Using Randomization to Break the Curse of Dimensionality[1]

AMS 556: Dynamic Programming

Research Paper Presentation

Kai Li

Applied Mathematics and Statistics, Stony Brook University

December 1, 2022

[1] Rust, J. (1997), Econometrica, 65(3): 487-516

Introduction

This paper discusses random versions of successive approximations and multigrid algorithms for computing approximate solutions to a class of finite and infinite horizon Markov decision processes.

Especially, the random algorithms presented by Rust are able to break the "curse of dimensionality" for discrete decision processes, a subclass of Markov decision processes with a continuous multi-dimensional state space but a finite action set.

Model Definitions and Notations

A Markov Decision Process (MDP) is defined through the following objects:

- A time index $t \in \{0, 1, \cdots, T\}$, $T \leq \infty$
- A state space S
- An action space A
- A family of constraint sets $s \to A(s) \subset A$
- A utility function $u(s, a)$
- A Markov transition density $p(s'|s, a)$
- A discount factor $\beta \in [0, 1)$

Model Definitions and Notations

An MDP is discrete if the state and action sets are discrete. An MDP is continuous if the state and action variables can assume a continuum of possible values.

Discrete MDPs can be solved exactly using backward induction, value iterations, and policy iterations. On the other hand, solutions to continuous MDPs can only be approximated to within some small solution tolerance $\epsilon > 0$ by discretizing the state space into a finite grid.

Model Definitions and Notations

Bellman's curse of dimensionality is an essential practical limitation that prevents one from solving continuous MDPs arbitrarily accurately.

It happens when the cardinality of the state space and/or the cardinality of the action space is too large for discrete MDPs. More generally, the curse of dimensionality is the well-known exponential increase in the time and space required to find an approximate solution to an MDP as the dimension gets larger.

Complexity Theory

Discrete computational complexity applies to finite problems that can be solved exactly. Let d be an integer indexing the size of a discrete problem. Let $\text{comp}(d)$ denote the minimal number of computer operations necessary to solve the hardest possible problem of size d (or ∞ if no algorithm capable of solving the problem).

Continuous computational complexity applies to continuous problems that cannot be solved exactly, but the true solution can be approximated to within some tolerance $\epsilon > 0$. Let d be an integer denoting the dimension of the space in which the continuous variable lives (commonly $\mathbb{R}^d$). Let $\text{comp}(\epsilon, d)$ denote the minimal computational cost to solve the hardest possible problem of size d with a maximum error of ϵ.

Complexity Theory

A discrete MDP is subject to a curse of dimensionality if $\text{comp}(d) = \Omega(2^d)$, i.e. if the lower bound on computational complexity grows exponentially fast as the dimension d increases.

A continuous MDP is subject to a curse of dimensionality if $\text{comp}(\epsilon, d) = \Omega(1/\epsilon^d)$.

If the computational complexity is bounded above by a polynomial function of d and ϵ, then the MDP is not subject to a curse of dimensionality. Such an MDP is also called tractable. A problem subject to the curse of dimensionality is called intractable.

Complexity Theory

Discrete MDPs with $|S|$ states and $|A|$ actions can be solved exactly in polynomial time, with an upper bound of $cT|A||S|^2$ for a finite MDP, where c is the time cost per arithmetic operation.

The complexity of continuous MDPs using a "one way multigrid" algorithm is given by

$$\text{comp}(\epsilon, d_s, d_a, \beta) = \Theta\left(\frac{T}{[(1-\beta)^2\epsilon]^{2d_s+d_a}}\right),$$

where d_s and d_a are, respectively, the number of state variables and the number of control variables.

Intuition behind Randomization

Rust's inspiration on using randomization to break the curse of dimensionality came from certain mathematical problems such as multivariate integration. Some mathematical problems are subject to a curse of dimensionality if we only consider deterministic algorithms. However, the problems become tractable after using randomization.

Randomization does not always succeed in breaking the curse of dimensionality. However, Rust proved that randomization does succeed in breaking the curse of dimensionality for discrete decision processes.

Definition

A discrete decision process (DDP) is an MDP with the following property: There is a finite set A such that $A(s) \subset A$ for each $s \in S$.

