Is Bang-Bang Control All You Need? Solving Continuous Control with Bernoulli Policies

Anonymous Author(s)
Affiliation
Address
email

Abstract

Reinforcement learning (RL) for continuous control typically employs distributions whose support covers the entire action space. In this work, we investigate the colloquially known phenomenon that trained agents often prefer actions at the boundaries of that space. We draw theoretical connections to the emergence of bang-bang behavior in optimal control, and provide extensive empirical evaluation across a variety of recent RL algorithms. We replace the normal Gaussian by a Bernoulli distribution that solely considers the extremes along each action dimension - a bang-bang controller. Surprisingly, this achieves state-of-the-art performance on several continuous control benchmarks - in contrast to robotic hardware, where energy and maintenance cost affect controller choices. Since exploration, learning, and the final solution are entangled in RL, we provide additional imitation learning experiments to reduce the impact of exploration on our analysis. Finally, we show that our observations generalize to environments that aim to model real-world challenges and evaluate factors to mitigate the emergence of bang-bang solutions. Our findings emphasise challenges for benchmarking continuous control algorithms, particularly in light of potential real-world applications.[1]

1 Introduction

Real-world robotics tasks commonly manifest as control problems over continuous action spaces. When learning to act in such settings, control policies are typically represented as continuous probability distributions that cover all feasible control inputs - often Gaussians. The underlying assumption is that this enables more refined decisions compared to crude policy choices such as discretized controllers, which limit the search space but induce abrupt changes. While switching controls can be undesirable in practice as they may challenge stability and accelerate system wear-down, they are theoretically feasible and even arise as optimal strategies in some settings. It is therefore important to investigate our underlying assumption in designing policies for learning agents, and analyze how deviations from expected behavior can be explained.

Practitioners have empirically observed that even under Gaussian policies extremal switching, or bang-bang control, may naturally emerge (e.g. in [21, 34, 43]). Thus, recent work focused on developing methods for preventing this behavior [7] or improving training when bang-bang control is the optimal policy structure [24]. However, understanding the performance, extent, and reasons for emergence of Bang-Bang policies in RL is largely an open research question. Its answer is important for designing future benchmarks and informing empirical and theoretical RL research directions.

In this paper we address these questions in two ways. First, we provide a theoretical intuition for bang-bang behavior in reinforcement learning. This is based on drawing connections to minimum-

[1] Please find videos and additional details at https://sites.google.com/view/bang-bang-rl

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

time problems where bang-bang control is often provably optimal. Second, we perform a set of experiments which optimize controllers via on-policy and off-policy learning as well as model-free and model-based state-of-the-art RL methods. Therein we compare the original algorithms with a slight modification where the Gaussian policy head is replaced with the Bernoulli distribution resulting in a bang-bang controller.

In addition to theoretical justifications, our empirical results confirm emergence of bang-bang policies in standard continuous control benchmarks. Across the board, our experiments also demonstrate high performance of explicitly enforced bang-bang policies: the modified policy head can even outperform the original method. In connection to optimal control theory, we demonstrate how action costs empirically lead to sub-optimal performance with bang-bang controllers. However, we also show the negative impact of action penalties on exploration with Gaussian policies. Due to the necessity of exploration in RL to find the optimal solution, this can result in a complex trade-off.

We also provide a discrete action space version of Maxmimum A Posteriori Policy Optimization (MPO) [1] that avoids relaxations or high variance gradient estimators, which many algorithms rely on when replacing Gaussians with discrete distributions. This is particularly useful as exploration, learning process and final performance are highly entangled in RL, and inaccurate or biased gradients can strongly affect learning. We furthermore aim to mitigate this entanglement by including results for distilling behaviour of a trained Gaussian agent into both a continuous and a discrete agent to compare their performance and provide further evidence for emergent bang-bang behavior.

In summary, our work contains the following key contributions:

1. We show competitive performance of bang-bang control on standard continuous control benchmarks by adapting several recent RL algorithms to leverage Bernoulli policies.
2. We draw theoretical connections to optimal control, motivating the emergence of bang-bang behavior in certain problem formulations even under continuous policy parameterizations.
3. We discuss possible extensions to reduce the emergence of bang-bang behaviour in RL agents, and highlight potential trade-offs this can induce with respect to exploration.

2 Related Work

Policy Representation Policies for continuous control are typically chosen to have continuous support and are commonly represented as Gaussians with diagonal covariance matrices [36, 16, 1, 18], Gaussian mixtures [46], Beta distributions [11], or latent variable models [15]. Some approaches consider discretizing the action space to facilitate exploration [14], but need to address the underlying curse of dimensionality [3, 12, 28, 42, 48]. The work by Tang and Agrawal [40] is the most closely related to our work. It shows that discretizing continuous action spaces can yield state-of-the-art performance for on-policy learning based on a sufficiently large number of discrete components. Here, we provide insight into how even the most extreme discretization of only bang-bang actions can yield state-of-the-art performance for a variety of on-policy and off-policy algorithms.

Bang-Bang Policies Bang-bang control problems have been considered very early on in reinforcement learning [44, 26, 2]. These early works mainly consider problems known to require such a policy type. While it was often observed that bang-bang policies naturally emerge [21, 34, 43] most literature focused on avoiding this behavior [7, 19, 22]. In contrast, our work focuses on understanding its nature, particularly in scenarios that were assumed to require continuous actions for obtaining an optimal controller.

