# Difference of Convex Functions Programming for Policy Optimization in Reinforcement Learning

Extended Abstract

Akshat Kumar
Singapore Management University
akshatkumar@smu.edu.sg

## ABSTRACT

We formulate the problem of optimizing an agent's policy within the Markov decision process (MDP) model as a difference-of-convex functions (DC) program. The DC perspective enables optimizing the policy iteratively where each iteration constructs an easier-to-optimize *lower bound* on the value function using the well known concave-convex procedure. We show that several popular policy gradient based deep RL algorithms (both for discrete and continuous state, action spaces, and stochastic/deterministic policies) such as actor-critic, deterministic policy gradient (DPG), and soft actor critic (SAC) can be derived from the DC perspective. Additionally, the DC formulation enables more sample efficient learning approaches by exploiting the structure of the value function lower bound, and when the policy has a simpler parametric form, allows using efficient nonlinear programming solvers. Furthermore, we show that the DC perspective extends easily to constrained RL and partially observable and multiagent settings. Such connections provide new insight on previous algorithms, and also help develop new algorithms for RL.

## KEYWORDS

Reinforcement learning; Policy gradient; DC programming

**ACM Reference Format:**
Akshat Kumar. 2024. Difference of Convex Functions Programming for Policy Optimization in Reinforcement Learning: Extended Abstract. In *Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), Auckland, New Zealand, May 6 – 10, 2024*, IFAAMAS, 3 pages.

## 1 INTRODUCTION

In reinforcement learning (RL), an agent seeks to optimize its long term reward by repeated interactions with its environment, and using past interactions to improve its behavior policy. The RL problem is often formulated using Markov decision processes (MDPs) [17].

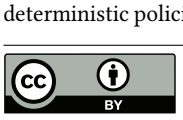
Existing RL approaches are based on policy optimization, and value iteration. In policy optimization, several policy gradient approaches have been developed that directly optimize the agent's policy using value function gradients. Policy gradient approaches have been developed for several settings such as for stochastic and deterministic policies [7, 14, 15, 18], and for continuous state, action spaces [6, 11]. Value based methods (such as Q-learning) learn the value function (or action-value function) from the data gathered via the environment simulator [20, 21]. Recently, maximum entropy RL based approaches augment the standard reward with the entropy of the policy [5, 6].

We focus on policy optimization for RL. Our contributions are as follows. *First*, we show how policy optimization can be formulated as a difference-of-convex functions (DC) program [12, 22]. *Second*, we show how several diverse policy gradient approaches for both deterministic and stochastic policies (and discrete/continuous state, action spaces, entropy-augmented rewards) can be derived from the DC perspective. In addition to being a unified policy gradient framework, the DC perspective provides opportunities for more sample efficient approaches by utilizing the structure of a lower bound on the value function derived using the concave-convex procedure (CCP), an iterative solution strategy for DC programs [12]. *Third*, we show the DC formulation extends easily for settings such as resource constrained RL [3], and multiagent decision making [9, 16].

**Related work:** There have been relatively few works exploring DC programming for RL. Piot et al. (2014) formulate optimizing the norm of the optimal Bellman residual as a DC program. However, their formulation does not show connections of DC programming with standard policy gradient methods used in RL. In contrast, our DC formulation extends easily to multiagent settings due to its close connections with policy gradient methods. Another closely related direction is probabilistic inference based optimal control. Reinforcement learning (and planning) in MDPs (and POMDPs) has been reformulated to that of probabilistic inference in a graphical model [4, 8, 10, 14, 19]. We show that for discrete state, action and stochastic policy setting, the value function lower bound optimized iteratively by CCP is exactly the same as the one derived using previous inference based approaches.