For simplicity, Rust assumed that $A(s) = A$ for all $s \in S$.

Discrete Decision Process

DDPs have finite action sets implies that the main work involved in carrying out the backward induction process is the numerical integration of the value function for each state. Since randomization is able to break the curse of dimensionality of numerical integration, randomization should also work for DDPs.

On the other hand, the DDP problem requires the calculation of an infinite number of multivariate integrals at each possible conditioning state $s \in S$. The DDP problem is also nonlinear in the sense that the current value function equals the maximum of the conditional expectation of the future value function. It is generally not able to break the curse of dimensionality in nonlinear problems.

It is not obvious that randomization can break the curse of dimensionality of DDP problems.

Dynamic Programming

In finite horizon problems ($T < \infty$), dynamic programming amounts to calculating the optimal decision rule $\alpha^* = (\alpha_0, \cdots, \alpha_T)$ by backward induction starting at the terminal period, T.

In the infinite horizon case $T = \infty$, there is no "last" period from which to start the backward induction to carry out the dynamic programming algorithm. However, if the per period utility functions u are uniformly bounded and the discount factor $\beta \in [0, 1)$, then we can approximate the solution by the method of successive approximations.

Bellman Operator

Definition

The Bellman operator $\Gamma : B(S) \to B(S)$ is a contraction mapping on the Banach space $B(S)$ of measurable functions $s \in S$ (under the supremum norm) defined by

$$\Gamma(V)(s) = \max_{a \in A(s)} \left[u(s,a) + \beta \int V(s') p(ds'|s,a) \right].$$

In the finite horizon case, the backward induction recursion, using the Bellman's equation, can be written as $V_{T-t} = \Gamma^t(V_T)$, $t = 0, \cdots, T$. In the infinite horizon case, Bellman's equation can be written as a fixed point condition on the Bellman operator: $V = \Gamma(V)$.

Random Bellman Operator

Definition

The random Bellman operator $\widetilde{\Gamma}_N : B \to B$ (where $B = C([0,1]^d)$[2] is given by

$$\widetilde{\Gamma}_N(V)(s) = \max_{a \in A} \left[u(s,a) + \frac{\beta}{N} \sum_{k=1}^{N} V(\widetilde{s}_k) p(\widetilde{s}_k | s, a) \right],$$

where $\{\widetilde{s}_1, \cdots, \widetilde{s}_N\}$ are iid draws with respect to Lebesgue measure λ from the unit hypercube $[0,1]^d$.

[2] B is the Banach space of all continuous, bounded functions $f : [0,1]^d \to \mathbb{R}$ under the supremum norm, $||f|| = \sup_{s \in [0,1]^d} |f(s)|$.

Performance Measures

Additionally define the linear operators $\widetilde{\Gamma}_{a,N}$ and Γ_a by

$$\widetilde{\Gamma}_{a,N}(V)(s) = u(s,a) + \frac{\beta}{N}\sum_{k=1}^{N} V(\widetilde{s}_k)p(\widetilde{s}_k|s,a),$$

$$\Gamma_a(V)(s) = u(s,a) + \beta\int V(s')p(s'|s,a)\lambda(ds').$$

Then we have

$$\widetilde{\Gamma}_N(V) = \max_{a\in A}\widetilde{\Gamma}_{a,N}(V),$$

$$\Gamma(V) = \max_{a\in A}\Gamma_a(V),$$

where the maximization on $\widetilde{\Gamma}_{a,N}(V)(s)$ is performed pointwise for each $s \in S$.

Normalized Random Bellman Operator

Note that $\widetilde{\Gamma}_N(V)$ may not be a contraction mapping almost surely because $\sum_{k=1}^N p(s_k|s,a)/N$ may not be 1. We can construct a normalized random Bellman operator $\widehat{\Gamma}_N$ so that it is guaranteed to be a contraction mapping for all N and all sample points $\{s_1, \cdots, s_N\}$:

$$\widehat{\Gamma}_N(V)(s) = \max_{a \in A} \left[u(s,a) + \beta \sum_{k=1}^N V(s_k) p_N(s_k|s,a) \right],$$

where $p_N(s_k|s,a)$ is defined by $p(s_k|s,a)/\sum_{i=1}^N p(s_i|s,a)$ if $\sum_{i=1}^N p(s_i|s,a) > 0$, or 0 otherwise.

