Low-rank MDP Approximation via Moment Coupling

Amy B.Z. Zhang, Itai Gurvich *
Operations Research and Information Engineering, Cornell University

Abstract. We propose a novel method—based on local moment matching—to approximate the value function of a Markov Decision Process. The method is grounded in recent work Braverman et al. (2020) that relates the solution of the Bellman equation to that of a PDE where, in the spirit of the central limit theorem, the transition matrix is reduced to its local first and second moments. Solving the PDE is not required by our method. Instead we construct a “sister” Markov chain whose two local transition moments are (approximately) identical with those of the focal chain. Because they share these moments, the original chain and its “sister” are coupled through the PDE, a coupling that facilitates optimality guarantees. We show how this view can be embedded into the existing aggregation framework of ADP, providing a disciplined mechanism to tune the aggregation and disaggregation probabilities. This embedding into aggregation also reveals how the approximation’s accuracy depends on a certain local linearity of the value function. The computational gains arise from the reduction of the effective state space from $N$ to $N^{1+\epsilon}$ as one might intuitively expect from approximations grounded in the central limit theorem.

1. Introduction

Dynamic programming is the fundamental technique for solving sequential decision problems. The key object of analysis is the Bellman optimality equation. As the dimension of the state space increases, the computational burden of solving the Bellman equation becomes prohibitive. Approximate dynamic programming (ADP) is a family of algorithms developed to address this computational challenge by reducing—through various mechanisms—the dimensionality of the problem.

A central ADP theme is that of value-function approximation. One a priori imposes a lower dimensional structure on the value function—assuming, for example, that it is an affine combination of pre-specified basis functions—and optimizes the combination parameters. One expects computational gains if the number of basis functions is small relative to the size of the state space. Even when these function-approximation algorithm converge, the performance—in terms of optimality gaps—depends on the choice of the basis functions; these are often chosen based on ad-hoc knowledge of the problem’s structure.

What we propose here is a method that, instead of imposing a lower dimensional structure on the value function, approximates directly the Markov chain by a lower-rank one. The Bellman equation for this lower-rank chain is itself lower dimensional and hence more tractable.

* This work is supported by NSF grant CMMI-1662294
The lower-rank sister chain is a “non-identical twin” of the original chain. The two are coupled through their local-transition first and second moment. Specifically, these first moments are given by the vector \( \mu(x) \) and the matrix \( \sigma^2(x) \):

\[
\mu(x) = \mathbb{E}_x[X_1 - x], \quad \sigma^2(x) = \mathbb{E}_x[(X_1 - x)(X_1 - x)^\top].
\]

These are collapsed ”statistics” of the full transition matrix. Implicit in these definitions is our focus on chains where there is a natural notion of physical distance and it is most useful to fix attention to state spaces of the form \( \mathbb{Z}^d \cap \times_{i=1}^d [\ell_i, u_i] \).

The premise that coupling two chains via their moments should produce small approximation gaps is grounded in recent work \cite{Braverman2020} that connects the Taylor expansion of value functions to nearly optimal policies.\(^1\)

While the math that supports this statement is non-trivial, the intuition is rather simple. Fix a chain \((X_t, t = 1, 2, \ldots)\) on \( \mathbb{Z}^d \) with transition probability \( P \), and consider the infinite horizon \( \alpha \)-discounted reward

\[
V(x) = \mathbb{E}_x \left[ \sum_{t=0}^{\infty} \alpha^t c(X_t) \right].
\]

The value \( V \) solves the fixed point equation \( V(x) = c(x) + \alpha PV(x) \), which we re-write as

\[
0 = c(x) + \alpha (PV(x) - V(x)) - (1 - \alpha) V(x).
\]

If we pretend that \( V \) has a continuously thrice differentiable extension to the reals \( \mathbb{R}^d \), then

\[
PV(x) - V(x) = \mu(x)^\top D V(x) + \frac{1}{2} \text{trace}(\sigma^2(x)^\top D^2 V(x)) + \text{Remainder},
\]

where the remainder depends on the third derivative of the continuous extension. At least intuitively, the fixed-point equation translates to the solution of a partial differential equation.

We do not advocate using this PDE as a computational alternative but, rather, as a link (a “coupling”) between the chain \( P \) and a more tractable one. Put simply, if we construct a chain \( \tilde{P} \) on \( \mathbb{Z}^d \) with the same local moment functions \( \mu(\cdot) \) and \( \sigma^2(\cdot) \) it, too, would induce the same PDE.

To the extent that the quality of the PDE as an approximation depends only on those moments (as functions over the state space), we have a mechanism to bound the gap between the value of the two chains. Among all sister chains, we want one that is tractable in terms of value-function computation.

At this point our work plugs into, and connects naturally, to the known aggregation method in ADP. Aggregation reduces the dimensionality of the Bellman equation by solving it for a small (in

\(^1\) That work itself is closely related to the vast literature on diffusion approximations; see \cite{2}.)
relative terms) number of “meta-states”, denoted by \( L \). The extent of the reduction in computational effort depends on how \( L \) compares to the number of “detailed” states \( N \); the fewer the meta states, the less demanding the computation of the value function. In a finite-state-space setting, evaluating the performance of a given control then requires inverting an \( L \times L \) matrix instead of the \( N \times N \) matrix.

The design parameters of aggregation—the so-called aggregation and disaggregation matrices—are typically chosen in an ad-hoc manner. Moment matching offers a principled way to choose these that is grounded in approximation/optimality gap bounds. Though the construction of the sister chain, via moment matching, adds computational complexity that could compromise the gains of aggregation.

The classical moment problem in probability has a long history; see Prékopa (1990) and the references therein. Our challenges here deviate from the classical moment problem. Most fundamentally, we are facing a simultaneous problem as we are trying to match the first two moment of all of \( N = |S| \) random variables — one for each state, where the random variable for state \( x \) has the distribution \( P_x \)—via convex combination of the same (small) set of random variables. This, as will be made evident through simple examples, is generally impossible.

What we do, instead, is prioritize the first moment over the second. We use as our meta states a grid of spaced out states in the state space. These “representative states” are the effective support of the chain \( \tilde{P} \). We match perfectly the first local moment \( \mu(\cdot) \)—a feasible and simple task—while maintaining, through non-constant spacing, a handle on the second-moment mismatch. The coarseness of the reduced state space is directly informed by the mathematical analysis.

Interestingly, once the grid is (carefully) set, our mechanism for matching the first moment is equivalent to approximating the value at a state \( x \) by a distance-weighted interpolation of the values at the corner of the grid box to which \( x \) belongs. In particular, the more “locally linear” that the value is, the more accurate the approximation. Such an interpolation is rather intuitive — moment matching gives it mathematical support.

We prove that \( \tilde{V} \) that with \( L = (N^{1+\varepsilon}) \) meta states, the value of the sister chain, is a good approximation to the true value \( V \) in the following sense:

\[
|V(x) - \tilde{V}(x)| = O \left( \mathbb{E}_x \left[ \sum_{t=0}^{\infty} \alpha^t \frac{c(X_t)}{(1 + \|X_t\|)^\varepsilon} \right] \right) = o(V(x)),
\]

where \( \varepsilon \in (0, 1) \) is a design variable. The closer it is to 0, the fewer meta states (computation is easier) but the larger the gap bound. This result means that the gap is proportional to the infinite-horizon discounted value with a scaled down cost function \( c(x)/(1 + \|x\|)^{\varepsilon} \); with \( \varepsilon = 0 \) computation is easier but the gap is of the order of the value itself.
These guarantees are not fully general; to use PDE theory, we require that the first and second transition moments satisfy some smoothness properties. These make mathematically precise the intuitive connection to the central limit theorem. Most useful, however, is the way in which the mathematical analysis informs the design of the aggregation scheme.

When we embed moment matching in an approximate policy iteration algorithm the computational gains are further magnified. Savings are realized in both the policy evaluation and update steps. Importantly, our moment-based design of the aggregation and disaggregation matrices is policy independent. In turn, they are computed once and do not have to be updated on each iteration.

Setting up the algorithmic framework for moment-matching based MDP is the first contribution of our paper. The second is to provide approximation guarantees. These (and the uncovering of their dependence on the coarseness) inform the algorithm design. We illustrate the computational value through several numerical examples.

**Notation.** Unless stated otherwise, $\| \cdot \|$ corresponds to the Euclidean norm on $\mathbb{R}^d$ ($d$ will be clear from the context). We write $y = x \pm \epsilon$ to denote $\|y - x\| \leq \epsilon$. We use $\mathbb{R}_+^d$ and $\mathbb{Z}_+^d$ to denote the non-negative reals in $\mathbb{R}^d$ and integers in $\mathbb{Z}^d$, and use $\mathbb{R}_{++}^d$ and $\mathbb{Z}_{++}^d$ when they are strictly positive. For a function $f : A \to \mathbb{R}^d$ and a set $B \subseteq A$, $|f|_B = \sup_{x \in B} \|f(x)\|$. We use $\Gamma$ to denote a universal constant whose value might change from one line to the next but that does not depend on the state $x$ or the discount factor $\alpha$. Where useful we will point out its dependencies. We write $f(x) \lesssim q(x)$ to mean $f(x) \leq \Gamma q(x)$ and $f(x) \cong q(x)$ if both $f(x) \lesssim q(x)$ and $q(x) \lesssim f(x)$.

**A comment on organization.** We focus for much of the manuscript on the value approximation for a given policy, namely on the study of a so-called Markov reward process. This is then made a (central) module in an approximate policy iteration algorithm in §8 All lemmas that are stated in the main body of the paper are proved in the appendix.

## 2. Literature

ADP is concerned with approximating solutions to complex control problems where the size of the state space prohibits exact computation of the value function and/or the optimal control policy. The literature on ADP is vast. Key gains in computation are achieved by restricting the search for value functions to an architecture—a pre-specified family of functions. In linear architectures, for example, value functions are restricted to linear combinations of pre-specified features. More recent methods use neural networks as the underlying architecture; see e.g. Bertsekas (2018) and Gijsbrechts et al. (2019); Vanzuchelen et al. (2020) for recent applications to operations management problems.
Two questions must be posed to any architecture-based ADP algorithm: (1) Does the algorithm converge to the best choice of parameters within the given architecture. In the case of a linear architecture, for example, does the algorithm produce the best feature coefficients; (2) Such convergence may not mean much if the architecture is inadequate for the problem at hand, so we must also ask how well the “best” choice within the given architecture approximates the original problem of interest.

The first question was answered affirmatively for linear architectures; see Tsitsiklis and Van Roy (1996, 1997). This was followed by improvements to convergence rates; see e.g. Devraj and Meyn (2017). There are, however, few approximation algorithms with theoretical guarantees on the optimality gaps— that is, on how well the prescribed (approximate) control performs in the original system. Furthermore, while approximate dynamic programming has known significant practical success, the choice of the architecture often builds on ad-hoc intuition about the problem at hand, rather than on a principled approach to its construction.

Our focus is not on convergence rates for a given approximation architecture but, rather, on a new architecture with optimality-gap guarantees. Our architecture does not rely on a value function approximation. Instead it approximates the controlled Markov chain by matching its local moments.

The approach we put forth piggy backs on state aggregation methods to produce an algorithm that relates the approximation error of a Markov Chain Moment Problem. Specifically, given the original (controlled) chain we build a new chain that matches in local transition moments, through the choice of aggregation mechanism. In other words, what we propose is a principled approach to tune the aggregation design variables.

State aggregation has a long history; see Bean et al. (1987); Tsitsiklis and Van Roy (1996); Whitt (1978) to name a few. We primarily follow the exposition in Bertsekas (2017). Much of the literature focuses on hard aggregation, either fixing cluster memberships a priori or updating them based on value estimates, e.g. Baras and Borkar (2000); Bertsekas and Castanon (1989). Soft aggregation has also been identified as a useful approximation infrastructure for reinforcement learning in Singh et al. (1995) for its flexibility.

Our construction of the “low rank” sister chain is based on a relatively simple matching of the first moment. If instead the transition matrix \( P \) is itself low rank, matrix factorization techniques can be used to identify the aggregation parameters, see Duan et al. (2019); Ghasemi et al. (2020).

Our algorithm offers a principled method for selecting the aggregation parameters backed by performance guarantees. We construct mapping from detailed states to meta-states based on local moments of the controlled chain, without requiring structural assumptions or knowledge of value function estimates.
Moment-based approximations—inspired by the central limit theorem and functional version thereof—have been extremely successful in queueing theory facilitating the analysis and optimization of highly complex queueing networks. Some of the “import” of the mathematical theory from the control of queues to general dynamic programs has been achieved in [Braverman et al. (2020)] where the connections to queueing theory are thoroughly discussed.

We use the mathematical constructs in [Braverman et al. (2020)] as a starting point for an algorithmic framework. What we adopt is the view that matching local moments—a collapsed “statistic” of the full transition matrix—has the potential to produce small optimality gaps. How to do so algorithmically — how to construct the sister chain $\tilde{P}$ for computational gains — is the question we address in the current paper. In the process of developing our algorithm, we expand on [Braverman et al. (2020)] to allow for some mismatch in the second moment between the focal chain and its sister in our bounds.

Finally, our work is indirectly related to sensitivity analysis for MDP (and POMDP); see e.g. [Mastin and Jaillet (2012), Ross et al. (2009)] which study, among other things, sensitivity to changes in the transition distribution. We bound the value-differences between two chains in terms of their local transition moments, a “collapsed” statistic of the transition matrix.

3. The model

We consider the infinite-horizon discounted reward for a discrete-time Markov chain on a finite state space $S \subseteq \mathbb{Z}^d \cap \times_{i=1}^d [\ell_i, u_i]$. Let $N = |S|$ be the size of the state space. $P$ is the transition matrix with $p_{xy}$ equal to the probability of transitioning from $x$ to $y$ in one step; $c : S \rightarrow \mathbb{R}^+$ is the cost function. We assume that the function $c$ is norm like; that there is a $k \in \mathbb{Z}_+$ and a point $x_0 \in S$ such that

$$\frac{1}{\Gamma} \|x - x_0\|^k \leq |c(x)| \leq \Gamma(1 + \|x - x_0\|)^k.$$  

Since one can shift the state space, we will assume w.l.o.g. that $x_0 = 0$.

Finally, $\alpha \in (0, 1)$ is the discount factor. This so-called “Markov reward” process is characterized by the tuple $C = <S, P, c, \alpha>$. The value function is then given by

$$V(x) = \mathbb{E}_x \left[ \sum_{t=0}^{\infty} \alpha^t c(X_t) \right], \quad x \in S,$$

where $\mathbb{E}_x[\cdot]$ is the expectation with respect to the law $P_x$.

For a function $f : S \rightarrow \mathbb{R}$ we use the operator notation $Pf(x) := (Pf)(x) = \sum_y p_{xy}f(y) = \mathbb{E}_x[f(X_1)]$. As is standard, the function $V : S \rightarrow \mathbb{R}$ is the unique solution to the equation $TV = V$, where

$$TV(x) = c(x) + \alpha PV(x).$$
We will refer to this as the Bellman equation despite the absence of a control decision here. This allows for continuity of language with optimization in §8. Since the state space is finite, $V$ can be computed via the matrix inversion formula $V = (I - \alpha P)^{-1}c$.

In our analysis we will sometimes refer to the maximal jump size of $P$ from $x$

$$\Delta_x := \sup_{y : p_{xy} > 0} \|y - x\|.$$  (maximal jump)

4. Tayloring reconsidered

Consider two Markov Reward Processes. The first, $C = \langle S, P, c, \alpha \rangle$, is driven by the focal chain $P$. The other, $\tilde{C} = \langle S, \tilde{P}, c, \alpha \rangle$, is driven by the sister chain $\tilde{P}$; $\tilde{C}$ differs from $C$ only in terms of the transition probability matrix.

A “replacement” of a chain with a proxy is useful only insofar as it yields computational benefits by, say, being of lower rank. It seems ambitious to require $P$ and a lower rank $\tilde{P}$ to be close in some reasonable matrix norm unless $P$ is itself low rank. Instead, it makes sense to measure the distance between transition matrices in terms of their impact on the value function.

**Definition 4.1** Given a function $f : S \to \mathbb{R}$, and two transition probability matrices $P, \tilde{P}$ on $S$, let

$$\delta_f[P, \tilde{P}] := |\delta_f[P, \tilde{P}](\cdot)|^*_S,$$

where

$$\delta_f[P, \tilde{P}](x) := |\mathbb{E}_x[f(X_1)] - \mathbb{E}_x[f(X_1)]| = |\tilde{P}f(x) - Pf(x)|.$$

**Lemma 4.1**

$$|V - \tilde{V}|^*_S \leq \frac{\alpha}{1 - \alpha} \left( \delta_V[P, \tilde{P}] + \delta_{\tilde{V}}[P, \tilde{P}] \right).$$

The bound in Lemma 4.1 seems problematic, as it requires information about $V$, the very construct whose computation we seek to avoid. It is valuable, however, in that it identifies $|\mathbb{E}_x[V(X_1)] - \mathbb{E}_x[V(X_1)]|$ as a central object of study. It makes clear that, in comparing two chains, what matters is the local behavior: how the one step change in value under $P$ (i.e. $\mathbb{E}_x[V(X_1)] - V(x)$) compares to that under $\tilde{P}$ (i.e. $\mathbb{E}_x[V(X_1)] - V(x)$).