## 2 MDP AND RL SETTING

A Markov decision process (MDP) model is defined using the tuple $(S, A, T, r)$. An agent can be in one of the states $s_t \in S$ at time $t$. It takes an action $a_t \in A$, receives a reward $r(s_t, a_t)$, and the world transitions stochastically to a new state $s_{t+1}$ with probability $P(s_{t+1}|s_t, a_t) = T(s_t, a_t, s_{t+1})$. We assume that rewards are non-negative ($r \geq 0$), and future rewards are discounted using a factor $\gamma < 1$. Initial state distribution is denoted using $b_0(s)$. An agent's behavior is governed by a policy $\pi$ with $\pi(a|s)$ denoting the probability of taking action $a$ given state $s$. We first consider the standard setting with discrete state, action spaces, and stochastic policy. In the RL setting, transition and reward functions are not known.





*Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), N. Alechina, V. Dignum, M. Dastani, J.S. Sichman (eds.), May 6 – 10, 2024, Auckland, New Zealand.* © 2024 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

## 3 DC PROGRAMS AND CONCAVE-CONVEX PROCEDURE

We first describe the difference-of-convex functions (DC) programming framework, and the concave-convex procedure (CCP) to solve DC programs [12, 22]. Our goal would be to reformulate the MDP objective as an instance of DC programming. The DC programming and *concave-convex procedure* are a popular approach to optimize a general non-convex function expressed as a *difference* of two convex functions. We describe it here briefly. Consider the optimization problem:

$$\min\{g(x) : x \in \Omega\} \tag{1}$$

where $g(x) = u(x) - v(x)$ is an arbitrary function with $u$ , $v$ being real-valued *convex* functions and $\Omega$ being a (possibly non-convex) set. The CCP method provides an iterative procedure that generates a sequence of points $x^l$ by solving the following convex program:

$$x^{l+1} = \arg\min \underbrace{\{u(x) - x^T \nabla v(x^l) : x \in \Omega\}}_{\hat{g}(x;x^l)} \tag{2}$$

Each iteration of CCP monotonically decreases the objective $g(x)$ and converges to a stationary point [12]. The key benefit of CCP is that problem (2) is often much easier to solve than the original problem (1) as the objective in (2) is convex (first term is convex, and second term is linear in $x$).

## 4 DC FORMULATION FOR MDPS

We first show how the MDP objective can be viewed from the lens of DC programming. We first focus on discrete state, action spaces, and stochastic tabular policies. Let where $\tau_{0:T} = (s_0, a_0, \ldots, s_T, a_T)$ be a T-step state-action trajectory. The probability $P^\pi(\tau_{0:T})$ is:

$$P^\pi(\tau_{0:T}) = b_0(s_0)\pi(a_0|s_0)\prod_{t=1}^{T} T(s_{t-1}, a_{t-1}, s_t)\pi(a_t|s_t) \tag{3}$$

Based on the above expression, the policy objective is:

$$J(\pi) = \sum_{T=0}^{\infty}\sum_{\tau_{0:T}} b_0(s_0)\pi(a_0|s_0)\prod_{t=1}^{T}\Big[T(s_{t-1}, a_{t-1}, s_t)\pi(a_t|s_t)\Big] \times \gamma^T r_T(s_T, a_T) \tag{4}$$

As each $\pi(a|s)$ must be positive, we use the substitution $\pi(a|s) = \exp(\lambda(a|s))$, and the objective becomes:

$$J(\boldsymbol{\lambda}) = \sum_{T=0}^{\infty}\sum_{\tau_{0:T}} b_0(s_0)\Big[\prod_{t=1}^{T} T(s_{t-1}, a_{t-1}, s_t)\Big] e^{\sum_{t=0}^{T}\lambda(a_t|s_t)}\gamma^T r_T(s_T, a_T) \tag{5}$$

Notice that $J(\boldsymbol{\lambda})$ is convex in $\boldsymbol{\lambda}$ as it is a non-negative combination of terms $e^{\sum_{t=0}^{T}\lambda(a_t|s_t)}$ (initial state distribution, transition function and reward function are non-negative). We relate maximizing $J$ to the DC program structure (1) as follows:

$$\min_{\boldsymbol{\lambda}\in\Lambda} 0 - J(\boldsymbol{\lambda}) \tag{6}$$

where function $u$ is 0, and $v$ is $J(\boldsymbol{\lambda})$, resulting in the DC objective $(u - v)$. The set $\Lambda$ is the set of all $\lambda$ that satisfy the normalization constraints $\sum_a e^{\lambda(a|s)} = 1\ \forall s$. We note that constraints are non-convex. Therefore, the above problem is still a non-convex optimization problem. However, as we show next, the CCP iteration in (2) is easier to solve than the original problem. Let $\boldsymbol{\lambda}^l$ denote the current estimate; next estimate is given as $\boldsymbol{\lambda}^{l+1} = \arg\min_{\boldsymbol{\lambda}\in\Lambda}(0 - \boldsymbol{\lambda}\cdot\nabla J(\boldsymbol{\lambda}^l))$. It is given also as:

$$\max_{\lambda\in\Lambda}\sum_{s,a}\lambda(a|s)\nabla_{\lambda(a|s)}J(\boldsymbol{\lambda}^l) \tag{7}$$

The gradient at $\boldsymbol{\lambda}^l$ can be analytically derived as $\nabla_{\lambda(a|s)}J(\lambda^l) = d^l(s,a)Q^l(s,a)$ where $d^l(s,a)$ is the occupancy measure for the policy encoded by $\boldsymbol{\lambda}^l$, and $Q^l(s,a) = \mathbb{E}\big[\sum_{\alpha=0}^{\infty}\gamma^\alpha r_\alpha | s_0 = s, a_0 = a\big]$ is the action value function.

**CCP Iteration:** The CCP iteration (7) using the gradients is given as: $\max_{\lambda\in\Lambda}\sum_{s,a}\lambda(a|s)d^l(s,a)Q^l(s,a)$ In the above problem, we can again re-substitute $\lambda(a|s) = \ln\pi(a|s)$ to get:

$$\max_{\{\pi(a|s)\forall s,a\}}\sum_{s,a} d^l(s,a)Q^l(s,a)\ln\pi(a|s) \tag{8}$$

$$\sum_a \pi(a|s) = 1\ \forall s \tag{9}$$

The above optimization problem is easier to solve than the original problem—occupancy measure and action-value function are for previous parameter estimate (or previous policy), and only variables to optimize are $\pi(a|s)$. In fact, the above problem is a convex optimization problem with closed form solution easily computed by solving KKT equations [2] as $\pi^{l+1}(a|s) = \frac{\pi^l(a|s)Q^l(s,a)}{\sum_a \pi^l(a|s)Q^l(s,a)}$.

This result is already known— Toussaint and Storkey derive the same optimization formulation (8) by casting MDP planning to that of probabilistic inference; Schulman et al. use a stochastic computation graph based modeling of an MDP and arrive at the same formulation as (8). However, such formulations require rewards to not being deterministically influenced by the parameters to optimize [14]. This assumption breaks down for deterministic policies (action $a_t = \mu(s_t)$, where $\mu$ is the policy, and $r_t = r(s_t, \mu(s_t))$), and also when optimizing maximum entropy policies [6]. In contrast to such previous approaches, we show in the paper's extended version that the DC formulation is highly flexible, and even extends in such cases.

**Partially observable and multiagent setting:** We also show (in extended version) that decision making in partially observable multiagent systems using models such as decentralized partially observable MDPs [1] can be formulated as a DC program, and CCP iteration can be approximately solved using gradient based methods. Thus, the DC perspective provides a high generalization ability to solve decision making problems in a variety of settings.

## 5 CONCLUSION

We have formulated the problem of optimizing an agent's policy in an MDP framework to that of a DC program. As a result, we applied the well known CCP approach for solving DC programs to that of planning and RL. We show that such connections provided new insights on several existing policy gradient approaches in a variety of settings. Such connections open the door to develop new approaches by exploiting the DC programming for RL.

## REFERENCES


[1] Daniel S. Bernstein, Shlomo Zilberstein, and Neil Immerman. 2000. The Complexity of Decentralized Control of Markov Decision Processes. In *Conference on Uncertainty in Artificial Intelligence*.

[2] D.P. Bertsekas. 1999. *Nonlinear Programming*. Athena Scientific.