Relation to Control Bang-bang controllers have been extensively investigated in optimal control research. Early works date to the 50s and 60s [37, 4] when it was shown that bang-bang policies arise as optimal controllers in minimum time problems [27]. More recent work in bang-bang control focused on, e.g., further analysis [8, 45], numerical computation [9, 31, 32], and consideration of more specialized systems [10, 25, 29, 47]. Another related line of work are switched dynamical systems where each mode corresponds to a bang-bang configuration. In these settings, one typically optimizes for switching times given a mode sequence [39] or penalizes the switching frequency as the control effort [38]. Unlike our work, research in this space typically does not consider the dynamics of the learning process in a deep RL setting. However, we can still leverage this existing body of work to inform our explanation for the seemingly surprising emergence of bang-bang policies in RL.

3 Optimal Control with Continuous-Time Deterministic Dynamics

We start by describing connections between the emergence of bang-bang behavior in optimal control and common continuous control reinforcement learning problems. First, we characterize three common reward structures together with the optimal controllers they induce. Then, we frame the solutions and their underlying assumptions in the context of our reinforcement learning setting.

We consider a continuous-time deterministic dynamical system with state $s \in \mathbb{R}^n$ and action $a \in \mathbb{R}$. The continuous time-setting simplifies analysis and provides a good approximation under the high sampling rates that are common on continuous control benchmarks and real-world robotic system. We define the non-linear control-affine system dynamics as

$$\dot{s}(t) = f(s(t)) + g(s(t))a(t), \quad 0 \leq t \leq T,$$

with $f : \mathcal{S} \to \mathcal{S}$, $g : \mathcal{S} \to \mathcal{S}$, t is the time, and T is the final time. We assume action $a(t)$ to be a scalar in the set $\mathcal{A} = \{a(t) \in \mathbb{R} \mid |a(t)| \leq 1\}$. The following derivations can be directly extended to multidimensional inputs [23, Sec. 5.2]. For simplicity, we would like to

$$\text{maximize} \quad \int_0^T r(s(t)) - c(a(t))\mathrm{d}t,$$

where $r : \mathcal{S} \to \mathbb{R}$ is the state reward and $c : \mathcal{A} \to \mathbb{R}$ is the action cost.

Maximum state reward (MS) Consider the case when $c(a(t)) = 0$, meaning that there is no action cost. The corresponding Hamiltonian is

$$H(s(t), a(t), p(t)) = r(s(t)) + p(t)^T(f(s(t)) + g(s(t))a(t)),$$

where $p(t)$ is the costate variable. From Pontryagin's maximum principle [6, Prop. 3.3.1][23, Sec. 5.4], the necessary optimality condition for action $a^\star(t) \in \mathcal{A}$ and $0 \leq t \leq T$ follows as

$$H(s^\star(t), a^\star(t), p(t)) \geq H(s^\star(t), a(t), p^\star(t)), \quad \forall a(t) \in \mathcal{A}, \tag{1}$$

$$\text{therefore,} \quad p^\star(t)^T g(s^\star(t))a^\star(t) \geq p^\star(t)^T g(s^\star(t))a(t), \quad \forall a(t) \in \mathcal{A}. \tag{2}$$

The optimal action must, then, maximize the right-hand side over $a(t) \in \mathcal{A}$, hence

$$a^\star(t) = \begin{cases} -1 & p^\star(t)^T g(s^\star(t)) < 0 \\ \text{undetermined} & p^\star(t)^T g(s^\star(t)) = 0 \\ 1 & p^\star(t)^T g(s^\star(t)) > 0. \end{cases}$$

If the undetermined condition does not happen, this is referred to as *bang-bang* control.

Minimum fuel-type cost (MF) Consider the case when $c(a(t)) = |a(t)|$. By following similar derivations as in the previous paragraph [23, Sec. 5.5], we obtain the optimal action

$$a^\star(t) = \begin{cases} -1 & p^\star(t)^T g(s^\star(t)) < -1 \\ 0 & |p^\star(t)^T g(s^\star(t))| < 1 \\ 1 & p^\star(t)^T g(s^\star(t)) > 1 \\ \text{undetermined} & |p^\star(t) g(s^\star(t))| = 1. \end{cases}$$

If the undetermined condition does not happen, this is referred to as *bang-off-bang* control. Additional derivations can be found in Appendix A.

Minimum energy-type cost (ME) Consider the case when $c(a(t)) = a(t)^2$. In general, this formulation leads to non bang-bang optimal control [6, Example 3.3.2].

Singular arcs and chattering behavior In the undetermined conditions the Hamiltonian is no longer dependent on $a(t)$ and the necessary condition does not allow for determining $a^\star(t)$. These cases are commonly referred to as *singular arcs*. A sufficient condition to avoid singular arcs is having a linear time-invariant dynamical system that is controllable [23, Sec. 5.6]. If an optimal control problem has singular arcs, a bang-bang solution may have to switch infinitely often in a finite time interval to achieve the desired state $s^\star(t)$. This behavior is referred to as *chattering* [30] or *Zeno's phenomenon* [49]. It does not appear in our formulation as we are considering discrete-time dynamical systems with a finite number of discretization points.

Domain	Cartpole	Cartpole	Cheetah	Dog	Finger	Humanoid	Quadruped	Walker
Task	Swingup	Sparse	Run	Walk	Spin	Walk	Run	Walk
Type	MS&ME	MS	MS	MS	MS	MS&ME	MS	MS

Table 1: DeepMind Control Suite tasks with their objective types based on Section 3.

Discretization effect The Pontryagin maximum principle cannot be directly extended to discrete-time dynamical systems without introducing additional assumptions. For example, Bertsekas [6, Prop. 3.3.2] derives a discrete time version which, in order to provide necessary optimality conditions, requires convexity of the set $\mathcal{A}$. If we consider only bang-bang control actions, the set $\mathcal{A}$ is not convex and the discrete-time derivations no longer hold [6, Example 3.10]. Nevertheless, we find that in this work the discretizaton intervals are small enough to not affect optimality of the learned policies.