This localization makes Taylor-expansion (initially, heuristically) a natural lens through which to study approximation gaps. We make the following observation. If $V$ has a thrice continuously differentiable extension to $\mathbb{R}^d$, then

$$\mathbb{E}_x[V(X_1)] = V(x) + \mu(x)'DV(x) + \frac{1}{2}{\text{trace}}(\sigma^2(x)'D^2V(x)) \pm \frac{1}{6} ||D^3V||\Delta^3_x,$$
where, recall, \( \Delta_x := \sup_{y : P_{xy} > 0} \| y - x \| \) is the maximal jump of the chain from state \( x \), and

\[
\mu(x) = \mathbb{E}_x[X_1 - x], \quad \sigma^2(x) = \mathbb{E}_x[(X_1 - x)(X_1 - x)^\top].
\]

The expectation \( \mathbb{E}_x[V(X_1)] \) for the sister chain can be expanded analogously. If \( \tilde{P} \) shares the first two local moments with \( P \), i.e.

\[
\tilde{\mu}(x) := \mathbb{E}_x[X_1 - x] \approx \mathbb{E}[X_1 - x] \quad \text{and} \quad \tilde{\sigma}^2(x) := \mathbb{E}[(X_1 - x)(X_1 - x)^\top] \approx \mathbb{E}[(X_1 - x)(X_1 - x)^\top],
\]

then

\[
\mathbb{E}_x[V(X_1)] \approx V(x) + \tilde{\mu}(x)'DV(x) + \frac{1}{2} \text{trace}(\tilde{\sigma}^2(x)'D^2V(x)) 
\]

\[
\approx V(x) + \mu(x)'DV(x) + \frac{1}{2} \text{trace}(\sigma^2(x)'D^2V(x)) \approx \mathbb{E}_x[V(X_1)],
\]

in turn,

\[
\delta_V[P, \tilde{P}](x) = | \mathbb{E}_x[V(X_1)] - \mathbb{E}_x[V(X_1)] | \approx 0.
\]

Here \( \approx 0 \) should be interpreted as “\( \delta_V \) being small relative to the value function \( V \)”; the precise mathematical meaning of \( \approx 0 \) is exposed further below. This informal derivation makes clear that (1) if a low-rank sister chain has the same moments as the focal chain, its value may provide a good approximation to that of the focal chain. To be low rank, this chain might have larger jumps, so that (2) in designing this sister chain we must keep its jumps small, at least in regions of the state space where the third derivative is substantial.

**Example 1 (The simple random walk)** Consider the simple absorbing random walk on the integers: \( P_{x, x+1} = P_{x, x-1} = 1/2 \) for all \( x = 1, \ldots, n - 1 \) and \( P_{00} = P_{nn} = 1 \). It is easy to see that \( \mathbb{E}_x[X_t] = x \) for all \( t \geq 0 \) so that \( \mathbb{E}_x[\sum_{t=0}^\infty \alpha^t X_t] = \frac{x}{1-\alpha} \). The same conclusion holds for the “simpler” chain that jumps in one step to one of the end points: \( \tilde{P}_{xn} = 1 - \tilde{P}_{x0} = x/n \). Observe that \( \mu(x) = \tilde{\mu}(x) = 0 \) for all \( x \). Thus, \( P \) shares the local first moments as well as value function with a sister chain \( \tilde{P} \) that has only two states.

Example 1 is rather unique. One should not expect a perfect value-function match in general, certainly not with such a coarse state-space. Our bounds in §10 will capture the dependence of the approximation’s accuracy on the “density” of the meta-states.

The informal derivation through Taylor expansion is useful for developing intuition but does not provide a basis for algorithm design. The value \( V \) is not apriori known so it is impossible to “refer” to its continuous extension. To circumvent this, Braverman et al. (2020) develops a framework for obtaining indirectly an approximate continuous solution. A short summary of this earlier work is
useful. Consider a chain on $\mathbb{Z}^d$. The value $V$ solves the Bellman equation $V(x) = c(x) + \alpha PV(x)$ which we find useful to re-write as

$$0 = c(x) + \alpha (PV(x) - V(x)) - (1 - \alpha) V(x).$$

Pretending that the function $V$ is twice continuously differentiable, 2nd-order Taylor expansion yields the partial differential equation (PDE)

$$0 = c(x) + \alpha \left[ \mu(x)'DV(x) + \frac{1}{2} \text{trace}(\sigma^2(x)'D^2V(x)) \right] - (1 - \alpha) V(x),$$

defined now over $\mathbb{R}^d$. While this equation has been arrived-to purely formally, the following is a valid mathematical question: what is the relationship between a solution $\hat{V}$ (if it exists) to this equation on $\mathbb{R}^d$, and $V$ that solves the original discrete-state-space Bellman equation.

Two chains $<S,P,c,\alpha>$ and $<S,\tilde{P},c,\alpha>$ with the same local moment functions $\mu(\cdot)$ and $\sigma^2(\cdot)$ induce the same reduction to a continuous-state space PDE so that bounds $|V - \hat{V}|$ and $|\hat{V} - \tilde{V}|$ produce, as a corollary, a bound on $|V - \tilde{V}|$. This is the path we take.

5. Sister-chain construction via aggregation

Aggregation effectively creates a new Markov chain on a smaller state space. The tuning of the aggregation parameters is tantamount to selecting for this chain a transition matrix from a restricted family of such. The flexibility this offers makes it an ideal vehicle for our moment-matching algorithm.

5.1. Aggregation Preliminaries

Recall that $N = |S|$ denotes the number of states in the original MDP, and let $\mathcal{M} = \{1, \ldots, L\}$ be a family set of meta states; obviously $L \leq N$. We refer to [Bertsekas, 2012, Chapter 6] for a full discussion and include below the minimal ingredients for a self-contained exposition. Two weight matrices govern the mapping between $S$ and $\mathcal{M}$;

- **Aggregation probabilities**: For each detailed state $x \in S$, the probability that $x$ aggregates (or "groups") into $k$, $g_{xk} \geq 0$, represents the degree of membership of detailed state $x$ in meta-state $k \in \mathcal{M}$. The $N \times L$ matrix $G = \{g_{xk}\}$ is non-negative and row-stochastic. **Hard aggregation** is the special case where the meta-states form a partition of the state-space, and each state $x \in S$ "belongs" to a single partition: $g_{xk} = 1$ for one and only one $k \in \mathcal{M}$. The more general case is referred to as **soft aggregation**.

- **Disaggregation probabilities**: For each meta-state $l \in \mathcal{M}$, the probability that $l$ disaggregates (or "un-groups") into $x$, $u_{lx} \geq 0$, is the degree to which meta-state $l$ is represented by detailed state $x \in S$. The $L \times N$ matrix $U = \{u_{lx}\}$ is also non-negative and row-stochastic. If some meta-state $l$ is represented by a single state $x_l$, i.e. $u_{lx_l} = 1$, we refer to this $x_l$ as the **representative state** of meta-state $l$. 
Having fixed the matrix $G$ and $U$, one solves an aggregated Bellman equation on the meta-states:

$$ R(l) = \sum_{x \in S} u_{lx}(c(x)) + \alpha \sum_{y \in S} p_{xy} \sum_{k \in \mathcal{M}} g_{yk} R(k), \; l \in \mathcal{M}, \quad (2) $$

whose matrix form $R = UC + \alpha UPG$ reduces to

$$ R = (I - \alpha UPG)^{-1} UC. \quad (3) $$

The function $R$ is the value function of the aggregate problem.

In the case of “hard aggregation” the true value function is assumed to be constant over each subset in the partition. We say that $x \in S_k$ (or in “cluster” $k$) if $g_{xk} = 1$, and approximate its value with the aggregate value $R(k)$. The following is known.

**Proposition 5.1 (hard aggregation bound, Proposition 4.2 [Bertsekas (2018)])** The unique $R$ satisfies

$$ |R(k) - V(x)| \leq \frac{|\epsilon(k)|_{\mathcal{M}}}{1 - \alpha}, \; k \in \mathcal{M}, \; x \in S_k, $$

where

$$ \epsilon(k) = \max_{x,y \in S_k} |V(x) - V(y)|. \quad (4) $$

This bound makes explicit that we want to “group” together states that are similar in their value. Since one does not want (or cannot) compute the exact value one must have insight into the problem to identify this grouping. The bound we obtain here for soft aggregation with our choice of $U,G$ has a similar flavor; see Theorem 2 further below.

In the case where each meta-state $l$ has a representative state $x_l$ (i.e. $U$ is binary), it is convenient to think of $\mathcal{M}$ as the set of representative states $S^0 := \{ x \in S : x_l = x \text{ for some } l \in \mathcal{M} \}$. A soft aggregation (non-binary $G$) then interpolates detailed states from the representative ones. This coarse grid scheme is what we use in our algorithm.

### 5.2. A low rank chain on $S$

It is clear that aggregation produces dynamics on the space $\mathcal{M}$ of meta-states with transition law $UPG$. However our aim, recall, is to build a sister chain on $S$. Theorem 1 provides a simple but powerful starting point, relating aggregation to a sister chain with transition matrix $\tilde{P}$ that is composed of $P$ and $GU$:
Theorem 1 (aggregation as sister chain) Consider the value $\tilde{V}$ of a Markov chain on the original detailed state space $\mathcal{S}$ with the transition matrix

$$\tilde{P} = P GU$$

(here $\tilde{P}_{xy} = \sum_{z \in \mathcal{S}, l \in \mathcal{M}} p_{xz} g_{zl} u_{ly}$).

The aggregate value $R$ in (2) equals $U \tilde{V}$, and $\tilde{V} = c + \alpha PG \tilde{R}$.

Proof: From the Bellman equation for this chain, we have that the value $\tilde{V}$ satisfies

$$\tilde{V}(x) = c(x) + \alpha \tilde{P}\tilde{V}(x) = c(x) + \alpha PGU \tilde{V}(x).$$

Define $\tilde{R} := U \tilde{V}$. By the above, we also have $\tilde{V} = c + \alpha PG \tilde{R}$. Moreover, multiply both sides by $U$ gives us

$$\tilde{R} = Uc + \alpha UPG \tilde{R}.$$

This $\tilde{R}$ is in fact the unique solution to the aggregate Bellman equation.

In this way, aggregation gives rise to a family of lower rank chains with law $\tilde{P}[G,U] := PGU$ on the detailed state space, which we call the $(G,U)$-lifted chain. The lifted chain’s value is $c + \alpha PG \tilde{R}$ and, as such, is obtained from the lower dimensional $R$, reducing the computational complexity. In the control context, this will allow us to avoid full policy optimization on $\mathcal{S}$ during policy iteration; see §8.

The sister chain will thus be a lifted chain whose parameters are tuned for moment matching. In an architecture with representative states, $R = U \tilde{V}$ simplifies to $R(l) = \tilde{V}(x_l)$. In that case the degrees of freedom are in (a) the choice of the representative states and (b) the design of the aggregation matrix $G$.

6. Producing the sister chain

The moments of the transition law $\tilde{P}[G,U] = PGU$ are given by

$$\tilde{\mu}[G,U](x) = \sum_y \tilde{P}_{xy}[G,U](y-x), \quad \tilde{\sigma}^2[G,U](x) = \sum_y \tilde{P}_{xy}[G,U](y-x)(y-x)^\top.$$

In our framework, aggregation is likely to work well if $\tilde{\mu}[G,U] \approx \mu$ and $\tilde{\sigma}^2[G,U] \approx \sigma^2$. To make this formal, it is useful to introduce the functions $W_1 : \mathbb{R}^d \to \mathbb{R}^d$ and $W_2 : \mathbb{R}^d \to \mathbb{R}^{d^2}$, given by $W_1(x) = x$ (the identity operator) and $W_2(x) = xx^\top$, so that

$$PW_1(x) = \mathbb{E}_x[X_1], \text{ and } PW_2(x) = \mathbb{E}_x[X_1X_1^\top].$$
Since \( \mu(x) = PW_1(x) - x \), and \( \sigma^2(x) = PW_2(x) + x\mu(x)^\top + \mu(x)x^\top - xx^\top \),

\[
\begin{align*}
\mu(x) - \tilde{\mu}[G,U](x) &= PW_1(x) - \tilde{PW}_1(x) \\
\sigma^2(x) - \tilde{\sigma}^2[G,U](x) &= PW_2(x) - \tilde{PW}_2(x) + x[\mu(x) - \tilde{\mu}(x)]^\top + [\mu(x) - \tilde{\mu}(x)]x^\top.
\end{align*}
\]

Furthermore, if \( G, U \) are chosen such that \( \tilde{\mu}[G,U](x) = \mu(x) \), equation (6) becomes

\[
\sigma^2(x) - \tilde{\sigma}^2[G,U](x) = PW_2(x) - \tilde{PW}_2(x).
\]

We observe that to be able to match both moments for a state \( x \), utilizing as support a subset \( \mathcal{Y} \subseteq \mathcal{S} \), necessitates the existence of a solution \( \alpha_{xy} \), \( x \in \mathcal{S}, y \in \mathcal{Y} \), to the family of equations (simultaneous in \( x \))

\[
\sum_{y \in \mathcal{S}} \alpha_{xy} W_1(y) = PW_1(x), \quad \sum_{y \in \mathcal{S}} \alpha_{xy} W_2(y) = PW_2(x), \quad x \in \mathcal{S}.
\]

Or in other words that any point in the \( n + n^2 \) dimensional scatter \( \{(PW_1(x), PW_2(x)), x \in \mathcal{S}\} \) can be written as a convex combination of states \( y \in \mathcal{Y} \). Note that this is necessary but not sufficient for aggregation-based moment matching. We would further need the convex combination \( \alpha \) be decomposable as \( PGU \) where \( G, U \) are valid aggregation and disaggregation matrices.

**Remark 1 (The (im)possibility of 2nd moment matching)** The existence of a small strict subset \( \mathcal{Y} \) of \( \mathcal{S} \) (and coefficients \( \alpha_{xy} \)) with the above property is not guaranteed. A simple example makes this abundantly clear and also captures the subtlety of (simultaneous) moment matching.

![Figure 1](image)

*Figure 1* The moment scatter plot for two different random walks on \([0,1,\ldots,20]\). Circles: Simple random walk with absorbing end points, where simultaneous matching of both moment is impossible — one cannot express a point as a convex combination of other points. Squares: a random walk where each point in the moment scatter can be written as a convex combination of the two end points.
Consider the simple absorbing random walk on \( [0, 1, \ldots, n] \), with \( P_{x,x+1} = P_{x,x-1} = \frac{1}{2} \) for all \( x = \{1, \ldots, N-1\} \) and \( \{0, n\} \) are absorbing states. Here we have \( E_x[X_1] = x \) for all \( x \) and \( E_x[X_i^2] = x^2 + 1 \{x \notin \{0, n\}\} \). Given the scatter \( \{(E_x[X_1], E_x[X_i^2]), x \in S\} \), one cannot express all points as a convex combination of (a common) small number of points; see the round markers in Figure 1. A piecewise linear approximation allows for matching the first moment while controlling, through the number of breakpoints, the quality of second moment match.

For contrast consider a chain that has \( P_{x0} = 1 - x/n \) and \( P_{xn} = x/n \) (absorbing in one step at the boundary). Here \( E_x[X_1] = x \) for all \( x \) and \( E_x[X_i^2] = nx \) and both moments can be matched using only two representative states corresponding to the end/corner points of the state-space; see the square markers in Figure 1.

Because of this general impossibility it seems natural to prioritize the first order match. The PDE view is informative here: errors in matching \( \mu \) should translate into approximation error that are proportional to the first derivative of \( \hat{V} \), whereas errors in the matching of \( \sigma^2 \) would only be multiplied by the second derivative. We will insist then on matching the first moment exactly while controlling the second moment mismatch.

Thus we use a coarse grid scheme where each meta state \( l \) maps to a representative state \( x_l \) (i.e. \( \mathcal{M} \subseteq S \)), and each row of \( G \) has weights over \( \mathcal{M} \). Then \( UW_1(l) = x_l, UW_2(l) = x_lx_l^\top \) and we are looking for \( G \) such that for all \( x \in S \)

\[
\sum_l [P_x G][x_l] = E_x[X_1], \quad \text{and} \quad \sum_l [P_x G][x_l x_l^\top] \approx E_x[X_1 X_1^\top].
\]

We show that for a state \( x \), \( G_x \) is straightforward to compute explicitly— it assigns weights to representative states proportionally to their distance from \( x \). Our choice of the coarse grid (hence the representative states) provides us control over the second moment mismatch. No optimization problem needs to be solved.

### 6.1. A coarse grid of representative states

Define spacing function \( q(z) = z^s \) for a spacing exponent \( s \in (0, 1) \). The choice of \( s \) is a trade-off between accuracy and computation: a smaller value of \( s \) produces a finer grid, which implies greater accuracy but a heavier computational burden. The analysis in \( [10] \) shows that \( s < \frac{1}{2} \) is required for accurate approximations while the complexity analysis reveals that \( s \geq \frac{1}{3} \) guarantees computational gains even under conservative estimates.\(^2\)

\(^2\) In the mathematical guarantees we use the spacing function \( q^\alpha(z) = (1 - \alpha)^{\frac{1}{2}} q(z) \). For \( \alpha = 0.99 \) for example, \( (1 - \alpha)^{\frac{1}{2}} \geq 0.3 \). We simplify the algorithm exposition by dropping this multiplicative constant.
The formal construction of the grid is tedious but straightforward. Recall that \( \mathcal{S} = \mathbb{Z}^d \cap \times_{i=1}^d [l_i, u_i] \). The grid is constructed symmetrically about the origin, so consider the positive portion of an axis \( i \). Index the grid with \( \{k\} \) and let \( f(k) \) be the axis value at index \( k \), which is given recursively by

\[
f(k + 1) = \lceil f(k) + q(f(k)) \rceil + 1.
\]

Take \( f(0) = \max \{0, \ell_i\} \) and set \( f(\bar{n}_i) = u_i \) for \( \bar{n}_i := \min \{k : f(k) \geq u_i\} \).

Each point on the grid is thus characterized by an index set \( \vec{k} = [k_1, ..., k_d] \), where \( k_i \in [-\bar{n}_i, \bar{n}_i] \), \( i = 1, ..., d \). These grid-points are the representative states \( \mathcal{S}_0 = \{x(\vec{k}) : x(\vec{k})_i = f(k_i)\} \). We construct the matrix \( U \) so that for every \( \vec{k} \in \mathcal{M} \)

\[
u_{\vec{x}(\vec{k})} = 1, \quad u_{\vec{y}y} = 0 \text{ for all } y \neq x(\vec{k})\] 

(Disaggregation matrix)

For notational simplicity, when the explicit value of \( \vec{k} \) is immaterial we will revert to using \( l \) and \( x_l \) for a meta state and its representative state.