It is easier to investigate the asymptotic properties of the sequence $\{\widetilde{\Gamma}_{a,N}(V)\}$ because it is a straightforward sample average of iid random observations.

Regularity Conditions

Rust specified the regularity conditions for DDP problems for subsequent results to apply. The regularity conditions amount to the requirement that u and p are Lipschitz continuous functions of s.

1. $S = [0,1]^d$.
2. $\exists K_u < \infty$, $\forall a \in A$, $\forall s, s' \in S$, $|u(s,a) - u(s',a)| \leq K_u |s - s'|$.
3. The transition probability has a density $p(s'|s,a)$ with respect to Lebesgue measure λ on $[0,1]^d$.
4. $\forall a \in A$, $\forall s, s', t \in S$, $|p(s'|s,a) - p(s'|t,a)| \leq K_p(s')||s - t||$ where $K_p(s')$ satisfies $\int K_p^2(s')\lambda(ds') < K_p^2 < \infty$.

Rust also assumed the following inequalities hold:

$$\max_{a \in A} \sup_{s \in S} |u(s,a)| \leq K_u,$$

$$\max_{a \in A} \sup_{s' \in S} \sup_{s \in S} p(s'|s,a) \leq K_p.$$

Randomization Complexity

Theorem

For each $V \in B$, we have

$$\sqrt{N}\left\|\Gamma(V) - \widetilde{\Gamma}_N(V)\right\| = \mathcal{O}_p(1),$$
$$\sqrt{N}\left\|\Gamma(V) - \widehat{\Gamma}_N(V)\right\| = \mathcal{O}_p(1).$$

Thus, the error in approximating $\Gamma(V)$ by $\widetilde{\Gamma}_N(V)$ is $\mathcal{O}_p(1/\sqrt{N})$, which is independent of the dimension d. Therefore, the random Bellman operator might be able to break the curse of dimensionality.

Expected Error Uniform Bound

The previous theorem is not sufficient to prove that the random Bellman operator succeed in breaking the curse of dimensionality because it is also necessary to find a uniform bound on the expected error $\mathbb{E}\left(\left|\left|\widetilde{\Gamma}_N(V) - \Gamma(V)\right|\right|\right)$.

Definition

1. Let $BL(K) \subset B$ denote the set of uniformly Lipschitz functions with Lipschitz bound K, i.e.,

$$BL(K) = \{f \in B : |f(s) - f(s')| \leq K||s - s'||, \, ||f|| \leq K\}.$$

2. Let $B(0, K)$ denote a ball of radius K centered at the origin in B, i.e.,

$$B(0, K) = \{x \in B : ||x|| \leq K\}.$$

Expected Error Uniform Bound

Theorem

For each $N \geq 1$, the expected error in the random Bellman operator $\widetilde{\Gamma}_N$ satisfies the uniform bound:

$$\sup_{p \in BL(K_p)} \sup_{V \in B(0,K_v)} \mathbb{E}\left(\left|\left|\widetilde{\Gamma}_N(V) - \Gamma(V)\right|\right|\right) \leq \frac{\gamma(d)|A|K_pK_v}{\sqrt{N}},$$

where the bounding constant $\gamma(d)$ is given by

$$\gamma(d) = \sqrt{\frac{\pi}{2}}\beta(1 + d\sqrt{\pi}C),$$

where $K_v \geq ||V||$, C is a universal constant independent of the Gaussian process $\widetilde{Z}_a(V)$ in $C([0,1]^d)$.

Expected Error Uniform Bound

Corollary

For each $N \geq 1$ the expected error in the normalized Bellman operator $\widehat{\Gamma}_N$ satisfies the uniform bound:

$$\sup_{p \in BL(K_p)} \sup_{V \in B(0,K_v)} \mathbb{E}\left(\left|\left|\widehat{\Gamma}_N(V) - \Gamma(V)\right|\right|\right) \leq 2\frac{\gamma(d)|A|K_pK_v}{\sqrt{N}},$$

where the bounding constant $\gamma(d)$ is given by

$$\gamma(d) = \sqrt{\frac{\pi}{2}}\beta(1 + d\sqrt{\pi}C),$$

where $K_v \geq ||V||$, C is a universal constant independent of the Gaussian process $\widetilde{Z}_a(V)$ in $C([0,1]^d)$.