Stochastic dynamics This simplified setup can be seen as a special case of reinforcement learning problems because of its deterministic dynamics. However, similar derivations can be done in case of stochastic action policies and dynamics. For example, this is the case when $\dot{s}(t) = f(s(t)) + g(s(t))a(t) + w(t)$, where $w(t) \in \mathbb{R}^n$ is a random disturbance. Stochastic dynamics, together with the time discretization, are the key components to bridge this setup and the standard RL frameworks, which are discussed in the next section.

Optimal controller Solving optimal control problems using the Pontryagin maximum principle is, in general, very challenging since it requires the numerical solution of a set of partial differential equations [6, 23]. Instead, we directly learn switching controllers as stochastic Bernoulli policies.

4 Reinforcement Learning Preliminaries

We formulate the control and reinforcement learning problems in the context of a Markov Decision Process (MDP) defined by the tuple $\{\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma\}$, where $\mathcal{S}$ and $\mathcal{A}$ denote the state and action space, respectively, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ represents the density of the transition distribution, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ the reward mapping, and $\gamma \in [0, 1)$ is the discount factor. Note that this reward can be obtained from the previous section on continuous time dynamics by defining $R(s_t, a_t) = \int_{\tau=t}^{t+1} r(s(\tau)) - c(a(\tau))\mathrm{d}\tau$. We define s_t and a_t to be the state and action at time t, with input constraints of the form $|a_t| \leq a_{max}$. Let $\pi_\theta(a|s)$ denote an action distribution parameterized by θ representing the policy and define the discounted infinite horizon return $G_t = \sum_{t'=t}^{\infty} \gamma^{t'-t} R(s_{t'}, a_{t'})$, where $s_{t+1} \sim \mathcal{T}(s_{t+1}|s_t, a_t)$ and $a_t \sim \pi_\theta(a_t|s_t)$. The objective is to learn the optimal policy maximizing the expected infinite horizon return $\mathbb{E}[G_t]$ under unknown dynamics and reward mappings. This can be achieved by modeling $\pi_\theta(a_t|s_t)$ as a distribution with a neural network predicting the corresponding parameters θ from s_t.

5 Experiments

In this section, we work to improve our understanding of empirical performance and characteristics of controllers learned via RL in continuous control domains. We compare performance and learnt distributions for Bernoulli and Gaussian policies on several tasks from the DeepMind Control Suite [41] based on a variety of popular RL algorithms. We then evaluate the robustness of the Bang-Bang policies to modifications of the environment and transfer under sensor disturbances. We select MPO for this comparison due to its robustness and the fact that it does not require additional changes to handle discrete distributions. Furthermore, we analyse the entanglement of exploration and final solution in RL. We focus on the final solution by distilling Bernoulli policies from a Gaussian teacher. Next, we discuss advantages and disadvantages of bang-bang behavior for exploration. Finally, we evaluate the effects of augmenting the objective by action penalties on exploration and final solutions of Gaussian and Bernoulli policies to demonstrate the trade-offs when mitigating bang-bang behavior.

Algorithm	Model	Learning	Inputs	Gradient Estimation
PPO [36]	model-free	on-policy	states	REINFORCE
SAC [16]	model-free	off-policy	states	Reparametrization
MPO [1]	model-free	off-policy	states	Expectation Maximisation
DreamerV2 [18]	model-based	off-policy	images	REINFORCE & Reparametrization

Table 2: Algorithms analyzed in conjunction with Bernoulli policies.

5.1 Algorithms

We consider Proximal Policy Optimization (PPO) [36], Soft Actor Critic (SAC) [16], Maximum A Posteriori Policy Optimization (MPO) [1] and DreamerV2 [18] as our baseline algorithms. These approaches differ widely in their learning method, optimization strategy, and input signals. A brief comparison of their characteristics is provided in Table 2 with additional description in Appendix B. Across all algorithms, we investigate the effects of replacing the commonly used Gaussian with a Bernoulli distribution and provide both qualitative and quantitative comparisons between the learned controllers. For the approaches that require reparameterization of Bernoulli policies we leverage the biased straight-through gradient (ST) estimator [5] as detailed in Appendix B. We place particular focus on MPO as a robust off-policy method which does not require gradient reparameterization and therefore avoids introducing additional estimation errors.

5.2 Solving Continuous Control Problems with Bang-Bang Policies

We investigate performance of Bang-Bang policies on several continuous control problems from the DeepMind Control Suite. Figure 1 provides learning curves for PPO, SAC, MPO, and DreamerV2.We note that restricting agents to only select minimum or maximum actions generally does not prevent learning on the tasks and even yields competitive performance. This applies across several domains and distinct algorithm designs (see Table 2).

The presence of action penalties reduces performance of the Bernoulli policies even when the resulting state-space trajectories appear optimal. For instance, bang-bang control is able to swing-up and stabilize on Cartpole Swingup but incurs steady-state cost during stabilization as it is unable to select low-magnitude actions. We also observe lower performance of Bernoulli policies for SAC on Cartpole tasks. SAC presents some unique challenges, as the target entropy parameter operates on different scales in the continuous and discrete distribution cases. We found that particularly domains with low-dimensional action spaces, such as Cartpole, can suffer from premature policy convergence. Finally, bang-bang control with DreamerV2 does not perform well on Finger Spin. The continuous policy already displays some difficulties in learning the task in comparison to PPO, SAC, and MPO. This could indicate that learning a latent model from images can be challenging on this task. Building the model under only extreme input signals potentially exacerbates these effects. Quantitatively, we observe strong performance when restricting the agent to employ Bernoulli policies.