Figure 2 (LEFT) illustrates the general pattern over \( \mathbb{Z}^2 \), with the red lines highlighting how the spacing along each axis scales with the distance to the origin on that axis.

![Figure 2](image)

**Figure 2** (LEFT) The grid with spacing exponent \( s = 0.5 \) and an encasing box \( \mathcal{B} \). (RIGHT) Aggregation matrix \( G \) is used to express each state \( y \) as a convex combination of the meta-states on its encasing box.

**Lemma 6.1 (number of meta-states)** With spacing exponent \( s \), the number \( L = |\mathcal{S}_0| = |\mathcal{M}| \) of representative (and hence meta-) states satisfies

\[
L \leq \left( \frac{\sqrt{2}}{1-s} \right)^d |\mathcal{S}|^{1-s}.
\]
The spacing exponent \( s \) is related to \( \varepsilon \) in our accuracy bound (see (1) and (10) below) by \( \varepsilon = 1 - 2s \) \((s < 1/2)\); in our numerical experiments we use \( s = 0.45 \) which already results in high accuracy. By Lemma 6.1 the number of meta states is bounded by \((2\sqrt{2})^dN^{1+s} \) for \( s \in [0, 1/2] \). In the special case where \( S = [0, r]^d \),

\[
\frac{L}{N} = \frac{|\mathcal{M}|}{|\mathcal{S}|} \leq \left( \frac{2\sqrt{2}}{r^s} \right)^d,
\]

implying that the bigger \( r - (2\sqrt{2})^{1/2} \) is, the more substantial the dimensionality reduction.

### 6.2. Aggregation matrix \( G \)

With the construction of representative states in the previous section, it is always feasible to find a \( N \times L \) stochastic matrix \( G \) to achieve perfect first moment matching

\[
y + \bar{\mu}(y) = \sum_i [P_y G_i]_i x_i = \mathbb{E}_y[X]_1 = y + \mu(y), \quad \forall y \in \mathcal{S}.
\]

There may be multiple matrices \( G \) that satisfy this moment matching. We construct ours as follows:

For each state \( y \in \mathcal{S} \) let \( g_{yl} \cdot \) be a distribution over \( \mathcal{M} \) such that

\[
\sum_l g_{yl} x_l = y.
\]

The matrix \( G \) with rows \( \{g_{yl} \cdot, y \in \mathcal{S}\} \) immediately satisfies first order moment matching because

\[
\mathbb{E}_x[X]_1 = \sum_y p_{xy} x_l = \sum_y p_{xy} y = \mathbb{E}_x[X]_1, \text{ for all } x \in \mathcal{S} \]

### Computing \( g_{yl} \cdot \)

For each \( y \in \mathcal{S} \) we identify the smallest enclosing box and write \( y \) as a convex combination of its corners, as illustrated in Figure 2 (RIGHT). For a given \( y \), let \( \mathcal{B} = \{\vec{k}_1, \ldots, \vec{k}_{2^d}\} \in \mathcal{B} \) be the set of \( 2^d \) meta (i.e. representative) states that form the box, restrict \( g_{yl'} = 0 \) for \( l' \notin \mathcal{B} \), and solve for

\[
\sum_{l \in \mathcal{B}} g_{yl} x_l = y, \text{ where } \sum_{l \in \mathcal{B}} g_{yl} = 1 \text{ and } g_{yl} \geq 0 \text{ for } l \in \mathcal{B}.
\]

This set of linear constraints has an explicit solution. Given a box \( \mathcal{B} \), let \( \bar{s}_i = \max_{x \in \mathcal{B}} x_i \) and \( \underline{s}_i = \min_{x \in \mathcal{B}} x_i \) for \( i \in [d] \). Then, give \( y \) and its enclosing box \( \mathcal{B} \), we write

\[
g_{yl} = \prod_{i=1}^d \left[ \mathbf{1}\{ (x_i) = \bar{s}_i \} \ast \frac{y_i - \bar{s}_i}{\bar{s}_i - \underline{s}_i} + \mathbf{1}\{ (x_i) = \underline{s}_i \} \ast \frac{\bar{s}_i - y_i}{\bar{s}_i - \underline{s}_i} \right]. \tag{9}
\]

Intuitively, \( g_{yl} \) weighs nearby representative states \( x_i \in \mathcal{B} \) proportional to their relative distance to state \( y \). The following summarizes the properties of \( G \).

\[\text{This is an instance of a more general fact. For perfect first moment matching it suffices that } G, U \text{ are such that, for each } y, (GU)_y \text{ is the distribution of } y + Z \text{ where } \mathbb{E}[Z] = 0. \text{ In that case, } (PGU)_x \text{ is the convolution of } P_x \text{ and a zero-mean jump and, consequently, has the same mean as } P_x : GUW_1(y) = y \implies PGDW_1(x) = x + \mu(x).\]
Lemma 6.2  The construction of $G$ in (9) satisfies $\sum_{l \in B} g_{yl} = 1$ and $\sum_{l \in B} g_{yl} x_l = y$. Also, $g_{yl} = 1$ when $y = x_l$ for $l \in M$. The choice of $S^0$ (hence $U$) and $G$ guarantees that $\bar{P} = PGU$ induces perfect first moment matching.

To bound the second moment mismatch, let $\Sigma(x) = \mathbb{E}_x[(X_1 - \mathbb{E}_x[X_1])(X_1 - \mathbb{E}_x[X_1])^\top]$ be the covariance matrix of $X_1$ starting at $x$. If $\Gamma$ is such that $|\Delta|^s \leq \Gamma$, then $\|\tilde{\Sigma}(x)\| \leq \Gamma$ for a re-defined constant.

Lemma 6.3 (second moment mismatch) Consider a Markov chain on $\mathcal{S} = [\ell_i, u_i]^d \cap \mathbb{Z}^d$. With $G, U$ produced by Algorithm 1, we have a constant $\Gamma$ such that

$$\|PGDW_2(x) - PW_2(x)\| \leq \Gamma(||x|| + \Delta x)^s.$$  

The mathematical bounds in §10 inform the choice of the coarseness parameter $s$ for the sister chain $\bar{P}$. Put simply, they reveal that we “can afford” state-dependent spacing $||x||^s$ for $s < 1/2$ between meta states and capture how the accuracy gap shrinks as $s$ decreases further from 1/2.

7. Algorithm and Complexity

The moment-matching (MoMa) aggregation algorithm in 1 summarizes our construction—via representative states and distance-proportional aggregation—of the design matrices $U, G$.

Algorithm 1  Moment-Matching (MoMa) aggregation with coarse grid

**Input:** State space $\mathcal{S} = [\ell_i, u_i]^d \cap \mathbb{Z}^d$, spacing exponent $s$.

**Output:** Aggregation structure with parameters $U, G$.

1. **Construct $s$-spaced grid:** Create grid-points $z^k_i, z^{-k}_i$ as appropriate for each axis $i$.
2. **Construct $U$:** For each meta-state $\tilde{k} = [k_1, ..., k_d]$ on the grid, assign $u_{\tilde{k}x_{\tilde{k}}} = 1$ where $[x_{\tilde{k}}]_i = z^k_i$.
3. **Construct $G$:** For each state $y$ and meta-states in its enclosing box $l \in \mathcal{B}$, compute distribution $g_{yl}$ such that $\sum_{l \in B} g_{yl} x_l = y$ and $g_{yl'} = 0$ for $l' \notin \mathcal{B}$.

Algorithm 2  Policy evaluation with MoMa aggregation

**Input:** Markov reward process $\mathcal{C} = <\mathcal{S}, P, c, \alpha>$, spacing exponent $s \in [\frac{1}{3}, \frac{1}{2})$.

**Output:** Approximate value $\tilde{V}$.

1. **MoMa aggregation:** Obtain $U, G$ using Algorithm 1
2. Solve $R = (I - \alpha UP G)^{-1} U c$.
3. Compute approximation $\tilde{V} = c + \alpha PGR$.  

Amy B.Z. Zhang, Itai Gurvich: ADP via Moment Coupling
The following theorem, a corollary of Lemma 4.1, shows that the quality of the approximation depends on the local linearity of the value function $V$ and its approximation $\tilde{V}$. We abbreviate here the notation

$$G\tilde{V}(y) = \sum_l g_{yl} \tilde{V}(x_l)$$

**Theorem 2** With the coarse grid scheme,

$$|V - \tilde{V}|_S \leq \frac{1}{1-\alpha} \left( |V - GV|_S + |\tilde{V} - G\tilde{V}|_S \right).$$

The gap depends, then, on how well the convex combination of the values $V(x_l)$ at neighboring representative states $x_l \in \mathcal{B}$ approximates the value $V(y)$ for $y$ in box $\mathcal{B}$; similarly for $\tilde{V}$. Our construction effectively interpolates the value at a point $y$ from those at the nearest grid points with weights corresponding to the relative distance from those grid points. This distance-based interpolation is the one that arises from perfect first moment matching. In particular, $\sum_l g_{yl}(x_l - y) = 0$, so that, pretending a smooth extension of $V$,

$$V(y) - \sum_l g_{yl} V(x_l) \approx -\sum_l g_{yl} D V(y)'(x_l - y) + O(\Delta y^2 D^2 V(y)) = O(\Delta y^2 D^2 V(y)).$$

The guarantees in §10 formalize this.

**Computational complexity.** The computational complexity of Markov Decision Problems (MDP) is well studied. For a detailed exposition of these issues see Littman et al. (2013), Blondel and Tsitsiklis (2000). The discussion in this section focuses on the evaluation step. We embed this discussion in the context of policy optimization in §8.

Recall that $N = |\mathcal{S}|$ is the number of states and $L = |\mathcal{M}|$ is the number of meta states. Per Lemma 6.1 we know $L = \mathcal{O}(N^{1-s})$. Value of $s$ closer to 1 are less expensive but more inaccurate. Let range $r$ be smallest integer such that $u_i - \ell_i \leq r$, for all $i \in [d]$, we have also that $N \leq (r + 1)^d$ and, in turn, that $L = \mathcal{O}(r^d(1-s))$.

Two ingredients determine the computational value of our approach. The first is the gain from matrix inversion. This is a gain that is embedded in aggregation and is independent of moment matching. The second is the loss inherent to our moment-based computation of the aggregation matrices $G, U$. We treat these two ingredients separately.

**Matrix inversion.** Computationally speaking, the key step in solving for $V = (I - \alpha P)^{-1} c$ is the inversion of the $N \times N$ matrix $(I - \alpha P)$. The complexity of matrix inversion is $\Omega(N^2 \log(N))$ (see Tveit (2003)) but $\mathcal{O}(N^3)$ is achieved by the standard Gauss-Seidel inversion. Solving for the
aggregate value $R \in \mathbb{R}^L$, on the other hand, requires the inversion of the smaller $L \times L$ matrix $(I - \alpha UPG)$ so that

$$\text{Gain} = \Omega(N^2 \log N) - O(L^3).$$

When $s \geq \frac{1}{3}$, $L^3 = O(N^{3(1-s)}) = O(N^2)$, so we have a gain of

$$\Omega(N^2 \log N - N^2) = \Omega(N^2 \log N).$$

This is a conservative estimate of the gain. On one hand, no known algorithm achieves the $N^2 \log N$ lower bound and, on the other, various algorithms are faster than Gauss-Seidel and require less than $O(L^3)$ for the aggregate problem. If we fix the inversion algorithm (to, say, Gauss-Seidel inversion) the aggregate matrix inversion takes $O(N^{3-3s})$ compared to $O(N^3)$ for the full one, approximately square root the time complexity when $s$ is close to $\frac{1}{3}$.

**Moment matching.** The matrix $G$ can be constructed as a linear program\(^4\). Leveraging our coarse grid scheme, we construct $G$ explicitly in (9). These operations take $O(Nd^2)$ time. This is compared against the gain of at least $O(N^2 \log N)$ with $s \geq \frac{1}{3}$. The total gain is then

$$\Omega(N^2 \log N - Nd^2).$$

### 8. Policy optimization

Some control notations to start. We let $A(x)$ be the set of feasible controls in state $x \in S$. We use the notation $\pi$ for a stationary policy; it is the function from $S \rightarrow A := \bigcup_{x \in S} A(x)$ such that $\pi(x)$ is the action the policy takes in state $x$. Let $p_{xy}^a$ denote the probability of transitioning from $x$ to $y$ under the action $a \in A(x)$, and $P^\pi$ for the transition matrix under policy $\pi$; $E^\pi_a$ (or respectively $E^\pi$) is the corresponding expectation.

The Bellman operator for a fixed policy $\pi$ is given by

$$T^\pi V(x) = c(x, \pi(x)) + \alpha[P^\pi V](x),$$

so that the value under $\pi$ is the solution to the fixed point equation $V^\pi = T^\pi V^\pi$ which is solved by matrix inversion; recall \(^3\). The optimization Bellman operator $T$ is given by

$$TV(x) = \max_{a \in A(x)} \{c(x, a) + \alpha[P^a V](x)\},$$

and the optimal value $V^*$ is the unique solution of the Bellman optimality equation $V^* = TV^*$. Denote the minimizing policy with $\pi^*$; if there are multiple optimal policies, we arbitrarily pick one.

\(^4\) It would have, per state $y$, $2^d$ variables (as the number of box corners) and $d + 1$ constraints (one constraint for each dimension $i \in [d]$ and an additional stochasticity constraint)
The first and second local moments depend on the state and the action taken in that state. We write
\[
\mu_a(x) = \mathbb{E}_x^a[X_1 - x], \quad \text{and} \quad \sigma^2_a(x) = \mathbb{E}_x^a[(X_1 - x)(X_1 - x)^\top], \quad x \in S,
\]
and denote with \(\tilde{\cdot}\) all analogous definitions for a sister chain.

The optimal aggregate value function is the fixed point of
\[
R(k) = \sum_{x \in S} u_{kx} \min_{a \in A(x)} \sum_{y \in S} p_{xy}^a [c(x, a) + \alpha \sum_{l \in M} g_{yl} R(l)].
\]
Although the value is defined only for \(k \in \mathcal{M}\), the minimizing policy, note, is defined on the full state space \(S\). We measure the performance of the approximate policy \(\pi'\) by comparing its value \((V^{\pi'})\) to the optimal value \((V^*)\):
\[
|V^*(x) - V^{\pi'}(x)| \quad \text{is the optimality gap.}
\]

### 8.1. Approximate PI with MoMa aggregation

The bound for a fixed policy in Lemma 4.1 extends to optimality gap:

**Lemma 8.1 (optimality gap)** Consider focal chain \(\mathcal{C}\) optimal value and policy \(V^*, \pi^*\) and the sister chain \(\mathcal{C}\) optimal \(\tilde{V}^*, \tilde{\pi}^*\). Then,
\[
|V^*(x) - \tilde{V}^*(x)| \leq \frac{\alpha}{1 - \alpha} (\delta_{V^*}[P^{\pi^*}, \tilde{P}^{\tilde{\pi}^*}] + \delta_{\tilde{V}^*}[P^{\pi^*}, \tilde{P}^{\tilde{\pi}^*}]).
\]

The following is proved then similarly to Theorem 2.

**Theorem 3**
\[
|V^* - \tilde{V}^*|_S \leq \frac{1}{1 - \alpha} \left( |V^* - GV^*|_S + |\tilde{V}^* - G\tilde{V}^*|_S \right).
\]

Thus the gap depends similarly on how well the optimal value under focal chain \(P\) and sister chain \(\tilde{P}\) are approximated by interpolating values at their nearest grid-points, weighed by the appropriate probabilities.

Once the policies \(\pi^*\) and \(\tilde{\pi}\) are fixed, moment matching supports—as seen in earlier sections—a small approximation gap with non-negligible computational gains. In embedding this within policy iteration it is important (indeed central) that our construction of the aggregation matrices \(G, U\) does not depend on the policy and, hence, does not have to be updated in each iteration.

The base algorithm is the aggregate analogue of standard policy iteration (PI) and alternates between evaluation and updating steps. Exact evaluation and/or update are replaced by approximate computations in approximate policy iteration (API); using aggregation, the \(k^{th}\) iteration proceeds as follows:
(i) Evaluation: for current policy $\pi^k$ and induced $P^{\pi^k}$, compute $R^k = (I - \alpha U P^{\pi^k} G)^{-1}Uc$ (see (2),(3)).

(ii) Update: find policy $\pi^{k+1}$ that satisfies

$$\pi^{k+1}(x) \in \arg\min_{a \in A(x)} \left\{ c(x, a) + \alpha \sum_{y \in S} \sum_{l \in M} g_{yl} R^k(l) \right\}$$

This is nothing but policy iteration for a chain on $\mathcal{M}$ with transition matrix $UPG$ so that, as follows from general theory, it is guaranteed to converge; see for example (Puterman, 1994, Proposition 6.4.2).

**Algorithm 3 (MoMa API)**

**Input:** Spacing exponent $s \in \left[ \frac{1}{3}, \frac{1}{2} \right)$.

**Output:** Policy $\pi^*$.  

1: MoMa aggregation: Obtain $U, G$ using Algorithm 1.

2: Set initial control on representative states $\bar{\pi}^0 : S^0 \rightarrow A$.

3: Compute the induced transition $\bar{P}^{\pi^0} : S^0 \rightarrow S$. Set $\bar{P}^0 \leftarrow \bar{P}^{\pi^0}$.

4: while convergence criterion is not met do

5: Policy Evaluation: Compute $R^k = (I - \alpha \bar{P}^k G)^{-1}Uc$.