[3] Abhinav Bhatia, Pradeep Varakantham, and Akshat Kumar. 2019. Resource Constrained Deep Reinforcement Learning. In *International Conference on Automated Planning and Scheduling*. 610–620.

[4] Matthew Fellows, Anuj Mahajan, Tim G. J. Rudner, and Shimon Whiteson. 2019. VIREL: A Variational Inference Framework for Reinforcement Learning. In *Advances in Neural Information Processing Systems*. arXiv:1811.01132

[5] Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. 2017. Reinforcement learning with deep energy-based policies. In *International Conference on Machine Learning*. 2171–2186. arXiv:1702.08165 https://arxiv.org/pdf/1702.08165.pdf

[6] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*. 2976–2989. arXiv:1801.01290 http://proceedings.mlr.press/v80/haarnoja18b/haarnoja18b.pdf

[7] Vijay R. Konda and John N. Tsitsiklis. 2000. Actor-critic algorithms. *Advances in Neural Information Processing Systems* (2000), 1008–1014. https://papers.nips.cc/paper/1786-actor-critic-algorithms.pdf

[8] Akshat Kumar and Shlomo Zilberstein. 2010. MAP Estimation for Graphical Models by Likelihood Maximization. In *Advances in Neural Information Processing Systems*. 1180–1188.

[9] Akshat Kumar, Shlomo Zilberstein, and Marc Toussaint. 2015. Probabilistic Inference Techniques for Scalable Multiagent Decision Making. *J. Artif. Intell. Res.* 53 (2015), 223–270. https://doi.org/10.1613/JAIR.4649

[10] Sergey Levine. 2018. Reinforcement Learning and Control as Probabilistic Inference: Tutorial and Review. *CoRR* abs/1805.0 (2018). arXiv:1805.00909

[11] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2016. Continuous control with deep reinforcement learning. *International Conference on Learning Representations* (2016). arXiv:1509.02971 https://arxiv.org/pdf/1509.02971.pdf

[12] Thomas Lipp and Stephen Boyd. 2016. Variations and extension of the convex-concave procedure. *Optimization and Engineering* 17, 2 (2016), 263–287. https://doi.org/10.1007/s11081-015-9294-x

[13] Bilal Piot, Matthieu Geist, and Olivier Pietquin. 2014. Difference of Convex functions programming for Reinforcement Learning. In *Advances in Neural Information Processing Systems*. 2519–2527.

[14] John Schulman, Nicolas Heess, Theophane Weber, and Pieter Abbeel. 2015. Gradient estimation using stochastic computation graphs. In *Advances in Neural Information Processing Systems*, Vol. 2015-Janua. 3528–3536. arXiv:1506.05254

[15] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. 2014. Deterministic policy gradient algorithms. In *International Conference on Machine Learning*, Vol. 1. 605–619.

[16] Arambam James Singh, Duc Thien Nguyen, Akshat Kumar, and Hoong Chuin Lau. 2019. Multiagent Decision Making For Maritime Traffic Management. In *AAAI Conference on Artificial Intelligence*. 6171–6178.

[17] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement Learning: An Introduction* (second ed.). The MIT Press. http://incompleteideas.net/book/the-book-2nd.html

[18] Richard S. Sutton, David McAllester, Satinder Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*. 1057–1063.

[19] Marc Toussaint and Amos J Storkey. 2006. Probabilistic inference for solving discrete and continuous state Markov Decision Processes. In *International Conference on Machine Learning*. 945–952. https://doi.org/10.1145/1143844.1143963

[20] Mnih Volodymyr, Kavukcuoglu Koray, Silver David, Rusu Andrei A, Veness Joel, Bellemare Marc G, Graves Alex, Riedmiller Martin, Fidjeland Andreas K, and Ostrovski Georg. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529.

[21] Christopher J C H Watkins and Peter Dayan. 1992. Q-learning. In *Machine Learning*. 279–292.

[22] A. L. Yuille and Anand Rangarajan. 2003. The concave-convex procedure. *Neural Computation* 15, 4 (2003), 915–936. https://doi.org/10.1162/08997660360581958