The results in Figure 1 suggest that across most domains, we do not require a continuous action space. To investigate this phenomenon, we analyze the action distributions of converged policies. Figure 2 provides binned actions aggregated over an evaluation episode for converged Bernoulli and Gaussian policies in several domains. We note that the Gaussian policies consistently resort to extremal actions, particularly on the Finger, Walker and Quadruped tasks. The Cartpole behavior splits into two phases, leveraging bang-bang control for fast swing-up and near-zero actions during stabilization. Qualitatively, we find that the Gaussian policies tend to converge to bang-bang behavior.

5.2.1 Disentangling Exploration and Final Solution

The trained Bernoulli and Gaussian policies display strong similarities in their quantitative and qualitative performance. Since the final performance is affected by both exploration and the ability to represent the optimal controller, the Bernoulli policies could compensate for lack of expressivity by enabling better exploration. To focus on the final solution and further investigate similarities between converged policies, we leverage behavioral cloning to investigate the extent to which a Bernoulli policy is able to learn from a Gaussian teacher. The student acts in the environment while being supervised by a converged Gaussian. We maximize log-probability over actions and additionally

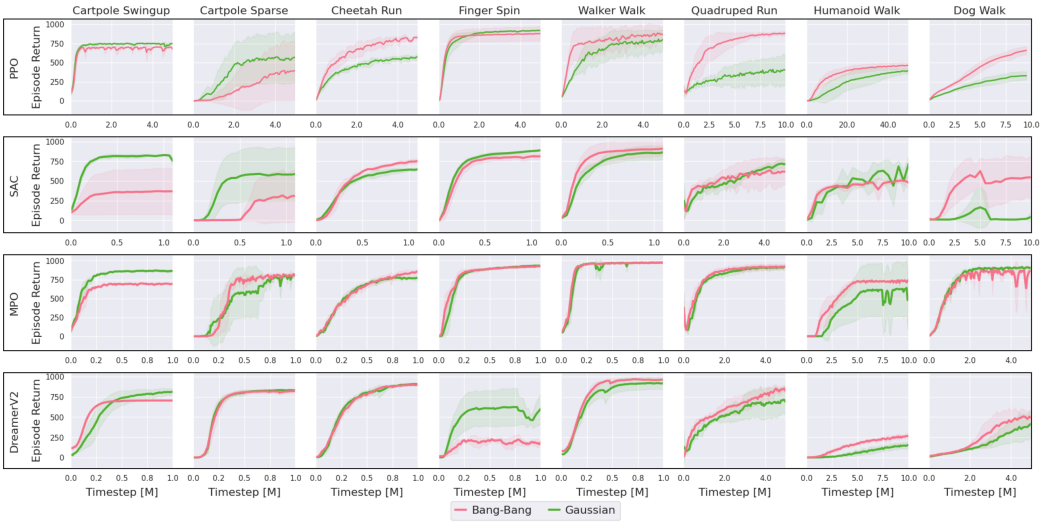

Figure 1: Comparison of Bang-Bang and Gaussian policy heads for solving continuous control tasks. We evaluate PPO, SAC, MPO, and DreamerV2 on several domains from the DeepMind Control suite. Generally, we find that the Bang-Bang policies perform on par with the normal Gaussian policies.

Figure 2: Distribution of actions along a trajectory for MPO. We consider 11 bins per action dimension and aggregate over 1000 steps. The Gaussian policy exhibits bang-bang behavior in several domains. Action penalties, particularly on Cartpole Swingup (top left), can reduce bang-bang action selection.

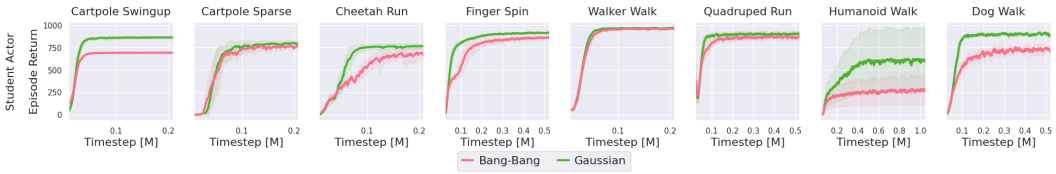

Figure 3: Behavioral cloning with a Gaussian teacher. The Bang-Bang and Gaussian students perform similarly well across several domains, further indicating that the teacher leverages bang-bang control.

discretize Gaussian targets for the Bernoulli student. Figure 3 provides learning curves for both Bernoulli and Gaussian students. Strong similarity between curves indicates strong bang-bang characteristics of the Gaussian teacher. The Bernoulli policy performs surprisingly well on all but the Humanoid task. This indicates that the Gaussian teachers consistently leverages bang-bang action selection and its differences from bang-bang behaviour have only limited impact on performance.