6: Policy Update:

$$\bar{\pi}^{k+1}(x_l) \leftarrow \arg\min_{a \in A(x_l)} \left\{ c(x_l, a) + \alpha [\bar{P}^k G R^k](x_l) \right\}, \text{ for } x_l \in S^0,$$

$$\bar{P}^{k+1} \leftarrow \bar{P}^{\pi^{k+1}}.$$

7: Full Update:

$$\tilde{\pi}(x) \leftarrow \arg\min_{a} \left\{ c(x, a) + \alpha [P^k GR](x) \right\}, \forall x \in S.$$

Into this general schema we add two ingredients:

1. **MoMa preprocess.** As detailed in [6] we build the $s$-spaced Grid, and create the binary disaggregation matrix $U(\text{Grid})$ that has $u_{lx_l} = 1$ for all grid points $x_l$. Next we compute the non-negative row-stochastic matrix $G(\text{Grid})$, so that the $y^{th}$ row is a $y$-mean distribution over the representative states. This construction does not depend on the transition matrix and, in turn, neither on the control. It is computed once and requires no update during the PI iterations.

2. **Reduction to PI on representative states.** To further reduce computational burden—especially in the updating step—we leverage a useful implication of the coarse grid scheme. In (aggregate) evaluation we solve for $R = Uc + U P^k G R$ by inversion $(I - \alpha U P^k G)^{-1}Uc$. Because
\(u_{tx} = 1\) (and \(u_{ty} = 0\) otherwise) we have \((UP^\pi)y = p_{xly}\), so that the only rows of \(P^\pi\) used are those corresponding to the representative states \(S^0 = \{x_1, \ldots, x_L\}\). Defining \(\tilde{P}^\pi\) to be the \(L \times N\) matrix with \(\tilde{p}_{x,y} = p_{xy}\) for \(x \in S^0, y \in S\), we re-write \(R = (I - \alpha \tilde{P}^\pi G)^{-1}Uc\).

Policy update can be similarly limited to \(S^0\), as they are the only states for which we wish to compute the induced \(\tilde{P}\). Define \(\tilde{\pi} : S^0 \to A\). We have at iteration \(k\)

\[
\tilde{\pi}^{k+1}(x_l) \leftarrow \arg \min_{a \in A(x_l)} \{c(x_l, a) + \alpha \tilde{p}_{aGR^k}(x_l)\} = \arg \min_{a \in A(x_l)} \{c(x_l, a) + \alpha \tilde{p}_{aGR^k}(x_l)\},
\]

where the equality follows from the fact that the \(1 \times N\) vector of probabilities \(p_{x\cdot}\) can be accessed from \(\tilde{P}\) instead of \(P\).

Thus, we can first run complete a full policy iteration on \(S^0\) and do a single update for the states \(x \in S \setminus S^0\) after convergence; this is step 7 of the algorithm.

With the reduction to representative states, convergence of Algorithm 3 also follows immediately from that of standard PI, applied here to the controlled chain \(\tilde{P}^\pi\) on \(S^0\). The value and policy to which this PI converges inherit an optimality-gap-bound from the approximation-gap-bound in Theorem 4.

### 8.2. Complexity of optimization

We expand the discussion of evaluation complexity in §7 to the policy iteration algorithm in its totality.

A potential difficulty in calculations such as these is that while the approximation algorithm is more efficient *per iteration* it might require more iterations to converge compared to the exact one, thus erasing any possible gains. Fortunately, the upper bounds on the number of iterations are much smaller for the aggregation PI compared to the exact PI because, recall, we perform updates only for states \(x \in S \setminus S^0\); see Hollanders et al. (2016); Mansour and Singh (1999); Scherrer (2013); Ye (2011).

In §7 we showed that the time used for moment matching optimization is made up for by the time saved from evaluating policies for \(L = |M| = O(N^{1-s})\) instead of \(N\) states. The time savings are *not* however limited to the evaluation step. Computation is reduced also because we perform the iterations, up to convergence, only on the representative states \(x \in S^0\).

Specifically, suppose the cost of policy update for a single state is \(m\); in the worst case \(m\) might correspond to comparing all feasible actions \(a \in A(x)\). In full PI, this is done for every state \(x \in S\) so the complexity is \(O(Nm)\). In Algorithm 3, on the other hand, we update only \(x \in S^0\), giving \(O(Lm) = O(N^{1-s}m)\). Moreover, while implicit in the algorithm, obtaining the control-induced transition matrix \(P^\pi\) at each iteration has non-negligible computation expense of \(O(N^2)\) for full PI, and reduced to \(O(LN) = O(N^{2-s})\) each in Algorithm 3.
Except for cases where the action space far exceeds the state space in magnitude, specifically $m > O(N \log N)$, the time complexity of matrix inversion in the evaluation step dominates, thus the gain in each iteration is still $O(N^2 \log N)$. These gains are multiplied by the number of iterations it takes for the aggregate values to converge; though one must also account for the time to perform one full policy update after convergence. With $T$ iterations we have

$$Gain = \Omega(TN^2 \log N - N d^d - N m) = \Omega(N^2 \log N).$$

9. Numerical experiments

We consider two operations-management problems that pose a computational challenge for exact methods. In both cases there is a natural way to scale up the complexity, starting from small instances where we can visualize the outcomes and proceeding to larger instances that test the computational benefits of MoMA. Importantly, both were studied using alternative approximation methods, providing a benchmark for our own.

All experiments reported in this section were run on a machine with Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz 3.41 GHz and 16.0GB of RAM, using 64-bit Python.

9.1. MoMa pre-process

Common to both examples is the MoMa pre-processing step as detailed in Algorithm 1. Figure visualizes the grid, representative states and the aggregating probabilities $G$ for the case of $d = 2$ and state space $[0, 40]^2 \cap \mathbb{R}^2$. The right-hand side in said figure confirms the linear scaling—in the number of states $N$—of the pre-processing.

9.2. Joint replenishment problem

A retailer carries two types of products. Demand for the products is independent (across products and time periods). There are two types of fixed ordering costs: (i) a minor ordering cost for placing an order for product $i$; and (ii) orders of both products can arrive in the same truck and a major ordering cost is incurred for each truckload. The number of truckloads then depends on the total amount ordered (of both products).

We follow the standard setup as very clearly laid out in Vanvuchelen et al. (2020). For simplicity, only full truckloads are considered.

At time $t$, the order amount $q_{i,t}$ for each item type $i = 1, 2$ is determined based on the inventory level $I_{i,t}$ at the end of the previous period. Lead time is assume to be 0 and orders arrive before the demand $d_{i,t}$ is realized. The system dynamics are given by

$$I_{i,t} = I_{i,t-1} + q_{i,t} - d_{i,t}$$
Figure 3  (LEFT) Highlighted with red ticks the representative states that form the grid. In green we plot, for one fixed meta-state, the aggregating probabilities \( G_l \) of each state into a fixed meta-state \( l \); only states that are no further than the nearest neighboring meta-states aggregate into \( l \) with positive probabilities; \( g_{yl} \) is proportional to the distance from state \( y \) to representative state \( x_l \) (RIGHT) We scale up the (two-dimensional) state space \([0,u]^2\) by raising the value of \( u \); pre-processing takes less than 25 minutes even for state space with size larger than a million.

Per-item holding cost \( H_i \) is incurred for product-\( i \) inventory per unit of time Per-item backorder cost \( B_i \) is incurred for unmet demand. The minor ordering cost for product \( i \) is \( k_i \), and \( K \) is the cost per truckload. The immediate cost function at period \( t \) is then

\[
c(I_t,q_t) = \sum_i (H_i[I_{i,t}]^+ + B_i[I_{i,t}]^- + k_i \mathbb{1}_{q_{i,t} > 0}) + K \left( \frac{\sum_i q_{i,t}}{TC} \right)
\]

where \( TC \) is the truck capacity.

9.2.1. Small instance  Demand is \( d_1 = \mathcal{U}\{0,5\}, d_2 = \mathcal{U}\{0,3\} \) and each truck can carry 6 items. The parameters are as in Table 1. The only difference between the two products is the minor ordering cost. The discount factor is \( \alpha = 0.99 \).

| item type | \( d \)  | \( H \) | \( B \) | \( k \) | \( K \) | \( l_i \) | \( u_i \) |
|-----------|--------|--------|--------|--------|--------|--------|--------|
| i=1       | \( \mathcal{U}\{0.5\} \) | 1      | 19     | 40     | 75     | -30    | 40     |
| i=2       | \( \mathcal{U}\{0.3\} \) | 1      | 10     | 75     | -30    | 40     |

Table 1  Demand parameters for the small instance of the joint replenishment problem.

We truncate the inventory for each item at 40 and the backorder is capped at 30 units for each item type; this means the order quantity must satisfy \( q_{i,t} \leq 40 - I_{i,t} \). The total number of states is \( N = 5041 \). We take the MoMa spacing exponent to be \( s = 0.45 \), resulting in \( L = 400 \) meta states.

First, we test the evaluation performance of MoMa; see Algorithm 2. The instance is small enough that we can compute the exactly optimal policy \( \pi^* \). We take \( P = P^{\pi^*} \) as the transition
function for the focal chain and compute the approximate value \( \tilde{V}(x) = c(x) + \alpha PGR(x) \), where \( R = (I - \alpha UPG)^{-1} Uc \) is the aggregate value. This is displayed against the exact value \( V = (I - \alpha P)^{-1} c \) in Figure 4; the mean and max (over the state space) of the evaluation gap as percentages of the exact value are 0.51 % and 0.92 % respectively.

We consider optimization next. We obtain the candidate policy \( \tilde{\pi} \) using Algorithm 3 and compare the value \( V^{\tilde{\pi}} \) of this policy against the optimal value \( V^* \); see Figure 4 (RIGHT). The relative optimality gap \( |V^{\tilde{\pi}} - V^*|/V^* \), has a mean of 1.38 % and max of 2.73 %. Based on simulation, Vanvuchelen et al. (2020) reports an optimality gap with mean 0.46 % max 0.91 % for their neural network method. Computation times are not reported in Vanvuchelen et al. (2020).

Theorem 2 relates our approximation’s quality to the “local linearity” of the values \( V \) and \( \tilde{V} \), i.e, to how well the value at a state is a distance-proportional interpolation of the values at the gridpoints. Figure 5 shows that such local linearity holds for the joint replenishment problem and explains why we are observing such impressive accuracy.
Next we consider a larger instance studied in [Vanvuchelen et al. (2020)]. The parameters are as reported in Table 2. In addition each truck can carry 33 items and the discount factor is $\alpha = 0.99$.

We cap the capacity at 120. For each item type, and backorder at 50 units for each item type. The state space is then $S = [-50, 120]^2$ and the total number of states is $N = |S| = 29241$; using $s = 0.45$ we have $L = 1089$ meta states.

| item type | d | H | B | k | $\ell_i$ | $u_i$ |
|-----------|---|---|---|---|---------|-------|
| i=1       | $\mathcal{U}\{15,25\}$ | 7  | 19 | 40 | 400     | -50   |
| i=2       | $\mathcal{U}\{5,15\}$   | 1  | 19 | 10 | 400     | -50   |

Table 2 Deman parameters for the larger instance of the joint replenishment problem.

The performance of MoMA is captured in Figure 7. The evaluation gap as a percent of the exact value function has mean 0.11 % and max 0.13 %. The optimality gap as a percent of the optimal value function has mean 0.32 % and max 1.29 %. In terms of computation time, MoMA API took less than 6 minutes to converge, whereas exact PI took more than 2 hours; see detailed breakdown of runtime in seconds in Table 3. Note that the time used for each step is averaged across iterations and quoted with the unit of seconds per iteration, whereas the time cost of MoMA preprocess is incurred only once and quoted with the unit of seconds. Exact runtime is not reported in [Vanvuchelen et al. (2020)].

The last step of the algorithm—to obtain the optimal actions for all states—always requires one full update. That cost is unavoidable unless one interpolates the control obtained for the representative states.

5 That is we restrict order quantity to satisfy $q_{i,t} \leq 120 - I_{i,t} + \text{minimum demand.}$
Figure 7  (TOP LEFT) Comparison of the approximately evaluated value function vs the exact against the state space. (BOTTOM LEFT) Their ratio against the Euclidean distance to the origin. (RIGHT) Comparison of the performance of the approximate policy $V^\pi$ vs the optimal value $V^*$ in (TOP), their ratio in (BOTTOM), both against the Euclidean distance to the origin.

| algorithm      | update    | compute $P$ | evaluation | # iter | MoMa preprocess | total   |
|----------------|-----------|-------------|------------|--------|----------------|---------|
| Exact PI       | 696.90 /iter | 12.76 /iter | 119.79 /iter | 9      | /              | 7597.68 |
| MoMa API       | 35.64 /iter  | 0.51 /iter  | 0.10 /iter  | 9      | 30.28          | 357.15  |

Table 3  Runtime breakdown for the larger instance of the joint replenishment problem.

9.3. Inpatient-flow optimization

This second example follows on Dai and Shi (2017) which is already re-considered in Braverman et al. (2020). It is a hospital routing problem with multiple patient types and dedicated hospital wards; patients waiting in queue for beds in their specialized wards could be routed (or “overflowed’) to a different ward in the hospital at a cost. The $J$ internal wards are the server pools in this discrete-time queuing model, and the $N_j$ beds in each constitutes the servers. Arrivals of type-$j$ in a period $t$ follow a Poisson random variable with mean $\lambda_j$, and arrivals are independent across types and time periods. Once admitted to ward $j$, a patient’s length of stay in pool $j$ (the time occupying a bed) is geometrically distributed with mean $1/p_j$.

An arriving type-$j$ patient is immediately assigned a bed in pool $j$ when available, and otherwise waits for service in an (infinite) type-$j$ queue; the queue is truncated for numerical experiments. We let $X_j(t)$ be the number of patients either in service in pool $j$ or waiting in queue $j$; we use $X(t)$ for the vector process.

While waiting, a patient in queue $j$ incurs a holding cost $H_j$ per period of delay. A waiting type-$j$ patient can be re-routed to (an unsaturated) pool $j \neq i$ at a cost of $B_{ij}$ and served immediately. A re-routing from $i$ to $j$ can happen only if there are available server in pool $j$; we do not re-route a waiting customer to another queue. This overflow decision is made at the start of the time period,
before departures and arrivals are realized. Let \( U_{ij}(t) = U_{ij}(X(t)) \) be the number of customers overflown from buffer \( i \) to pool \( j \) at time period \( t \). The action space in state \( x \) is then

\[
A(x) = \left\{ u \in \mathbb{Z}^{J \times J} \mid \sum_{i \neq j} u_{ij} \leq (N_j - x_j)^+, \sum_{j \neq i} u_{ij} \leq (x_i - N_i)^+ \right\}.
\]

The number of type-\( i \) patients routed to other pools cannot exceed the number waiting in buffer \( i \), \((x_i - N_i)^+\), and the number routed to pool \( j \) cannot exceed the number of available servers there, \((N_j - x_j)^+\).

The discrete-time dynamics are given by

\[
X_i(t) = X_i^P(t-1) + A_i(t-1) - D_i(X_i^P(t-1))
\]

where \( A_i(t) \sim \text{Poisson}(\lambda_i) \) is the number of type-\( j \) arrivals, \( D_i(x) \sim \text{Binomial}(x \wedge N_i, p_i) \) is the number of type-\( i \) departures, and where

\[
X_i^P(t-1) = X_i(t-1) + \sum_{j \neq i} U_{ji}(X(t-1)) - \sum_{j \neq i} U_{ij}(X(t-1))
\]

is the post-action state. The cost is incurred immediately after the re-routing but before arrivals and service completions (i.e., before the realization of randomness). It is given by

\[
c(x,a) = \sum_i \sum_{j \neq i} B_{ij} u_{ij} + \sum_i H_i \times (x_i - \sum_{j \neq i} u_{ij} - N_i)^+
\]

### 9.3.1. Small instance

Here we consider an instance with 2 wards so again \( S \subseteq \mathbb{Z}_2^+ \). Parameters are listed in Table 4. They are chosen so that \( \lambda_1/p_1 = 14 > 12 \) for the ward 1 and \( \lambda_2/p_2 = 8 < 12 \) for ward 2, resulting in pressure to overflow patients from the overloaded ward 1 to ward 2; \( B_{12} = 5 > B_{21} = 1 \) so that such overflow is costly.

Each of the queue is truncated at 30 and the resulting state space is \( S = [0,42]^2 \cap \mathbb{Z}_2^+ \) and \( N = |S| = 1849 \). We take the MoMa spacing exponent to be \( s = 0.45 \) resulting in \( L = 196 \) meta states.

| \( i \) | \( \lambda_i \) | \( p_i \) | \( H_i \) | \( B_{i1} \) | \( B_{i2} \) | \( N_i \) | \( \ell_i \) | \( u_i \) |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 1   | 3.5 | 0.25 | 5   | /   | 5   | 12  | 0   | 42  |
| 2   | 2.8 | 0.35 | 5   | /   | 1   | 12  | 0   | 42  |

Table 4 Parameter setting for the small instance of the hospital routing problem.

We first use MoMa for evaluation using the exact optimal policy and compare the value of the focal chain \( V = V^* \) to \( \tilde{V} \). Figure 8 (LEFT) shows the gap. The approximation seems is visibly inaccurate and the accuracy is the worst for “small” states, namely near the origin. This is consistent with our accuracy guarantees where the gap-bound is smaller the farther one is from the origin.
Figure 8 (LEFT TOP) Exact vs approximate values: against the state space. (LEFT BOTTOM) Their ratio against the Euclidean distance to origin. (RIGHT) Ratio of $V, \tilde{V}$ against the interpolated value.

Figure 8 explains this in the terms of Theorem 2. We see that $V$ does not have the clear “local linearity” we observed in the replenishment problem; at least not near the origin.

It is important to note however that the “shapes” of $V$ and $\tilde{V}$ are very similar. This matters for optimization: an optimal control $\pi^*$ satisfies

$$\pi^*(x) \in \arg\max_{a \in A(x)} \{c(x,a) + \alpha(E_x[V(X_1)] - V(x))\},$$

so the main influence of value $V$ on the prescribed action is through the increment $E_x[V(X_1)] - V(x)$. The control computed using $\tilde{V}$ will be influenced by the approximate increment

$$\tilde{E}_x[\tilde{V}(X_1)] - \tilde{V}(x) = PGR(x) - \tilde{V}(x).$$

Figure 9(LEFT) plots these increments and captures how close they are.