Break the Curse of Dimensionality

Using the error bound in the previous theorem and the complexity of randomization, Rust showed that the random Bellman operator succeeds in breaking the curse of dimensionality for finite and infinite horizon DDP problems.

Random Multigrid Algorithm

The Random Multigrid Algorithm consists of the following steps:

1. Choose $\epsilon > 0$, the number of simulations N sufficiently large that the expected error in the solution will be less than ϵ.
2. Draw iid uniform random samples $\{\widetilde{s}_1, \cdots, \widetilde{s}_N\}$ which will subsequently remain fixed in each of T backward induction steps.
3. Backward induction begins with a value function $\widehat{V}_T \in \mathbb{R}^N$ given by

$$\widehat{V}_T(\widetilde{s}_i) = \arg\max_{a \in A} u(\widetilde{s}_i, a),\ i = 1, \cdots, N.$$

4. Subsequent value functions V_{T-t}, $t = 0, \cdots, T$, are generated by successive application of the $\widehat{\Gamma}_N$ operator:

$$\widehat{V}_{T-t}(\widetilde{s}_i) = \widehat{\Gamma}_N^t(V_T)(\widetilde{s}_i),\ t = 0, \cdots, T,\ s_i = 1, \cdots, N.$$

Upper Bound on the Worst Case Complexity

Theorem

Randomization breaks the curse of dimensionality of solving finite horizon DDPs: i.e., an upper bound on the worst case complexity of the class of randomized algorithms for solving a T-period DDP problem with $|A|$ possible actions, discount factor $\beta \in (0,1)$ and utility function and transition probability (u,p) satisfying regularity conditions 1 to 4 is given by

$$comp^{wor\text{-}ran}(\epsilon, d) = \mathcal{O}\left(\frac{Td^4|A|^5K_u^4K_p^4}{(1-\beta)^8\epsilon^4}\right).$$

Upper Bound on the Worst Case Complexity

Corollary

Randomization breaks the curse of dimensionality of solving infinite horizon DDPs: i.e., an upper bound on the worst case randomized complexity of the infinite horizon DDP problem is given by

$$comp^{wor\text{-}ran}(\epsilon, d) = \mathcal{O}\left(\frac{\ln[1/(1-\beta)\epsilon]d^4|A|^5K_u^4K_p^4}{|\ln(\beta)|(1-\beta)^8\epsilon^4}\right).$$

Summary

This paper has established upper bounds on the randomized complexity of finite and infinite horizon DDP problems. Rust showed that randomization succeeds in breaking the curse of dimensionality of a subclass of DDP problems satisfying a Lipchitz condition. They assumed the Lipschitz bounds K_u and K_p on (u, p) are constants independent of the problem dimension, d. The algorithms will also break the curse of dimensionality if K_u and K_p increase polynomially in d.

A Comment on "Using Randomization to Break the Curse of Dimensionality"[3]

A weakness of Rust's approach is the "needle in the haystack problem". In particular, this problem arises when the state transition function has a spike that is too sharp for the random sampler to detect. Rust claimed that this problem can lead $V_N(s, a)$ to be a poor estimate of the true expected value function and that it can be a much more serious problem in higher dimensional problems.

[3] Bray, R. (2022), Econometrica, 90(4): 1915-1929

A Comment on "Using Randomization to Break the Curse of Dimensionality"

Bray showed that Rust's approach can avoid the "needle in the haystack problem" and therefore break the curse of dimensionality only when the problem can be recast so that all but a vanishingly small fraction of state variables behave like history-independent uniform random variables.

More precisely, only around $\ln(d)$ out of d state variables may meaningfully depend on past states and actions. The other state variables must resemble exogenous iid shocks. Bray defined such a sequence of dynamic programs to be nearly memoryless.