5.3 Robustness Under Perturbations

We observe similar converged performance of the Bernoulli and Gaussian policies. An argument against bang-bang control for articulated robotic systems from a controls perspective can be stability, particularly in the absence of ideal sensing or actuation delays that may trigger overshooting. We

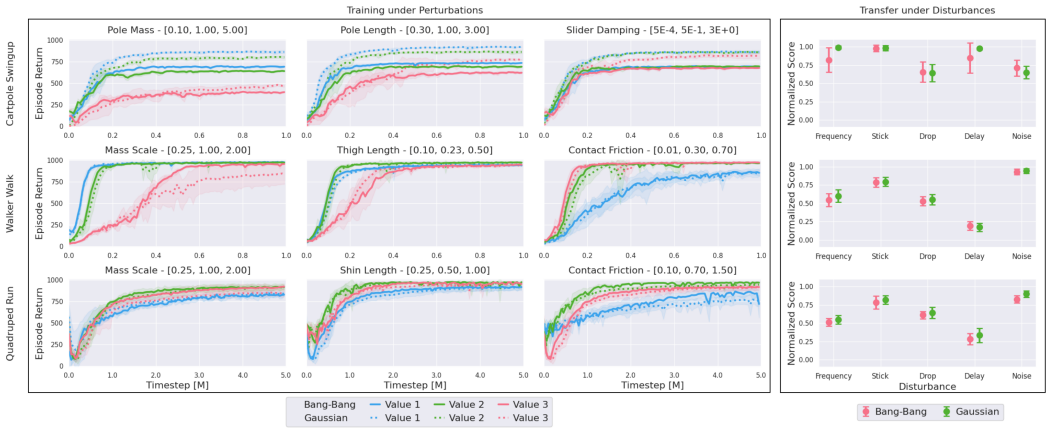

Figure 4: Robustness of the Bang-Bang policies. Left: training with perturbed environment parameters. Each subplot compares the effect of changing one parameter on training performance. Right: transfer under disturbances, performance normalized with respect to undisturbed scenario. Bang-Bang and Gaussian policies perform comparably, indicating similar robustness in simulation.

therefore investigate the impact of sensor degradation and modification of the environment parameters based on the Real-World RL Challenge Framework [13].

Environment Modifications To assess whether our observations generalize to more challenging task variations we consider modifications of the environment parameters. Specifically, we investigate performance under changes to the system mass, agent morphology, and friction or damping characteristics across the Cartpole, Walker, and Quadruped domains. Figure 4 (left) provides learning curves for MPO with Bernoulli and Gaussian policies, where parameter values are provided at the top of each subplot in order. We note that both policy types perform comparably across all tasks and parameter values, indicating that the Bang-Bang policy is robust to variations of the environments.

Transfer under Disturbances We evaluate policy transfer under real-world inspired sensor degradation. We consider reduction of the control frequency, stuck sensor signals, dropped sensor signals, sensor delay, and sensor noise (details in Appendix D). The Gaussian action space is a superset of the Bernoulli action space, yielding a richer set of interactions that could translate to better generalization around the optimal trajectory. Figure 4 (right) provides normalized scores relative to performance under ideal sensing based on 40 trajectories. We find that the Bernoulli and Gaussian policies display similar performance. This indicates that robustness to disturbances is not negatively affected by limiting controls to bang-bang within the environments considered. A potential reason for this is that the robot dynamics act as a low-pass filter on the bang-bang input to effectively smooth input jumps.

5.4 Characteristics of Bang-Bang Control in RL settings

Learning bang-bang constrained policies can yield competitive performance that generalizes across several environment formulations and algorithm designs. Next, we discuss some characteristics of bang-bang control in RL settings and evaluate methods to avoid such behavior in Gaussian policies.

Intrinsic Exploration Bang-bang policies only apply extremal actions. By forcing highest magnitude actions, the intrinsic exploration characteristics of an agent are affected. Figure 5 provides two tasks for a point-mass controlled in x-y position. Each task is evaluated with no or quadratic action penalties (left, right). Here, we consider Bang-Bang, Bang-Off-Bang, and Gaussian policies. In the top row, the agent is always initialized at the center and does not receive additional rewards. The Bang-Bang policy exhibits the largest area coverage by virtue of only sampling large magnitude actions. This effect is magnified under action cost, as other agents limit exploration to reduce cost (see also Appendix E). In the bottom row, the agent starts at the center without resets and receives sparse rewards at the circles. The Bang-Bang policy again exhibits the largest coverage. While its large magnitude sampling facilitates escapes from local optima, it can also impact stabilization at

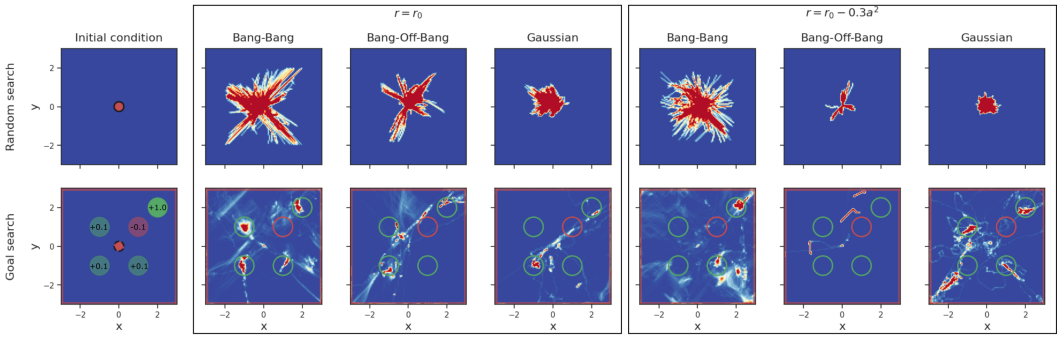

Figure 5: Pointmass exploration tasks. Top: the agent is reset every episode with no reward feedback. Bottom: the agent always continues and receives different sparse rewards. Bang-Bang control leverages extreme actions yielding strong passive exploration, unaffected by action penalties (top). This can facilitate escape from local optima and impede stabilization at global optima (bottom).