It is then less surprising that—where it matters most, i.e., in the context of optimization — MoMa performs exceedingly well here. This is confirmed in Figure 9(RIGHT), where $\tilde{\pi}$ is computed using MoMa API and its performance $V^{\tilde{\pi}}$ compared to the optimal $V$.

9.3.2. Three and four wards We replicate an instance studied in [Braverman et al. 2020] with 3 specialty wards with parameters as listed in 5. This instance has a load levels 0.7, i.e., $\lambda_i = 0.7N_ip_i$. We later consider also a higher 0.8 load. The discount factor is set to $\alpha = 0.99$. The total number of states is $N = 15625$, and with $s = 0.45$, there are $L = 1000$ meta states.

For this instance, exact PI converged in 8.4 hours. Meanwhile MoMa API converged in 46 minutes—a time saving of over 90 %; see Table 6 for a detailed runtime breakdown. Runtime of
Figure 9  (LEFT) The incremental changes against the Euclidean distance to the origin, (RIGHT) Their difference against the state space.

| $\lambda_i$ | $p_i$ | $H_i$ | $B_{i1}$ | $B_{i2}$ | $B_{i3}$ | $N_i$ | $\ell_i$ | $u_i$ |
|-----------|-------|-------|---------|---------|---------|-------|-------|-------|
| i = 1     | 2.8   | 0.4   | 10      | /       | 5       | 2     | 0     | 24    |
| i = 2     | 4.2   | 0.6   | 2       | 3       | /       | 7     | 0     | 24    |
| i = 3     | 0.7   | 0.1   | 6       | 7       | 9       | /     | 0     | 24    |

Table 5  Parameter setting for the 3-ward instance of the hospital routing problem with load level 0.7.

| algorithm   | update  | compute $P$ | evaluation | # iter | MoMa preprocess | total       |
|-------------|---------|-------------|------------|--------|----------------|-------------|
| Exact PI    | 5530.81 /iter | 423.30 /iter | 22.23 /iter | 5      | /              | 30327.26    |
| MoMa API    | 628.02 /iter | 46.58 /iter | 0.15 /iter | 4      | 23.20          | 2768.91     |

Table 6  Runtime breakdown for hospital routing instance with 3 wards.

Figure 10  (LEFT) Ratio of approximate values compared to convex combination of aggregate values for the constructed grid. (RIGHT) Bellman residual of the function approximation, as percentage of the approximate value function.

"below 10 minutes" is reported in [Braverman et al. (2020)](https://example.com), but without specifying whether it was for the setting of grid size 1728, 216 or 27.

The optimality gap $\frac{|V_\pi^* - V^*|}{V^*}$ of MoMA is reported in Figure 10(LEFT).
To put in context, the mean optimality gap of $0.92\%$ from MoMa API is comparable to the mean of $1.1\%$ in [Braverman et al. (2020)], while the max of $2.97\%$ compares quite favorably to the max of $20.6\%$ quoted.

We repeat this experiment with load of $0.8$ and observe similar performance; see Table 7.

Useful for larger instances, where the exact PI is no longer possible, is to consider the the Bellman residual—a common proxy for optimality. Given a candidate policy $\tilde{\pi}$, the Bellman residual is $\tilde{V}, \|T^x \tilde{V} - \tilde{V}\|$. Precise bounds—that relate the Bellman residual to the optimality gap—are developed in [Antos et al. (2008); Farahmand et al. (2010)]. In this instance the Bellman residuals, as a percentage of the approximate value, have a mean $0.80\%$ and a max $1.91\%$; see Figure 10 (RIGHT) and Table 7.

| Load level | TAPI mean error | TAPI max error | MoMa mean error | MoMa max error | max BR |
|------------|-----------------|----------------|-----------------|----------------|-------|
| 0.7        | 1.1 %           | 20.6%          | 0.92%           | 2.97%          | 1.91% |
| 0.8        | 0.5%            | 9.6%           | 0.91%           | 3.58%          | 1.04% |

Table 7  TAPI and MoMa API error (percent optimality gap) comparison with two load levels, and maximum Bellman Residual (BR) as percent of approximate value function.

We use the Bellman residual to experiment with a 4-ward instance where our computer memory resources no longer allow for the computation of the exact policy.

Here we consider $N_1 = 2, N_2 = 3, N_3 = 1, N_4 = 2$ and queue capacity of 12 each. Note that the dimensions are no longer equal in size. Here the state space size is $N = 50400$, which translates to the inversion of a matrix with 2.5 billion elements in full evaluation. Using $s = 0.45$ we have $L = 1512$.

| $\lambda_i$ | $p_i$ | $H_i$ | $B_{i1}$ | $B_{i2}$ | $B_{i3}$ | $B_{i4}$ | $N_i$ | $l_i$ | $u_i$ |
|-------------|-------|-------|----------|----------|----------|----------|-------|-------|-------|
| i = 1       | 0.32  | 0.2   | 10       | /        | 5        | 2        | 1     | 0     | 14    |
| i = 2       | 1.68  | 0.7   | 2        | 7        | /        | 1        | 2     | 0     | 15    |
| i = 3       | 0.4   | 0.5   | 6        | 7        | 9        | /        | 3     | 0     | 13    |
| i = 4       | 0.48  | 0.3   | 6        | 1        | 2        | 3        | /     | 0     | 14    |

Table 8  Parameter setting for the 4-ward instance of the hospital routing problem.

MoMa API converged in 5 iterations, taking a total of 5 hours. Even with sparse representations our memory resources do not allow for policy evaluation. However, to get a sense of the time scale (and hence a benchmark for comparison), we point out that a single full update took 14.6 hours, 17 times as much as the aggregate policy update in MoMa.

The Bellman residual of the resulting policy is shown in Figure 11, with mean of $0.32\%$ and max of $1.13\%$ relative to the approximate value function, smaller than even the 3-ward instance, which provides positive indication on the quality of the approximate policy.
10. Accuracy guarantees

We build on, and expand upon, the results of [Braverman et al. (2020)]. There, the moments \( \mu \) and \( \sigma^2 \) are derived directly from \( P \) and the focus is on the accuracy and optimality gaps that arise from using the continuous state space PDE to approximate the Bellman equation. Inherent to this then is that the error terms depend on (informally speaking) the third derivative of the solution \( \hat{V} \) to the differential equation.

Here we must correct the bounds for the case of mismatch of the second moment between \( \tilde{P} \) and \( P \). When comparing \( \tilde{V} \) to \( \hat{V} \) the accuracy will depend, as well, on bounds on the second derivative which multiplies this mismatch.

Because the guarantees depend on a PDE, some of the notation and language of that literature is unavoidable. The final result in Theorem 4 can, however, be read without familiarity with PDE language and key results.

To simplify the exposition we assume that the state space of \( P \) is unbounded and equal to all of \( \mathbb{Z}^d \). In various applications the states cannot go negative. Such “reflection” at the boundary causes issues that we will ignore for the sake of exposition. These “gaps” however are easily completed by reference to [Braverman et al. (2020)]. Also, for computation the state space is often truncated, but this is of secondary importance. We will assume that truncation is done at large enough values to have only minimal effect. This is formalized further below.

The PDE “induced” by the equation \( 0 = TV - V \) is given, recall, by

\[
0 = c(x) + \alpha \mu(x)'DV(x) + \alpha \frac{1}{2} \text{trace}(\sigma^2(x)'D^2V(x)) - (1 - \alpha)V(x).
\]  

(10)

The state space for this PDE is all of \( \mathbb{R}^d \). The moment functions \( \mu(x) = \mathbb{E}_x[X_1 - x] \) and \( \sigma^2(x) = \mathbb{E}_x[(X_1 - x)(X_1 - x)'] \), and the cost \( c(x) \), are defined only for \( x \in \mathbb{Z}^d \). With some abuse of notation \( c(x), \mu(x) \) and \( \sigma^2(x) \) in the PDE are the extensions of these to \( \mathbb{R}^d \). Assumption 10.1 below imposes condition on these extensions. Any chain on \( \mathbb{Z}^d \) whose local first and second moments are given by the functions \( \mu \) and \( \sigma^2 \) induces the same PDE. This PDE is defined relative to \( \mu, \sigma^2 \) and not
with $\tilde{\mu}, \tilde{\sigma}^2$. Under our construction in Algorithm 1, $\tilde{\mu} = \mu$, but $\sigma^2 \neq \tilde{\sigma}^2$. We mark the PDE solution, when it exists, with $\hat{V}$. The accuracy with which $\hat{V}$ approximates the true value $V$ depends on smoothness properties of $\mu, \sigma^2$ as well on the maximal jumps $|\Delta|^S$.

For a function $f : \mathbb{R}^d \to \mathbb{R}$, a constant $\vartheta \in (0, 1]$ and a set $B \subseteq \mathbb{R}^d$, we write

$$[f]_{\vartheta, B} = \max_{x, y \in B} \frac{|f(y) - f(x)|}{\|y - x\|^\vartheta}.$$ 

When $\vartheta = 1$, this is the (local) Lipschitz constant over $B$ and we drop the subscript $\vartheta$. We also remind the reader that $x \pm z$ is the set of points $\{y \in \mathbb{R}^d : \|y - x\| \leq z\}$.

**Assumption 10.1 (primitives)** The primitives $\mu, \sigma^2$ and $c$ satisfy the following assumptions

1. $\mu$ is globally bounded and Lipschitz and

$$(1 - \alpha)^{-1/2} |\mu|^B_r + (1 - \alpha)^{-1}[\mu]^B_r \leq \Gamma$$

2. $\sigma^2$ is globally bounded and Lipschitz with

$$|\sigma^2|^B_r + (1 - \alpha)^{-1/2}[\sigma^2]^B_r \leq \Gamma,$$

and satisfies the ellipticity condition: there exists $\lambda > 0$ such that

$$\lambda^{-1} |\xi|^2 \geq \sum_{i,j} \xi_i \xi_j \sigma_{ij}(x) \geq \lambda \|\xi\|^2, \text{ for all } \xi, x \in \mathbb{R}^d.$$

3. The cost function $c$ is norm-like and three times differentiable with

$$|D^i c|^B_r \leq \Gamma (1 + r^{k-i}), \text{ for } i = 0, 1, 2.$$ 

The requirement on $c$ (specifically on $[c]$) is satisfied, for example, if $c(x) = \sum_{i=1}^d c_i(x_i)$ where $c_i(\cdot) : \mathbb{R} \to \mathbb{R}$ is a polynomial of degree less than $k$. More importantly, the requirements — most importantly that on $\mu$ — specifies a relationship between the drift and the discount factor. This is the relationship that introduces a “central-limit-theorem-like” behavior. In its most basic setting, we consider $n$ random variables (and “horizon” of length $n$) and scale space by $\sqrt{n}$. Interpreting discounting as a random exponentially distribution horizon—we observe on average $1/(1 - \alpha)$ transitions. The requirement on $\mu$ means that the natural scale of the process fluctuation is $(1 - \alpha)^{-1/2}$.

To state the main result in this section, define

$$V_{-\varepsilon}[x] = \mathbb{E}_x \left[ \sum_{t=0}^{\infty} \alpha^t \frac{|c(X_t)|}{(1 + \|X_t\|)^\varepsilon} \right].$$
Theorem 4 (approximation gap) Suppose that $\widetilde{\Delta}_x \lesssim (1+(1-\alpha)\frac{1}{4}\|x\|^{\frac{1+\varepsilon}{2}})$ and that Assumption 10.1 holds. Then,

$$|V(x) - \hat{V}(x)| \lesssim \left(\frac{1}{1-\alpha}\right)^{\frac{k+3}{2}} + \frac{1}{\sqrt{1-\alpha}} V_1(x) + V_{-1}(x),$$

and the same holds with $V$ replaced by $\bar{V}$ everywhere. Consequently, for any $\kappa \geq \frac{1}{k} \left(\frac{1}{2} + \varepsilon\right) - \frac{1}{2}$ and all $x : \|x\| \geq (1 - \alpha)^{-(1+\kappa)}$,

$$|\bar{V}(x) - V(x)| \lesssim \frac{1}{\sqrt{1-\alpha}} (V_1(x) + \bar{V}_1(x) + V_{-1}(x) + \bar{V}_{-1}(x)).$$

The $\lesssim$ here does not depend on $x, \alpha$.

To prove this result we must study the PDE (10) and how well its solution approximates $V$ and $\bar{V}$. The existence and uniqueness of the PDE solution is typically considered on a smooth bounded domain and one must specify values (or derivative conditions) on the boundaries. To this end, let

$$\mathcal{B}_r := \{ x \in \mathbb{R}^d : \|x\| < r \}, \quad \tau(r) = \inf\{ t \geq 0 : X_t \notin \mathcal{B}_r \}.$$

We will effectively consider a family of PDEs with growing $r$ and establish bounds that do not depend on $r$; taking $r \uparrow \infty$ will produce Theorem [4]. Given a radius $r$, $C^{2,\theta}(\mathcal{B}_r)$ is the space of twice continuously differentiable functions $f : \mathcal{B}_r \to \mathbb{R}$ whose second derivative is Hölder continuous with parameter $\theta$, i.e., $[D^2 f]_{\theta, \mathcal{B}_r} < \infty$. The next lemma follows from standard PDE results; see (Gilbarg and Trudinger 2001, Theorem 6.14)

Define

$$\varrho := \frac{1}{\sqrt{1-\alpha}}, \quad \mathcal{B}_\varrho(x) := x \pm \varrho.$$

Lemma 10.1 (PDE derivative estimates) Fix a radius $r$ and suppose that Assumption 10.1 holds. Then, for any $\theta \in (0,1)$ the PDE (10), with the boundary condition $\hat{V}(x) = 0, x \in \partial \mathcal{B}_r$, has a unique solution $u \in C^{2,\theta}(\mathcal{B}_r)$. Furthermore, for all $x : x \pm \varrho \in \mathcal{B}_r$

$$|D^2 u|_{\mathcal{B}_\varrho} \leq \Gamma \left( \frac{\|x\|^{k-1}}{\sqrt{1-\alpha}} + \left( \frac{1}{1-\alpha} \right)^{\frac{k+1}{2}} \right).$$

The constant $k$ is as in Assumption 10.1 and $\Gamma$ does not depend on $\alpha, r, x$ but may depend on $\varrho$.

The error in the approximation of the value is bounded by the “integrated” second derivative up to the stopping time plus the “tail” of the value. When one considers only initial states in $x \in \mathcal{B}_r \subset \subset \mathcal{B}_{r,2}$ the latter is small.
Lemma 10.2 Suppose that \( \Delta_x, \bar{\Delta}_x \leq \Gamma(1 + \sqrt{\|x\|}) \) and that Assumption 10.1 holds. Then, fixing \( r \) and letting \( u \) be the solution over \( B_r \) and \( \tau = \tau(r^2) \), we have
\[
|V(x) - \hat{V}(x)| \leq \mathbb{E}_x \left[ \sum_{t=0}^{\tau(r^2)} \alpha^t |D^2\hat{V}|_{X_t, \Delta X_t} \Delta^2_{X_t} \right] + \mathbb{E}_x \left[ \sum_{t=\tau(r^2)+1}^{\infty} \alpha^t |c(X_t)| \right], \quad x \in B_r.
\]

Further, given \( \epsilon > 0 \), we can choose \( r_0 \) sufficiently large such that
\[
\mathbb{E}_x \left[ \sum_{t=\tau(r_0^2)+1}^{\infty} \alpha^t |c(X_t)| \right] \leq \epsilon, \quad x \in B_{r_0}
\]
and the same holds for \( |\hat{V}(x) - \hat{V}(x)| \) and \( \hat{V} \) with \( \mathbb{E}, \Delta \) replaced by \( \mathbb{E}, \bar{\Delta} \).

Proof of Theorem 4 Since we can make \( \epsilon \) arbitrarily small, we will simplify exposition by dropping it from further calculations below.

Plugging (2nd derivative) into Lemma 10.2
\[
|V(x) - \hat{V}(x)| \leq |V(x) - \hat{V}(x)| + |\hat{V}(x) - \hat{V}(x)|
\]
\[
\lesssim \frac{1}{\sqrt{1 - \alpha}} \mathbb{E}_x \left[ \sum_{t=0}^{\tau} \alpha^t \|X_t\|^{k-1} \Delta^2_{X_t} \right] + \frac{1}{\sqrt{1 - \alpha}} \mathbb{E}_x \left[ \sum_{t=0}^{\tau} \alpha^t \|X_t\|^{k-1} \Delta^2_{X_t} \right]
\]
\[
+ \left( \frac{1}{1 - \alpha} \right)^{\frac{k+3}{2}}.
\]

Because \( \Delta_x \lesssim 1 + (1 - \alpha)^{\frac{1}{4}} \|x\|^{\frac{1}{2}} \), \( \bar{\Delta}_x \lesssim 1 + \sqrt{1 - \alpha} \|x\|^{1-\epsilon} \) so that
\[
\frac{1}{\sqrt{1 - \alpha}} \mathbb{E}_x \left[ \sum_{t=0}^{\tau} \alpha^t \|X_t\|^{k-1} \Delta^2_{X_t} \right] \lesssim \frac{1}{1 - \alpha} \mathbb{E}_x \left[ \sum_{t=0}^{\tau} \alpha^t \|X_t\|^{k-1} \right] + \mathbb{E}_x \left[ \sum_{t=0}^{\tau} \alpha^t \|X_t\|^{k-\epsilon} \right]
\]
\[
\lesssim \frac{1}{1 - \alpha} + \frac{1}{\sqrt{1 - \alpha}} V_{-1}(x) + V_{-\epsilon}(x),
\]
and the same holds for the sister chain. This proves the first assertion of the theorem.

