(k-1)m\ell+p \geq k+1$, which, since $\epsilon_k(i) = 0$, gives

$$
v_{k+1}(i) = \left((T_{k+1,\ell})^m T_{k+1} v_k\right)(i) = \gamma^{\ell m+1} v_k(k+(k-1)m\ell+p) = -\gamma^{\ell m+1} 0 = 0,
$$

satisfiying (10.e).

In states $i > k+1+(km+1)\ell$ In these states, the action $\leftarrow$ is taken $\ell m+1$ times ending up in state $j > k+((k-1)m+1)\ell$, with value $v_k(j) = 0$, from which $v_{k+1}(i) = 0$ follows as required by (10.f).