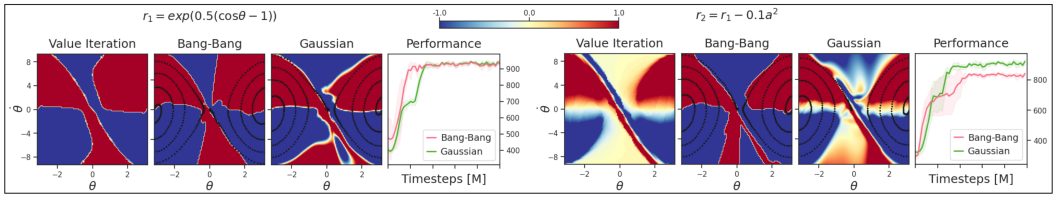

Figure 6: Learning to swing-up a pendulum under no action cost (left) and quadratic cost (right). The Bang-Bang policy converges faster without action penalties, the Gaussian yields higher returns with action penalties. Learned policy mappings are overlayed with sample swing-up trajectories (black).

global optima (Figure 5, right). Overall, Bang-Bang policies offers strong intrinsic exploration that is robust to action penalties.

Action Penalties Bang-Bang policies enable fast exploration and can arise as optimal controllers in the absence of action cost. However, robotics applications may suffer from extensive extremal action selection and action penalties can mitigate bang-bang behavior. We highlight these considerations on a pendulum swing-up task with no and quadratic action cost in Figure 6. Without action cost (left), both Bang-Bang and Gaussian policies learn to approximate the optimal action mapping and solve the task with the Bang-Bang policy offering a faster rise time. With action cost (right) which represents the minimum energy category in the optimal control setting (Section 3), the Bang-Bang policy is unable to represent the optimal action mapping and continues to incur steady-state cost (see also Appendix E).

We investigate the effect of augmenting the reward function of the more complex Walker and Quadruped tasks with action penalties. Here, we consider both quadratic cost to reduce action magnitude and difference cost to increase smoothness. In the latter case, we augment the system state by the previous action. Figure 7 (top) displays Gaussian means along a Quadruped Run trajectory for the different penalty structures and their combination. We observe the desired effect of lower magnitude and smoother transitions in the learned distributions. Figure 7 (bottom) provides training curves under the different reward structures, performance based on the nominal reward function, and transfer under disturbances (see Section 5.3) for the Walker (left) and Quadruped (right) tasks. We note that while action penalties help to mitigate bang-bang behavior, robustness of the resulting gaits is only slightly increased. Furthermore, the modified reward structure may yield reduced performance of trained agents as measured by the nominal reward function and observed on the Quadruped. These findings suggest that while potentially undesired bang-bang action selection can be reduced by introducing action penalties, naive engineering of the reward structure may force the agent into sub-optimal behaviors. The question then becomes what metrics to consider when evaluating agent performance in light of transfer to real-world systems. Pure return maximization may exploit both simulation and task peculiarities and is not sufficient for comparing the merit of different algorithms.

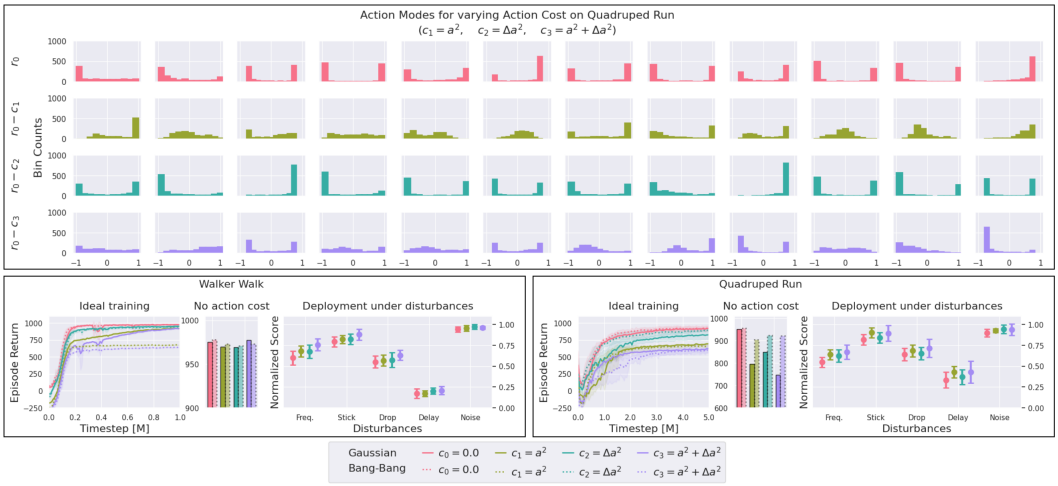

Figure 7: Impact of action penalties. Top: Gaussian means under varying penalties along a Quadruped Run trajectory. Bottom: training and transfer performance on Walker (left) and Quadruped (right). While action penalties help to mitigate bang-bang behavior, the resulting gaits only slightly increase robustness and may reduce performance as measured by the nominal reward function (Quadruped).

6 Discussion

In this paper, we investigate the emergence of bang-bang behavior in continuous control reinforcement learning and show that several common benchmarks can in fact be solved by bang-bang constrained Bernoulli policies. We draw theoretical connections to bang-bang solutions from optimal control and perform extensive experimental evaluations across a variety of state-of-the-art RL algorithms. In particular, we theoretically derive when bang-bang control emerges for continuous-time systems with deterministic dynamics and describe connections to continuous control RL problems considered here.

In our experiments, we find that many continuous control benchmarks in fact do not require continuous action spaces and can be solved (close to) optimally with learned bang-bang controllers for several recent model-based and model-free, on-policy and off-policy algorithms. We further demonstrate the efficiency of bang-bang control in more realistic variations of common domains based on the Real-World RL suite [13] and demonstrate how action costs affect the optimal controller.