To establish the second assertion, we will show that for any \( \kappa \geq 0 : k - 1 + \kappa - 2\varepsilon \geq 0 \) and \( x : \|x\| \geq (1 - \alpha)^{-(1+\kappa)} \),
\[
\left( \frac{1}{1 - \alpha} \right)^{\frac{k+3}{2}} \lesssim V_{-\epsilon}(x).
\]
To see this, let \( \bar{\Delta} = \sup_x \Delta_x \). Then, for all \( t \leq t_0(\alpha) = \frac{1}{2\bar{\Delta}(1 - \alpha)} \) and \( x : \|x\| \geq (1 - \alpha)^{-(1+\kappa)} \),
\[
\|X_t\| \geq \|x\| - \Delta t_0 \geq \frac{1/2}{(1 - \alpha)^{1+\kappa}}.
\]
In turn, with probability 1,
\[
\sum_{t=0}^{t_0} \alpha^t \frac{1 + |c(X_t)|}{(1 + \|X_t\|)^{\kappa}} \geq \sum_{t=0}^{t_0} \alpha^t \left( \frac{1/2}{1 - \alpha} \right)^{k(1+\kappa) - \varepsilon} \geq \gamma \left( \frac{1/2}{1 - \alpha} \right)^{k(1+\kappa) + 1 - \varepsilon}
\]
\[\text{Notice that for the focal chain, since jumps are bounded, we can take } \varepsilon = 1.\]
for some $\gamma < 1$ (that does not depend on $\alpha, x$). The last inequality follows from the fact that, as $\alpha \uparrow 1$,
\[ \sum_{t=t_0(\alpha)}^{\infty} \alpha^t = \alpha^{t_0(\alpha)} \sum_{t=0}^{\infty} \alpha^t = \frac{1}{1-\alpha} \alpha^{t_0(\alpha)}, \]
and noting that $\alpha^{t_0(\alpha)} \rightarrow e^{-\frac{\alpha}{\epsilon}} < 1$. Finally,
\[ \left( \frac{1}{1-\alpha} \right)^{k+3} \lesssim \left( \frac{1}{1-\alpha} \right)^{k(1+\kappa)+1-\epsilon}, \]
if $k(\frac{1}{2} + \kappa) - \frac{1}{2} - \epsilon \geq 0$. \hfill \Box

We conclude this section with a reference back to the MoMa algorithm.

**Theorem 5 (optimality gap of MoMa API)** Consider a controlled chain on $\mathbb{Z}^d$. Let $\pi^*$ be the optimal policy and $\bar{\pi}$ be the MoMa policy (produced by Algorithm 3). Suppose that Assumption 10.1 holds for $c(x, \pi(x)), \mu_{\pi(x)}(x), \sigma_{\pi(x)}^2(x)$ for both $\pi \in \{\bar{\pi}, \pi^*\}$. Then,
\[ V_{\bar{\pi}}(x) - V^*(x) \lesssim V_{\pi^*}(x) + V_{\pi^*}(x). \]

**Proof:** By Theorem 1, the policy $\bar{\pi}$ produced by MoMa API is optimal for the lifted chain. In particular, $\tilde{V} = \tilde{V}^* \leq \tilde{V}^{\pi^*}$. Then by Theorem 4
\[ V_{\bar{\pi}}(x) - V^{\pi^*}(x) \leq V_{\bar{\pi}}(x) - \tilde{V}^{\pi^*}(x) + \tilde{V}^{\pi^*}(x) - \tilde{V}^{\pi^*}(x) + \tilde{V}^{\pi^*}(x) - V^{\pi^*}(x) \]
\[ \leq V_{\bar{\pi}}(x) - \tilde{V}^{\pi^*}(x) + \tilde{V}^{\pi^*}(x) - V^{\pi^*}(x) \]
\[ \lesssim V_{\pi^*}(x) + V_{\pi^*}(x). \]

In the second inequality we used $\tilde{V}^{\pi^*}(x) \leq \tilde{V}^{\pi^*}(x)$ and in the third we applied (twice) the bounds from Theorem 4. \hfill \Box

11. Concluding remarks

What we offer here is an approach to ADP that achieves a synergy between a central-limit theorem view of control (the matching of moments) and a well-established algorithmic building block (aggregation). Our paper brings algorithmic relevance to some theoretical ideas introduced in Braverman et al. (2020), which itself builds on a long list of papers on CLT-based approximations.

The key here is the identification of aggregation as a stepping stone on which to build implementable algorithms that can be “matched” with the theory that approximates a discrete problem with a continuous one. The re-interpretation of aggregation as creating a new Markov chain on the
original state space, gives rise to a flexible infrastructure on which to superimpose moment matching. The approximation is grounded in math that informs a choice of the aggregation parameters that is consistent with optimality guarantees.

Next steps.

1. **m-step moment matching.** Our choice of $GU$, recall, is such that

$$\sum_z (GU)_{yz} = y = \mathbb{E}_y[X_0].$$

The second equality is trivial—the chain at time $t = 0$ is at its initial state $y$ and serves to re-interpret what we did here. By choosing $GU$ that matches the first moment at time $t = 0$, we guarantees that the $\tilde{P} = PGU$ matches the moment at time $t = 1$:

$$\sum_z (PGU)_{yz} = \mathbb{E}_y[X_1] = y + \mu(y).$$

More generally, if one chooses $G, U$ to match the first moment at time $t = m - 1$, i.e., $\sum_z (GU)_{yz} = \mathbb{E}_y[X_{m-1}]$, then $\tilde{P}$ matches the $m$-step moment:

$$\sum_z (PGU)_{yz} = \mathbb{E}_y[X_m].$$

Why might this be valuable? In the coarse grid approximation it is inevitable that the chain $\tilde{P}$ has large jumps relative to $P$ itself. The coarser the grid the larger the jumps of $\tilde{P}$. This affects the second moment mismatch which, in turn, affects the quality of the approximation. Because it lumps multiple transitions together, the $m$-step chain $P^m$ has larger jumps than $P$ that are possibly better aligned with those of $\tilde{P}$. This, in some settings, may improve performance. To make this point more concrete, some initial developments are offered in §A.2 of the appendix.

2. **Approximation-Estimation tradeoff and a hierarchy of models:** Consider a setting where $P$ is not known in advance but rather estimated based on observations. As transitions are performed, the estimate of the matrix $P$ improves. It seems intuitively reasonable that the approximate model, based as it is on a Taylor approximations, will induce smaller variance over finite samples but, because of the approximation, will have a larger bias. Given the uncertainty early in the horizon it may, nevertheless, make sense to use a coarser and computationally cheaper model and gradually transition – as more samples are collected — to a more accurate model. Within our framework, such a gradual transition, is enabled naturally by the coarseness (spacing) exponent $s$. The bias-variance view offers then a lens through which to explore the interaction of ADP approximation and parameter estimation, one that is natural within the Tayloring/moment matching framework we put forth here.
3. Aggregation based RL. Aggregation has already been espoused as an aid in simulation-based policy iteration in Bertsekas (2018). Useful in our algorithm is the fact that the matrix $G$ does not depend on the transition probability matrix $P$ so that, in contrast to other architectures, it requires no updating within iterations. This may have useful implications for sample complexity. From an analysis perspective, the coarse grid approximation is appealing. Because the construction leads itself to a Markov chain (a lower rank one) on a subset of states, one can build on existing studies; see e.g. Haskell et al. (2016).

Acknowledgements This work is supported by NSF grant CMMI-1662294. We thank Anton Braverman and Nathalie Vanvuchelen for generously sharing their dynamic programming codes for the examples in our §.

References

Antos A, Szepesvári C, Munos R (2008) Learning near-optimal policies with bellman-residual minimization based fitted policy iteration and a single sample path. Machine Learning 71(1):89–129.

Baras JS, Borkar VS (2000) A learning algorithm for Markov decision processes with adaptive state aggregation. Decision and Control, 2000. Proceedings of the 39th IEEE Conference on, volume 4, 3351–3356 (IEEE).

Bean JC, Birge JR, Smith RL (1987) Aggregation in dynamic programming. Operations Research 35(2):215–220.

Bertsekas DP (2012) Dynamic Programming and Optimal Control, Vol. II, 4th Edition: Approximate Dynamic Programming (Athena Scientific, Belmont).

Bertsekas DP (2017) Dynamic programming and optimal control 4th edition, volume II (Athena Scientific, Belmont).

Bertsekas DP (2018) Feature-based aggregation and deep reinforcement learning: A survey and some new implementations. arXiv:1804.04577.

Bertsekas DP, Castanon DA (1989) Adaptive aggregation methods for infinite horizon dynamic programming. IEEE transactions on Automatic Control 34(6):589–598.

Bertsekas DP, Tsitsiklis JN (1996) Neuro-Dynamic Programming (Athena Scientific, Belmont).

Blondel VD, Tsitsiklis JN (2000) A survey of computational complexity results in systems and control. Automatica 36(9):1249–1274.

Braverman A, Gurvich I, Huang J (2020) On the Taylor expansion of value functions. Operations Research 68(2):631–654, URL http://dx.doi.org/10.1287/opre.2019.1903.

Dai JG, Shi P (2017) A two-time-scale approach to time-varying queues in hospital inpatient flow management. Operations Research 65(2):514–536.
Devraj AM, Meyn SP (2017) Fastest convergence for Q-learning. \textit{arXiv preprint:1707.03770}.

Duan Y, Ke T, Wang M (2019) State aggregation learning from Markov transition data. \textit{Advances in Neural Information Processing Systems}, 4486–4495.

Farahmand AM, Szepesvári C, Munos R (2010) Error propagation for approximate policy and value iteration. \textit{Advances in Neural Information Processing Systems}, 568–576.

Ghasemi M, Hashemi A, Vikalo H, Topcu U (2020) Identifying sparse low-dimensional structures in Markov chains: A nonnegative matrix factorization approach. \textit{2020 American Control Conference (ACC)}, 1093–1098 (IEEE).

Gijsbrechts J, Boute RN, Van Mieghem JA, Zhang D (2019) Can deep reinforcement learning improve inventory management? performance on dual sourcing, lost sales and multi-echelon problems. \textit{Performance on Dual Sourcing, Lost Sales and Multi-Echelon Problems (July 29, 2019)}.

Gilbarg D, Trudinger NS (2001) \textit{Elliptic Partial Differential Equations of Second Order} (Springer-Verlag, New York).

Haskell WB, Jain R, Kalathil D (2016) Empirical dynamic programming. \textit{Mathematics of Operations Research} 41(2):402–429.

Hollanders R, Gerencsér B, Delvenne JC, Jungers RM (2016) Improved bound on the worst case complexity of policy iteration. \textit{Operations Research Letters} 44(2):267–272.

Klebaner FC (2005) \textit{Introduction to stochastic calculus with applications} (World Scientific Publishing Company, Singapore).

Littman ML, Dean TL, Kaelbling LP (2013) On the complexity of solving Markov decision problems. \textit{arXiv preprint arXiv:1302.4971}.

Mansour Y, Singh S (1999) On the complexity of policy iteration. \textit{Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence}, 401–408.

Mastin A, Jaillet P (2012) Loss bounds for uncertain transition probabilities in Markov decision processes. \textit{Decision and Control (CDC), 2012 IEEE 51st Annual Conference on}, 6708–6715 (IEEE).

Prékopa A (1990) The discrete moment problem and linear programming. \textit{Discrete Applied Mathematics} 27(3):235–254.

Puterman ML (1994) \textit{Markov Decision Processes: Discrete Stochastic Dynamic Programming} (John Wiley & Sons, Hoboken).

Ross S, Izadi M, Mercer M, Buckeridge D (2009) Sensitivity analysis of pomdp value functions. \textit{Machine Learning and Applications, 2009. ICMLA’09. International Conference on}, 317–323 (IEEE).
Scherrer B (2013) Improved and generalized upper bounds on the complexity of policy iteration. *Advances in Neural Information Processing Systems*, 386–394.

Singh SP, Jaakkola T, Jordan MI (1995) Reinforcement learning with soft state aggregation. *Advances in Neural Information Processing Systems*, 361–368.

Tsitsiklis JN, Van Roy B (1996) Feature-based methods for large scale dynamic programming. *Machine Learning* 22(1-3):59–94.

Tsitsiklis JN, Van Roy B (1997) Analysis of temporal-difference learning with function approximation. *Advances in Neural Information Processing Systems*, 1075–1081.

Tveit A (2003) On the complexity of matrix inversion. *Mathematical Note* 1.

Vanvuchelen N, Gijsbrechts J, Boute R (2020) Use of proximal policy optimization for the joint replenishment problem. *Computers in Industry* 119:103239.

Whitt W (1978) Approximations of dynamic programs, i. *Mathematics of Operations Research* 3(3):231–243.

Ye Y (2011) The simplex and policy-iteration methods are strongly polynomial for the Markov decision problem with a fixed discount rate. *Mathematics of Operations Research* 36(4):593–603.
Appendix

A.1. Proofs of Auxiliary Lemmas

Proof of Lemma 4.1 This is a special case of Lemma 8.1 obtained, trivially, by assuming that the actions space $\mathcal{A}(x)$ contains a single action (so that $\pi^* = \tilde{\pi}^*$). We refer the reader to that proof later in this appendix.

Proof of Lemma 6.1 Recall that $\mathcal{S} = \times_{i=1}^d [\ell_i, u_i]$. Consider first the case that $\ell_i \geq 0$. Fix $s \in [\frac{1}{3}, \frac{1}{2}]$ and define for $k \in \mathbb{Z}_+$ the recursion $f(0) = \ell_i$ and

$$f(k + 1) = [f(k) + f^s(k)] + 1,$$  \hfill (13)

that defines, recall (8), the grid points on the $i^{th}$ axis. The number $k^*_i = \inf\{k : f(k) \geq u_i\}$ is then the number of grid points on this axis. We will bound this number by considering a continuous lower bound on $f$.

Defined for $x \geq 0$ the function $h(x) = ((1-s)x)^{\frac{1}{1-s}} + \ell_i$. Its Taylor expansion has form

$$h(k + 1) = h(k) + ((1-s)k)^{\frac{s}{1-s}} + \frac{1}{2} s ((1-s)k)^\frac{2s-1}{1-s}.$$  \hfill (14)

Note that $2s - 1 < 0$ for $s < \frac{1}{2}$, so we have $((1-s)k)^\frac{2s-1}{1-s} \leq 1$, thus the last term is upper bounded by $\frac{1}{2}$. Then combined with (13), we know that if there is a $k \in \mathbb{Z}_+$ such that $h(k) \leq f(k)$, we have $h(k') \leq f(k')$ for all $k' \in \mathbb{Z}_+$ where $k' \geq k$.

Since $h(0) = f(0) = \ell_i$, it is clear that $h(k) \leq f(k)$ for all $k \in \mathbb{Z}_+$. We argue the slightly stronger claim

$$h(x) \leq f(|x|) \text{ for } x \geq 2.$$  \hfill (14)

Let us use (14) to complete the proof of the lemma.

Defining now $k^*_h := \inf\{k : h(k) = u_i\}$ we have $f([k^*_h]) \geq h(k^*_h) = u_i$, thus $k^*_i \leq k^*_h$. Since $h$ is a continuous increasing function $h(k^*_h) = \ell_i + ((1-s)k^*_h)^{\frac{1}{1-s}} = u_i$ so that $k^*_h = (u_i - \ell_i)^{1-s}/(1-s)$, and, in turn,

$$k^*_i \leq \frac{(u_i - \ell_i)^{1-s}}{1-s}.$$  \hfill (15)

The case that the $i^{th}$ axis has $\ell_i \leq 0$. Define $f_-(0) = u_i$ and recursively $f_-(k) = [f_-(-(k-1)) - f_-(-(k-1)^{1-s})] - 1$. It follows identically that

$$k^*_f \leq \frac{(-\ell_i - (-u_i))^{1-s}}{1-s}.$$  \hfill (16)

If $\ell_i \leq 0$ and $u_i \geq 0$, we treat the negative portion $[\ell_i, 0]$ and the positive portion $[0, u_i]$ separately to obtain that the number of grid points on the $x$ axis satisfies $k^* \leq \frac{u_i^{1-s} + u_i^{1-s}}{1-s}$. In all cases then, $k$ is upper bounded by

$$\prod_i \left\{1_{\{u_i > 0\}} \frac{u_i^{1-s}}{1-s} + 1_{\{\ell_i < 0\}} \frac{(-\ell_i)^{1-s}}{1-s}\right\} \leq \prod_i \frac{2(u_i^{\ell_i} + 1)^{1-s}}{1-s} \leq \prod_i \frac{\sqrt{2} u_i^{1-s}}{1-s} = \left(\frac{\sqrt{2}}{1-s}\right)^d N^{1-s},$$
where we used \( N = \Pi_i r_i \).

It remains only to prove (14). Because \( h \) is decreasing in \( s \), the maximum value over \( s \in [\frac{1}{3}, \frac{1}{2}] \) is achieved at \( s = \frac{1}{3} \). For the basis of the induction, notice that if \( \ell_i = 0 \), \( f(1) = 1 \) and \( f(2) = 3 \), then for \( x \in [2,3) \),

\[
    h(x) = \left( (1 - s)x \right)^{\frac{1}{2}} \leq 2^{\frac{3}{2}} < 3 = f([x]).
\]

If \( \ell_i \geq 1 \), \( f(1) \geq [\ell_i + \ell_i^2] + 1 \geq \ell_i + 2 \) so that, again,

\[
    h(x) = \left( (1 - s)x \right)^{\frac{1}{2}} \leq \left( \frac{4}{3} \right)^{\frac{3}{2}} + \ell_i < 2 + \ell_i < f([x]).
\]

Suppose now that (14) holds up to \( k_1 \in \mathbb{Z}_+ \), and for \( x \in [k_1, k_1 + 1) \). Then, for such \( x \), we have by the induction assumption that \( h(x) \leq f(k_1) \) and, in turn, for \( y = x + 1 \in [k_1 + 1, k_1 + 2) \)

\[
    h(y) = h(x) + h^*(x) \pm \frac{1}{4} \leq f(k_1) + f^*(k_1) \pm \frac{1}{4} \leq f(k_1 + 1) = f([y]),
\]

and this completes the induction.

\[\square\]

**Proof of Lemma 6.2.** For fixed \( y \in S \) and \( x_i \) in its enclosing box \( B^d \), we defined

\[
g_{yl} = \Pi_{i=1}^d \left( 1 \{ [x_i]_i = \bar{s}_i \} \frac{y_i - \bar{s}_i}{\bar{s}_i - \underline{s}_i} + 1 \{ [x_i]_i = \underline{s}_i \} \frac{\bar{s}_i - y_i}{\bar{s}_i - \underline{s}_i} \right)
\]

where \( \bar{s}_i, \underline{s}_i \) are the upper and lower values along axis \( i \in [d] \) for corners of the hyperbox. There are \( 2^d \) such corners. To simplify notation, let

\[
h_i(l) = \begin{cases} 
    1 & \text{if} \ (x_i)_i = \bar{s}_i, \\
    0 & \text{if} \ (x_i)_i = \underline{s}_i.
\end{cases}
\]

Also, set \( w_i(y) = \frac{y_i - \underline{s}_i}{\bar{s}_i - \underline{s}_i} \), so that \( 0 \leq w_i(y) \leq 1 \) is such that \( y_i = w_i(y) * \bar{s}_i + (1 - w_i(y)) * \underline{s}_i = y_i \).

Finally, define \( F_i(l) = h_i(l) * w_i(y) + (1 - h_i(l)) * (1 - w_i(y)) \), so that

\[
g_{yl} = \Pi_{i=1}^d F_i(l).
\]

Since \( F_i(l) \in [0,1] \) also \( g_{yl} \in [0,1] \).

We will argue by induction that \( \sum_{l \in B^d} g_{yl} = 1 \). For \( d = 1 \), \( \sum_{l \in B^d} g_{yl} = w_1(y) + (1 - w_1(y)) = 1 \). Now suppose for \( d' = d - 1 \), \( \sum_{l \in B^{d'}} \Pi_{i=1}^{d'} F_i(l) = 1 \). Suppose also, w.l.o.g, that the indices are ordered such that for \( k = 1, \ldots, 2^{d-1} \), \( l_{2k-1}, l_{2k} \in B^{d'} \) differ only on axis \( d \), specifically \( h_{d'}(l_{2k-1}) = 1, h_{d'}(l_{2k}) = 0 \).

Then we have

\[
\sum_{l \in B^{d'}} g_{yl} = \sum_{l' \in B^{d'}} \Pi_{i=1}^{d'} F_i(l) \\
= w_{d'}(y) \Pi_{i=1}^{d'-1} F_i(l_1) + (1 - w_{d'}(y)) \Pi_{i=1}^{d'-1} F_i(l_2) + \ldots + w_{d'}(y) \Pi_{i=1}^{d'-1} F_i(l_{2^{d'-1}}) + (1 - w_{d'}(y)) \Pi_{i=1}^{d'-1} F_i(l_{2^{d'-1}}) \\
= w_{d'}(y) \sum_{k=1}^{2^{d'-1}} \Pi_{i=1}^{d'-1} F_i(l_{2k-1}) + (1 - w_{d'}(y)) \sum_{k=1}^{2^{d'-1}} \Pi_{i=1}^{d'-1} F_i(l_{2k}) \\
= (w_{d'}(y) + 1 - w_{d'}(y)) \sum_{l' \in B^{d'-1}} F_i(l') \\
= 1
\]
where the last equality is due to the inductive assumption, completing the induction.

Now we want to show \( \sum_{l \in B^d} g_{yl} x_l = y \), i.e.,

\[
\sum_{l \in B^d} [x_l]_i \Pi_{i=1}^d \left( 1 \{ [x_l]_i = \bar{s}_i \} \ast \frac{y_i - \bar{s}_i}{\bar{s}_i - \bar{s}_l} + 1 \{ [x_l]_i = \bar{s}_l \} \ast \frac{\bar{s}_i - y_i}{\bar{s}_i - \bar{s}_l} \right) = [y]_j
\]

for \( j = 1, \ldots, d \). Notice that \( [x_l]_j = h_j(l) \bar{s}_j + (1 - h_j(l)) \bar{s}_l \), so when \( d = 1 \), the quantity on the left hand side simply evaluates to \( \text{LHS} = \bar{s}_1 w_1(y) + \bar{s}_1 (1 - w_1(y)) \), satisfying the equality.

For \( d \geq 2 \),

\[
\text{LHS} = \bar{s}_j w_j(y) \sum_{l \in B^d, h_j(l) = 1} \Pi_{i \neq j} F_i(l) + \bar{s}_j (1 - w_j(y)) \sum_{l \in B^d, h_j(l) = 0} \Pi_{i \neq j} F_i(l)
\]

\[
= (\bar{s}_j w_j(y) + \bar{s}_j (1 - w_j(y))) \sum_{l \in B^{d-1}} g_{yl} = y_j
\]

using the first part. Since this is true for any \( j = 1, \ldots, d \), this concludes the proof.

\( \square \)

**Proof of Lemma 6.3.** Fix \( y \in \mathcal{S} \) and let \( \hat{x}^1, \ldots, \hat{x}^{2^d} \) be the corners of the box \( B \) that contains \( y \). The second order Taylor expansion of the function \( W_2(z) = zz^T \) has

\[
W_2(\hat{x}) = W_2(y) + DW_2(x)'(\hat{x} - y) + \frac{1}{2} \Gamma \| \hat{x} - y \|^2,
\]

where we use the fact that \( \| D^2 W_2 \| \leq \Gamma(d) \). Taking the coordinate-wise smallest corner of the box (say this is point \( l_0 \)), every point \( y \) in the box satisfies \( y_i \in [\hat{x}^0_i, \hat{x}^{l_0}_i + q(|\hat{x}^{l_0}_i|)] \) so that \( \| \hat{x} - y \|^2 \leq \max_i q^2(|\hat{x}^{l_0}_i|) \leq q^2(\| y \|) \)

Since, by construction, \( \sum_i g_{yl} \hat{x}_i = y \) we have that \( \sum_i g_{kl} (\hat{x}^l - y) = 0 \) and, in turn, that

\[
GDW_2(y) = \sum_i g_{kl} W_2(\hat{x}^l) = W_2(y) + \Gamma q^2(\| y \|).
\]

Finally, since \( \max_{y : p(y) > 0} q^2(\| y \|) \leq \Gamma q^2(\| x \| + \Delta_x) \), we conclude that

\[
PGDW_2(x) = PW_2(x) \pm \Gamma q^2(\| x \| + \Delta_x),
\]

as required.

\( \square \)

**Proof of Lemma 2.** Recall from §4 that \( \delta_f[P, \hat{P}] := |\delta_f[P, \hat{P}](\cdot)|_{\mathcal{G}} \) where

\[
\delta_f[P, \hat{P}](x) := |\mathbb{E}_x[f(X_1)] - \mathbb{E}_x[f(X_1)]| = |\hat{P} f(x) - P f(x)|.
\]
With the coarse grid scheme, it takes on the form
\[
\delta_V[P, \tilde{P}](x) = |PV(x) - \tilde{PV}(x)| = \left| \sum_y p_{xy} V(y) - \sum_y p_{xy} \sum_l g_{yl} V(x_l) \right|
\leq \sum_y p_{xy} |V(y) - \sum_l g_{yl} V(x_l)|
\]
so that \( \delta_V[P, \tilde{P}] \leq \max_{x \in S} \sum_y p_{xy} |V(y) - \sum_l g_{yl} V(x_l)| \). Then by Lemma 4.1, we have
\[
|V - \tilde{V}|^*_S \leq \max_{x \in S} \frac{1}{1 - \alpha} \sum_y p_{xy} \left( |V(y) - \sum_l g_{yl} V(x_l)| + |\tilde{V}(y) - \sum_l g_{yl} \tilde{V}(x_l)| \right)
\leq \frac{1}{1 - \alpha} \left( |V - GV|^*_S + |\tilde{V} - G\tilde{V}|^*_S \right)
\]
\[\square\]

**Proof of Lemma 10.1.** The existence and uniqueness of a solution \( \tilde{V} \in C^{2,1-\vartheta}(\overline{B}_r) \) to the Dirichlet problem follows from Assumption 10.1 and (Gilbarg and Trudinger, 2001, Theorem 6.14). Indeed, the assumed Lipschitz continuity of \( \mu, \sigma^2 \) implies, in particular, that they are both H"older continuous for any \( \vartheta \in (0, 1) \) on any bounded set and, in particular, on \( B_r \).

Let \( u \) be this solution and let us write
\[
u(x) = \frac{c(x)}{1 - \alpha} - \frac{f}{1 - \alpha}, \text{ where } f := c - (1 - \alpha)u.
\]
Since \( c \) twice continuously differentiable, \( f \) inherits its smoothness from that of \( u \) as established above. In particular,
\[
D^2u = \frac{1}{1 - \alpha} (D^2c - D^2f).
\]
Then,
\[
D^2u|_{\mathcal{E}_c(x)} \leq \frac{1}{1 - \alpha} \left( |D^2c|_{\mathcal{E}_c(x)} + |D^2f|_{\mathcal{E}_c(x)} \right) \leq \frac{1}{1 - \alpha} \Gamma \left( \|x\|^{-2} + \left( \frac{1}{1 - \alpha} \right)^{\frac{k^2}{2}} + |D^2f|_{\mathcal{E}_c(x)} \right).
\]

To complete the bounds we must bound \( |D^2f|_{\mathcal{E}_c(x)} \). The function \( f \), notice, solves the equation
\[
\frac{1}{2} \text{trace}(\sigma(y)D^2f(y)) + \mu(y)'Df(y) - (1 - \alpha)f(y) = \frac{1}{2} \text{trace}(\sigma(y)D^2c(y)) + \mu(y)'Dc(y).
\]
(15)

Bounds on the derivative of \( f \) will now follow from general derivative estimates for PDEs. To be self contained we quote here (Gilbarg and Trudinger, 2001, Theorem 6.2).

Also by (Gilbarg and Trudinger, 2001, Page 61), for \( \vartheta \in (0, 1) \) and a set \( B_r \subset \mathbb{R}^d \), \( f \in C^{2,\vartheta}(B_r) \)
\[
|f|_{2,\vartheta,B_r} := \sup_{x \in B_r} |u(x)| + \sup_{x \in B_r} D_x f(x) + \sup_{x \in B_r} D^2_x f(x) + \sup_{x,y \in B_r} \frac{D^2 f(x) - D^2 f(y)}{\|y - x\|^{\vartheta}},
\]
where \( d_x = \text{dist}(x, \partial B_r) \leq r \) is the distance from \( x \) to the boundary \( B_r \), and \( d_{x,y} = \min\{d_x, d_y\} \).
With some abuse of notation, let
\[ B_\rho(x) := \{ y : \| y - x \| \leq \rho \} = x + \rho, \quad \text{and} \quad B_\frac{\rho}{2}(x) := \{ y : \| y - x \| \leq \rho/2 \} = x + \rho/2. \]

Then, \[ |f|^*_{2,\theta,B_\rho(x)} \geq \sup_{y \in B_\rho(x)} d_y^2 \| D^2 f(y) \| \geq \sup_{y \in B_{\frac{\rho}{2}}(x)} d_y^2 \| D^2 f(y) \|. \]
For all \( y \in B_{\frac{\rho}{2}}(x) \), notice, \( d_y \geq \rho/2 \) (the distance from the boundary is greater than \( \rho/2 \)) so that
\[ |f|^*_{2,\theta,B_\rho(x)} \geq \sup_{y \in B_{\frac{\rho}{2}}(x)} d_y^2 \| D^2 f(y) \| \geq \frac{\rho^2}{4} \| D^2 f \|_{B_\rho(x)}, \]
and, thus, that
\[ |D^2 f|^*_{B_{\frac{\rho}{2}}(x)} \leq \frac{4}{\rho^2} |f|^*_{2,\theta,B_\rho(x)}. \]

In this way a bound on \( |f|^*_{2,\theta,B_\rho(x)} \) will produce the bound stated in Lemma 10.1.

**Theorem 6** (Theorem 6.2 in [Gilbarg and Trudinger (2001)]) *Let \( \Omega \) be an open subset of \( \mathbb{R}^d \), and let \( u \in C^{2,0}(\Omega) \) be a bounded solution in \( \Omega \) of the equation*
\[
\frac{1}{2} \text{trace}(\sigma^2(y)D^2u(y)) + \mu(y)'Du(y) - \beta(y)u(y) = g(y)
\]
*where \( f \) is in \( C^0(\Omega) \) and there are positive constants \( \lambda, \Lambda \) such that the coefficients satisfy*
\[
\lambda^{-1} \| \xi \|^2 \geq \sum_{i,j} \xi_i \xi_j \sigma_{ij}(x) \geq \lambda \| \xi \|^2, \quad \text{for all} \ \xi, x \in \mathbb{R}^d,
\]
*and*
\[
|\sigma^{2^{(0)}}_{0,\theta,\Omega}, \mu^{(1)}_{0,\theta,\Omega}, \beta^{(2)}_{0,\theta,\Omega} | \leq \Lambda.
\]
*Then, *
\[
|u|^*_{2,\theta,\Omega} \leq C \left( |u|^*_{0,\theta,\Omega} + |g|^{(2)}_{0,\theta,\Omega} \right),
\]
*where \( C = C(d, \theta, \lambda, \Lambda). \)*

In this theorem we take
\[
g(y) := \frac{1}{2} \text{trace}(\sigma^2(y)D^2c(y)) + \mu(y)'Dc(y) \text{ and } \beta = 1 - \alpha.
\]

Per our observations, with \( \varrho(x) \equiv (1 - \alpha)^{-1/2} \), \( |\sigma^{2^{(0)}}_{0,\theta,B_\rho(x)}, |\mu^{(1)}_{0,\theta,B_\rho(x)}, |\beta^{(2)}_{0,\theta,B_\rho(x)} | \leq \Gamma \) so we can take
\( \lambda = \Gamma \), to conclude that
\[
|D^2 f|^*_{B_{\frac{\rho}{2}}(x)} \leq 4(1 - \alpha) |f|^*_{2,\theta,B_\rho(x)} \leq \Gamma(1 - \alpha) \left( |f|^*_{B_\rho(x)} + |g|^{(2)}_{0,\theta,B_\rho(x)} \right) = \Gamma \left( (1 - \alpha) |f|^*_{B_\rho(x)} + \sqrt{1 - \alpha} |g|^*_{B_\rho(x)} + |g|^*_{B_\rho(x)} \right). \tag{16}
\]

\footnote{Here, we use the following simple fact: for a Lipschitz continuous function \( f \),
\[
\sup_{x,y} \alpha^{(2+\theta)} f(y) - f(x) \|y - x\|^\theta \leq \alpha^{2+\theta} |f|^*_{B_\rho(x)} \theta^{1-\theta} \leq \theta |f|^*_{B_\rho(x)}.\]
By our assumption on $c$ and $\mu$

$$
|g|_{B_0^*} \leq |\mu|_{B_0^*} |Dc|_{B_0^*} + |\sigma^2|_{B_0^*} |D^2c|_{B_0^*} \leq \sqrt{1-\alpha} \|x\|^{k-1} + \|x\|^{k-2},
$$

and

$$
[g]_{B_0^*} \leq |\sigma^2|_{B_0^*} |D^2c|_{B_0^*} + |\mu|_{B_0^*} |Dc|_{B_0^*} + |\mu|_{B_0^*} |Dc|_{B_0^*} + |\mu|_{B_0^*} |Dc|_{B_0^*} \leq \Gamma \left( (1-\alpha) \|x\|^{k-1} + \sqrt{1-\alpha} \|x\|^{k-2} + \|x\|^{k-3} \right),
$$

so that

$$
\sqrt{1-\alpha} |g|_{B_0^*} + [g]_{B_0^*} \leq \Gamma \left( (1-\alpha) \|x\|^{k-1} + \sqrt{1-\alpha} \|x\|^{k-2} + \|x\|^{k-3} \right). \tag{17}
$$

It remains to bound $|f|_{B_0^*}$, where, recall, $f = c - (1-\alpha)u$. We will do so directly by studying a related diffusion process. Specifically, consider the process

$$
\hat{X}_i(t) = x_i + \int_0^t \alpha \mu_i(\hat{X}_s)ds + \sum_{j=1}^d \int_0^t \alpha \sigma_{ij}(\hat{X}_s)dB_j(s), \tag{18}
$$

where $B_j(\cdot)$ is a standard Brownian motion. Our requirement in Assumption 10.1 guarantee the existence of this process as a strong solution of this stochastic differential equation; see e.g. Klebaner [2005] Theorem 5.4). It is then a standard argument that the function

$$
u(x) = \mathbb{E}_x \left[ \int_0^\tau e^{-(1-\alpha)s} c(\hat{X}_s)ds \right], \tag{19}
$$

where $\tau = \inf \{ t \geq 0 : \hat{X}_t \in \partial B_{\rho} \}$ is the solution to the PDE (10) with the boundary condition $u(x) = 0$ when $x \in \partial B_{\rho}$. By Ito’s formula [Klebaner, 2005, Chapter 4]

$$
c(\hat{X}_s) = c(x) + \sum_i \alpha \int_0^s \mu_i(\hat{X}_u) c_i(\hat{X}_u)du + \alpha \frac{1}{2} \sum_{i,j} \int_0^s \sigma_{ij}(\hat{X}_u) c_{ij}(\hat{X}_u)du + \sum_{i,j} \int_0^s c_i(\hat{X}_u) \sigma_{ij}^2(\hat{X}_u) dB_j(u).
$$