Since exploration, learning and the final solution are entangled in RL, we perform additional experiments with focus on behavioural cloning of a trained Gaussian controller to compare the performance of bang-bang and continuous policies without the additional challenges arising from exploration. It is important to note that we still require sampling-based optimisation, which has become popular to handle continuous action spaces. However, the main reason is often not the continuous action space but the potential high-dimensionality of the action space which would otherwise become intractable.

Benchmark design is a complex task with the goal of measuring progress that is transferable to impactful real-world applications. We demonstrate that the learnt controllers for continuous control benchmarks without action penalties can exhibit bang-bang properties which can be damaging to real-world systems. At the same time, we also show that including action penalties can significantly impact exploration for Gaussian policies. Integrating the impact of task design on both exploration as well as final solution properties and investigating algorithms which can overcome local optima while still learning smooth controllers is an important direction for future work.

Societal Implications: It is easy to underestimate societal impacts of basic research. However, by improving understanding of emerging RL policies, this work contributes to broadening real-world applicability. On the one hand, the resulting algorithms can be used in ways not intended by the researchers such as simplifying autonomous surveillance. On the other hand, they can be used in many beneficial ways such as using robotics and automation for improving productivity and workplace safety. Even desirable outcomes can come with side-effects such as job loss resulting in societal transformation costs. While the authors ultimately believe that societal benefits of this work outweigh its harms, these considerations need to be re-evaluated on a constant basis as new applications emerge.

References

[1] A. Abdolmaleki, J. T. Springenberg, J. Degrave, S. Bohez, Y. Tassa, D. Belov, N. Heess, and M. Riedmiller. Relative entropy regularized policy iteration. *arXiv preprint arXiv:1812.02256*, 2018. 2, 5, 14

[2] C. W. Anderson. Learning to Control an Inverted Pendulum with Connectionist Networks. In *Proceedings of the American Control Conference (ACC)*, 1988. 2

[3] R. Bellman. Dynamic programming. Technical report, RAND CORP SANTA MONICA CA, 1956. 2

[4] R. Bellman, I. Glicksberg, and O. Gross. On the "bang-bang" control problem. *Quarterly of Applied Mathematics*, 14(1), 1956. 2

[5] Y. Bengio, N. Léonard, and A. Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013. 5, 14

[6] D. Bertsekas. *Dynamic Programming and Optimal Control*, volume 1. 4 edition, 2005. 3, 4

[7] S. Bohez, A. Abdolmaleki, M. Neunert, J. Buchli, N. Heess, and R. Hadsell. Value constrained model-free continuous control. *arXiv:1902.04623*, 2019. 1, 2

[8] E. Casas, D. Wachsmuth, and G. Wachsmuth. Sufficient Second-Order Conditions for Bang-Bang Control Problems. *SIAM Journal on Control and Optimization*, 55(5), 2017. 2

[9] E. Casas, D. Wachsmuth, and G. Wachsmuth. Second-Order Analysis and Numerical Approximation for Bang-Bang Bilinear Control Problems. *SIAM Journal on Control and Optimization*, 56(6), 2018. 2

[10] N. Chen, Y. Wang, and D.-H. Yang. Time-varying bang–bang property of time optimal controls for heat equation and its application. *Systems & Control Letters*, 112, 2018. 2

[11] P.-W. Chou, D. Maturana, and S. Scherer. Improving stochastic policy gradients in continuous control with deep reinforcement learning using the beta distribution. In *International conference on machine learning*, pages 834–843. PMLR, 2017. 2

[12] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel. Benchmarking Deep Reinforcement Learning for Continuous Control. In *International Conference on Machine Learning. (ICML)*, 2016. 2

[13] G. Dulac-Arnold, N. Levine, D. J. Mankowitz, J. Li, C. Paduraru, S. Gowal, and T. Hester. An empirical investigation of the challenges of real-world reinforcement learning. *arXiv preprint arXiv:2003.11881*, 2020. 7, 9

[14] G. Farquhar, L. Gustafson, Z. Lin, S. Whiteson, N. Usunier, and G. Synnaeve. Growing action spaces. In *International Conference on Machine Learning (ICML)*, 2020. 2

[15] S. Fujimoto, D. Meger, and D. Precup. Off-Policy Deep Reinforcement Learning without Exploration. In *International Conference on Machine Learning (ICML)*, 2019. 2

[16] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning (ICML)*, 2018. 2, 5, 14

[17] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, and S. Levine. Soft actor-critic algorithms and applications. Technical report, 2018. 14

[18] D. Hafner, T. Lillicrap, M. Norouzi, and J. Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020. 2, 5, 14

[19] B. J. Hodel. Learning to Operate an Excavator via Policy Optimization. *Procedia Computer Science*, 140, 2018. 2

[20] M. Hoffman, B. Shahriari, J. Aslanides, G. Barth-Maron, F. Behbahani, T. Norman, A. Abdolmaleki, A. Cassirer, F. Yang, K. Baumli, S. Henderson, A. Novikov, S. G. Colmenarejo, S. Cabi, C. Gulcehre, T. L. Paine, A. Cowie, Z. Wang, B. Piot, and N. de Freitas. Acme: A research framework for distributed reinforcement learning. *arXiv preprint arXiv:2006.00979*, 2020. 14

[21] S. H. Huang, M. Zambelli, J. Kay, M. F. Martins, Y. Tassa, P. M. Pilarski, and R. Hadsell. Learning Gentle Object Manipulation with Curiosity-Driven Deep Reinforcement Learning. *arXiv:1903.08542*, 2019. 1, 2