Using the boundedness of $\mu$ and $\sigma^2$ it is easily proves that, for all $x \in B_{\rho}$ (recall that (19) is defined in the larger ball $B_{\rho}^*$), $c(x) \mathbb{E}_x \left[ \int_{t=\tau}^\infty e^{-(1-\alpha)s} \right] \leq \epsilon$. Thus, we conclude that

$$
|\nu(x) - c(x)| = \left| \mathbb{E}_x \left[ \int_0^\tau e^{-(1-\alpha)s} c(\hat{X}_s)ds \right] - c(x) \right| \leq \epsilon + \int_0^\infty e^{-(1-\alpha)s} (A(s) + B(s) + C(s)ds),
$$

where

$$
A(s) := \mathbb{E}_x \left[ \sum_i \int_0^s \mu_i(\hat{X}_u) c_i(\hat{X}_u)du \right],
$$

$$
B(s) := \mathbb{E}_x \left[ \sum_{i,j} \int_0^s \sigma_{ij}^2(\hat{X}_u) c_{ij}(\hat{X}_u)du \right],
$$

$$
C(s) := \mathbb{E}_x \left[ \sum_{i,j} \int_0^s c_i(\hat{X}_u) \sigma_{ij}(\hat{X}_u)dB_j(u) \right].
$$
Since $|D^i c(x)| \leq \Gamma(1 + \|x\|^{k-i})$ for $i = 0, 1, 2$ and since $\mu, \sigma^2$ are globally bounded, we have

$$A(s) \leq \Gamma \left( s + \sum_i \mathbb{E}_x \left[ \int_0^s |\mu|_{g^e_{s-\theta}} \|\hat{X}_u\|^{k-1} du \right] \right),$$

$$B(s) \leq \Gamma \left( s + \sum_{ij} \mathbb{E}_x \left[ \int_0^s |\sigma^2|_{g^e_{s-\theta}} \|\hat{X}_u\|^{k-2} du \right] \right),$$

and

$$C(s) \leq \Gamma \left( s + \sqrt{\mathbb{E}_x \left[ \int_0^s (|\sigma|_{g^e_{s-\theta}})^2 \|\hat{X}_u\|^{2(k-1)} du \right] } \right).$$

This last bound follows, again, from a standard result on Brownian integrals (Klebaner, 2005, Theorem 4.3). From (18) and the global boundedness of $\mu$ and $\sigma^2$ we have, for any $l \in \mathbb{Z}_+$ and $t$, that (recalling (18))

$$\mathbb{E}_x[|\hat{X}(t)|] \leq \Gamma \left( \|x\| + \sum_i \mathbb{E}_x \left[ \int_0^t |\mu_i(\hat{X}_u)| du \right] + \sum_j \mathbb{E}_x \int_0^t \sigma_{ij}(\hat{X}_u) dB_j(u) \right]$$

$$\leq \Gamma((|x| + \sqrt{1 - \alpha} t + t^{1/2}),$$

where we use our assumption that $|\mu|_{g^e_{s-\theta}} \leq \Gamma \sqrt{1 - \alpha}$. Thus,

$$A(s) \leq \int_0^s \left( 1 + \sqrt{1 - \alpha} \mathbb{E}_x[\|\hat{X}_u\|^{k-1}] du \right) \leq \Gamma \left( 1 + \sqrt{1 - \alpha} (s \|x\|^{k-1} + \sqrt{1 - \alpha}^{-1} s^k + s^{k+1}) \right).$$

We can repeat the same for $B(s), C(s)$. Multiplying by $(1 - \alpha)$ we conclude

$$|c - (1 - \alpha)u| = (1 - \alpha)|\frac{c}{1 - \alpha} - u|$$

$$\leq (1 - \alpha)e + (1 - \alpha) \int_0^\infty e^{-(1-\alpha)s} (A(s) + B(s) + C(s)) ds$$

$$\leq \Gamma \left( \frac{\|x\|^{k-1}}{\sqrt{1 - \alpha}} + \left( \frac{1}{1 - \alpha} \right)^{\frac{k+1}{2}} \right). \quad (20)$$

We then have that

$$\Gamma(1 - \alpha)^{f|_{g^e_{s(x)}}} \leq \Gamma \left( \sqrt{1 - \alpha} \|x\|^{k-1} + \left( \frac{1}{1 - \alpha} \right)^{\frac{k+1}{2}} \right). \quad (21)$$

Combining this with (17) we have that

$$\frac{|D^2 f|_{g^e_{s(x)}}}{1 - \alpha} \leq \Gamma \left( \frac{\|x\|^{k+1}}{\sqrt{1 - \alpha}} + \left( \frac{1}{1 - \alpha} \right)^{\frac{k-1}{2}} \right).$$

Finally, recalling $\|D^2 c\| \leq \Gamma(1 + \|x\|^{k-2})$, we also have

$$\frac{|D^2 u|_{g^e_{s(x)}}}{1 - \alpha} = \frac{|D^2 c|_{g^e_{s(x)}}}{1 - \alpha} + \frac{|D^2 f|_{g^e_{s(x)}}}{1 - \alpha}$$

$$\leq \Gamma \left( \frac{\|x\|^{k-1}}{\sqrt{1 - \alpha}} + \frac{\|x\|^{k-2}}{1 - \alpha} + \left( \frac{1}{1 - \alpha} \right)^{\frac{k+1}{2}} \right)$$

$$\leq \Gamma \left( \frac{\|x\|^{k-1}}{\sqrt{1 - \alpha}} + \left( \frac{1}{1 - \alpha} \right)^{\frac{k+1}{2}} \right),$$
as stated.

Proof of Lemma 10.2. The proof of the first part is a simpler version of that of [Braverman et al., 2020] Theorem 1) and we refer the reader there.

We turn to second part. By the definition of $\Delta_x$, $\|X\|_{t+1} \leq \|X_t\| + \Delta_x \leq \Gamma(1 + \|X_t\| + \sqrt{\|X_t\|})$. In particular, given $\kappa$, there exists $m(\kappa)$ such that if $\|x\| \geq m(\kappa)$, $\|X_{t+1}\| \leq (1 + \kappa)\|X_t\|$. Overall,

$$
\|X_{t+1}\| \leq \max\{(1 + \kappa)\|X_t\|1\{\|X_t\| \geq m(\kappa)\}, 2m(\kappa)\}. 
$$

(22)

By Assumption 10.1 $|c(x)| \leq \Gamma(1 + \|x\|^k)$ so that $|c(X_{t+1})| \leq 1 + \max\{(1 + \kappa)\|X_t\|1\{\|X_t\| \geq m(\kappa)\}, 2m(\kappa)\}^k$. Thus,

$$
\mathbb{E}_x \left[ \sum_{t=\tau(\tilde{r}_0)+1}^{\infty} \alpha^t c(X_t) \right] \leq \mathbb{E}_x \left[ \frac{\alpha^{\tau(\tilde{r}_0)}}{1 - \alpha} \right] + \mathbb{E}_x \left[ \sum_{t=\tau(\tilde{r}_0)+1}^{\infty} \alpha^t ((1 + \kappa)^t \|x\|^k) \right].
$$

Choosing $\kappa$ such that $\beta = (1 + \kappa)^k < 1$ we then have (notice that $\alpha < \beta$)

$$
\mathbb{E}_x \left[ \sum_{t=\tau(\tilde{r}_0)+1}^{\infty} \alpha^t c(X_t) \right] \leq \frac{\mathbb{E}_x \left[ \beta^{\tau(\tilde{r}_0)} \right]}{1 - \alpha}.
$$

Equation (22) implies that, for $x \in \mathcal{B}_{\tau_0}$ $\|X\| \leq (1 + \kappa)^t \|x\| \leq (1 + \kappa)^t \tau_0$ with probability 1. In turn, $\tau(\tilde{r}_0) = \inf\{t \geq 0 : X_t \notin \mathcal{B}_{\tau_0}\} \geq \frac{\log(1 + \kappa)}{\log(1 + \kappa)}$ with probability 1, so that $\mathbb{E}_x \beta^{\tau(\tilde{r}_0)} \downarrow 0$ as $\tau_0 \uparrow \infty$. Choosing $\tau_0$ large enough then concludes the proof.

Proof of Lemma 8.1. The proof follows a standard argument; see e.g. (Bertsekas and Tsitsiklis, 1996) Proposition 6.2). Let $J_1(x) = V^*(x) + \sqrt{\frac{\alpha}{1 - \alpha}} \delta_{V*}[P^{\pi*}, \tilde{P}^{\pi*}]$.

$$
\tilde{T} J_1(x) = \min_{a \in A(x)} \left[ c(x, a) + \alpha \sum_{y \in S} \tilde{p}_{xy} J_1(y) \right] 
\leq \min_{a \in A(x)} \left[ c(x, a) + \alpha \sum_{y \in S} \tilde{p}_{xy} \left( V^*(y) + \frac{\alpha}{1 - \alpha} \delta_{V*}\left[ P^{\pi*}, \tilde{P}^{\pi*} \right] \right) \right] 
= \min_{a \in A(x)} \left[ c(x, a) + \frac{\alpha^2}{1 - \alpha} \delta_{V*}\left[ P^{\pi*}, \tilde{P}^{\pi*} \right] + \alpha \sum_{y \in S} \left[ p_{xy}^2 V^*(y) + \tilde{p}_{xy} V^*(y) - p_{xy}^2 V^*(y) \right] \right] 
\leq c(x, \pi^*(x)) + \alpha P^{\pi*} V^*(x) + \alpha \left| \tilde{P}^{\pi*} V^*(x) - P^{\pi*} V^{\pi*}(x) \right| + \frac{\alpha^2}{1 - \alpha} \delta_{V*}\left[ P^{\pi*}, \tilde{P}^{\pi*} \right] 
\leq V^*(x) + \frac{\alpha}{1 - \alpha} \delta_{V*}\left[ P^{\pi*}, \tilde{P}^{\pi*} \right] = J_1(x)
$$

The above shows that $J_1(x) \geq \tilde{T} J_1(x)$. Since $\tilde{T} J_1(x) \rightarrow \tilde{V}(x)$, we have $J_1(x) \geq \tilde{V}(x)$ by monotonicity. Now repeat for the other side and let $J_2(x) = \tilde{V}^*(x) + \sqrt{\frac{\alpha}{1 - \alpha}} \delta_{V*}[P^{\pi*}, \tilde{P}^{\pi*}]$.

$$
T J_2(x) = \min_{a \in A(x)} \left[ c(x, a) + \alpha \sum_{y \in S} p_{xy}^2 J_2(y) \right]
$$
Consider a Markov reward process

\[ V(x) = \min_{a \in \Lambda(x)} \left[ c(x, a) + \alpha \sum_{y \in S} p_{xy} - \delta_\bar{\nu}[\bar{P}\bar{\pi}] + \alpha \sum_{y \in S} [\bar{P}_{xy} - \bar{P}_{xy}] V^*(y) \right] \]

\[ V(x) = \min_{a \in \Lambda(x)} \left[ c(x, a) + \frac{\alpha^2}{1-\alpha} \delta_\bar{\nu}[\bar{P}\bar{\pi}] + \alpha \sum_{y \in S} [\bar{P}_{xy} - \bar{P}_{xy}] V^*(y) \right] \]

\[ \leq c(x, \bar{x}^*(x)) + \alpha \bar{P}^* V^*(x) + \alpha |P^* \bar{V}^* - \bar{P}^* V^*| + \frac{\alpha^2}{1-\alpha} \delta_\bar{\nu}[\bar{P}\bar{\pi}] \]

\[ \leq \bar{V}^*(x) + \frac{\alpha}{1-\alpha} \delta_\bar{\nu}[\bar{P}\bar{\pi}] = J_2(x) \]

So we have \( J_2(x) \geq T J_2(x) \). Since \( T J_2(x) \to V^*(x) \), by monotonicity we have \( J_2(x) \geq V^*(x) \). We conclude that

\[ |V^*(x) - \bar{V}^*(x)| \leq \frac{\alpha}{1-\alpha} (\delta_{V^*}[P^*, \bar{P}^*] + \delta_\bar{\nu}[\bar{P}\bar{\pi}]). \]

as stated. □

### A.2. m-step moment coupling

This section is an informal complement to the first comment in the concluding remarks [11]. There, we argued that choosing \( G, U \) to match the \((m-1)\)-step moment, i.e., \( \sum_z (GU)_yz = \mathbb{E}_y[X_{m-1}] \), \( \bar{P} \) matches the \( m \)-step moment.

\[ \sum_z (PGU)_yz = \mathbb{E}_y[X_m]. \]

A similar property holds for the second moment: if \( GU \) matches the second moment \( \mathbb{E}_y[X_{m-1} X_{m-1}^\top] \), then \( \bar{P} \) matches the second moment at time \( m \). We refer to this generalization as \( m \)-step coupling.

As an extension of MOAMa, one expects that a sister chain based on \( m \)-step coupling approximate the value of the focal chain “sampled” every \( m \) step, namely that given \( \beta \in (0, 1) \):

\[ V^m(x) := \mathbb{E}_x \left[ \sum_{t=0}^{\infty} \beta^t c(X_{mt}) \right] \approx \mathbb{E}_x \left[ \sum_{t=0}^{\infty} \beta^t c(X_t) \right] = (I - \beta \bar{P})^{-1} c. \]

Since what we want to eventually approximate is the value \( V(x) = \mathbb{E}_x[\sum_{t=0}^{\infty} \alpha^t c(X_t)] \) of the focal chain, the following simple relationship is useful.

**Lemma A.2.1** Consider a Markov reward process \((\mathcal{S}, P, c, \alpha)\). Let \( V^m(x) := \mathbb{E}_x[\sum_{t=0}^{\infty} \alpha^t c(X_{mt})] \) and \( V(x) = \mathbb{E}_x[\sum_{t=0}^{\infty} \alpha^t c(X_t)] \). Then,

\[ V^m(x) = \frac{V(x)}{1 + \sum_{k=1}^{m-1} \alpha^k} - \frac{1}{1 + \sum_{k=1}^{m-1} \alpha^k} \left( \sum_{k=1}^{m-1} \alpha^k (\mathbb{E}_x[V^m(X_k)] - V^m(x)) \right). \]
Proof: Notice that

\[ V(x) = \mathbb{E} \left[ \sum_{t=0}^{\infty} \alpha^t c(X_t) \right] \]

\[ = \mathbb{E} \left[ \sum_{t=0}^{\infty} \alpha^{mt} c(X_{mt}) \right] + \sum_{k=1}^{m-1} \alpha^k \mathbb{E} \left[ \sum_{t=0}^{\infty} \alpha^{mt} c(X_{mt}) \right] \]

\[ = V^m(x)(1 + \sum_{k=1}^{m-1} \alpha^k) + \sum_{k=1}^{m-1} \alpha^k (\mathbb{E}_x[V^m(X_k)] - V^m(x)). \]

\[ \Box \]

Figure 12  A numerical illustration of Lemma A.2.1 (LEFT) \( \alpha = 0.8 \) (RIGHT) \( \alpha = 0.99 \). The relative error is below %5 in the former and below %1 in the latter.

Supposing that the chain is ergodic, we would have that the first term approaches \((1 - \alpha)V(x)\) as \(m \uparrow \infty\) and then \(\alpha \uparrow 1\), while the second shrinks to 0. We heuristically then take the approximation

\[ V^m(x) \approx \frac{V(x)}{1 + \sum_{k=1}^{m-1} \alpha^k}. \]

Since one expects, via moment matching, that the chain \(\tilde{P}\) that matches the \(m\)-step moment has \(\tilde{V} \approx V^m\) (notice that the discount factor for \(\tilde{V}\) is \(\beta = \alpha^m\)) we arrive at the approximation

\[ (1 + \sum_{k=1}^{m-1} \alpha^k)\tilde{V}_\beta(x) \approx (1 + \sum_{k=1}^{m-1} \alpha^k)V^m(x) \approx V(x). \]

A computational implementation of \(m\)-step coupling is not straightforward. With 1-step coupling, each \(\mathbb{E}_y[X_0] = y\) can be expressed as a convex combination of its enclosing box corners. It is no
longer clear that $E_y[X_1]$ can be expressed as the a convex combination of the values $E_{x_i}[X_1]$ in the corner points $x_1,\ldots$. The grid has to be designed more carefully.

Despite of this difficulty, the following numerical examples suggests that $m$-step coupling is a direction worth exploring.

**Example 2** Consider a (non-absorbing) random walk on $[1,\ldots,N]$ with two-step coupling. We generate $G$ and $\tilde{P}$ for 2-step coupling, i.e., so that $\tilde{E}_{x}[X_1] = E_{x}[X_2]$. The random walk is a simple one (i.e., jumps up by one or down by 1) with “reflecting boundaries” $P_{12} = P_{N,N-1} = 1$. Otherwise $P_{i,i+1} \in [0.5,0.6]$; the actual value was chosen as a random number $P_{i,i+1} = 0.5 - 0.1 \ast \text{rand}()$. This chain then has a downward drift.

We construct the grid based on spacing exponent $s \in \{0.35,0.45\}$. Figure 13 shows the growth of the second-moment (mis) match between $\tilde{P}$ and $P^2$. Figure 14 displays the value comparison $(\tilde{V} - V_s)/V_s$ where $\tilde{V} = (I - \alpha\tilde{P})^{-1}c$ and the scaled value $V_s = \frac{1}{1+\sqrt{\alpha}}(I - \sqrt{\alpha}P)^{-1}c$ for $c(i) = i^2$ and discount $\alpha = 0.95$. It also displays the first moment matching (to confirm it is 0, as by design).

The one-step construction is showing inferior performance for small states $x$. This may be because the variance gap between $\tilde{P}$ and $P$ (under the one-step construction) is larger than that between $\tilde{P}$ and $P^2$ under the two-step construction; see Figure 15. Since $P^2$ has larger jumps than $P$, the two-step approximation “suffers less” from the coarse grid. Generally, we conjecture that the two-step approximation will be beneficial where the local variance is small, so that the gains in the better matching of the second moment overwhelm the approximation error in Lemma A.2.1.
Figure 14 Value comparison for Example 13 (LEFT) $s = 0.35$ (RIGHT) $s = 0.45$ (BOTTOM) For $s = 0.35$ and $S = [1, \ldots, 100]$, with (BOTTOM LEFT) the zero-mean construction of $GU$ using 1-step coupling and (BOTTOM RIGHT) the 2-step construction. The latter shows better performance for small states.

Figure 15 The difference in variance matching. The dashed line depicts the difference (mismatch) between $\hat{PW}_2$ and, respectively, $PW_2$ (on the left) and $P^2W_2$ (on the right): (LEFT) 1-step coupling (RIGHT) 2-step coupling.