[22] E. Joos, F. Péan, and O. Goksel. Reinforcement Learning of Musculoskeletal Control from Functional Simulations. In *Medical Image Computing and Computer Assisted Intervention (MICCAI)*, 2020. 2

[23] D. E. Kirk. *Optimal control theory: an introduction*. Courier Corporation, 2004. 3, 4

[24] A. B. Kordabad, W. Cai, and S. Gros. MPC-based Reinforcement Learning for Economic Problems with Application to Battery Storage. In *Proceedings of the European Control Conference (ECC)*, 2021. 1

[25] K. Kunisch and L. Wang. The bang-bang property of time optimal controls for the Burgers equation. *Discrete & Continuous Dynamical Systems*, 34(9), 2014. 2

[26] J. Lambert and M. Levine. A two-stage learning control system. *Transactions on Automatic Control*, 15(3), 1970. 2

[27] J. P. LaSalle. Time Optimal Control Systems. *Proceedings of the National Academy of Sciences*, 45(4), 1959. 2

[28] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. In *International Conference on Learning Represesntations (ICLR)*, 2016. 2

[29] J. Lohéac and M. Tucsnak. Maximum Principle and Bang-Bang Property of Time Optimal Controls for Schrödinger-Type Systems. *SIAM Journal on Control and Optimization*, 51(5), 2013. 2

[30] L. Manita. Optimal operating modes with chattering switching in manipulator control problems. *Journal of Applied Mathematics and Mechanics*, 64(1), 2000. 3

[31] H. Maurer, C. Büskens, J.-H. R. Kim, and C. Y. Kaya. Optimization methods for the verification of second order sufficient conditions for bang–bang controls. *Optimal Control Applications and Methods*, 26(3), 2005. 2

[32] A. Münch and F. Periago. Numerical approximation of bang–bang controls for the heat equation: An optimal design approach. *Systems & Control Letters*, 62(8), 2013. 2

[33] R. Munos, T. Stepleton, A. Harutyunyan, and M. G. Bellemare. Safe and efficient off-policy reinforcement learning. *arXiv preprint arXiv:1606.02647*, 2016. 14

[34] G. Novati and P. Koumoutsakos. Remember and Forget for Experience Replay. In *International Conference on Machine Learning (ICML)*, 2019. 1, 2

[35] F. Pardo. Tonic: A deep reinforcement learning library for fast prototyping and benchmarking. *arXiv preprint arXiv:2011.07537*, 2020. 14

[36] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 2, 5, 14

[37] L. M. Sonneborn and F. S. Van Vleck. The Bang-Bang Principle for Linear Control Systems. *Journal of the Society for Industrial and Applied Mathematics Series A Control*, 2(2), 1964. 2

[38] B. Stellato, T. Geyer, and P. Goulart. High-speed finite control set model predictive control for power electronics. *IEEE Transactions on Power Electronics*, 32(5), 2017. 2

[39] B. Stellato, S. Ober-Blöbaum, and P. Goulart. Second-order switching time optimization for switched dynamical systems. *IEEE Transactions on Automatic Control*, 62(10), 2017. 2

[40] Y. Tang and S. Agrawal. Discretizing Continuous Action Space for On-Policy Optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020. 2

[41] Y. Tassa, S. Tunyasuvunakool, A. Muldal, Y. Doron, S. Liu, S. Bohez, J. Merel, T. Erez, T. Lillicrap, and N. Heess. dm_control: Software and tasks for continuous control, 2020. 4

[42] A. Tavakoli, F. Pardo, and P. Kormushev. Action Branching Architectures for Deep Reinforcement Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2018. 2

[43] T. G. Thuruthel, E. Falotico, F. Renda, and C. Laschi. Model-Based Reinforcement Learning for Closed-Loop Dynamic Control of Soft Robotic Manipulators. *IEEE Transactions on Robotics*, 35(1), 2019. 1, 2

[44] M. Waltz and K. Fu. A heuristic approach to reinforcement learning control systems. *IEEE Transactions on Automatic Control*, 10(4), 1965. 2

[45] G. Wang and Y. Zhang. Decompositions and bang-bang properties. *Mathematical Control & Related Fields*, 7(1), 2017. 2

[46] M. Wulfmeier, A. Abdolmaleki, R. Hafner, J. T. Springenberg, M. Neunert, N. Siegel, T. Hertweck, T. Lampe, N. Heess, and M. Riedmiller. Compositional Transfer in Hierarchical Reinforcement Learning. In *Robotics: Science and Systems (RSS)*, 2020. 2

[47] D.-H. Yang, B.-Z. Guo, W. Gui, and C. Yang. The Bang-Bang Property of Time-Varying Optimal Time Control for Null Controllable Heat Equation. *Journal of Optimization Theory and Applications*, 182(2), 2019. 2

[48] Y. Yue, Y. Tang, M. Yin, and M. Zhou. Discrete action on-policy learning with action-value critic. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2020. 2

[49] J. Zhang, K. H. Johansson, J. Lygeros, and S. Sastry. Zeno hybrid systems. *International Journal of Robust and Nonlinear Control*, 11(5), 2001. 3

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] We discussed our assumptions in the theoretical discussion (Sec. 3) and open questions with potential for further research (Sec. 6).
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] We discussed societal implications within Sec. 6 mentioning potential negative impacts.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] We discussed our assumptions in the theoretical discussion (Sec. 3)
   (b) Did you include complete proofs of all theoretical results? [N/A] Our discussion adapts existing theory from optimal control for RL and does not involve new proofs.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] Our code will be made publicly available upon publication.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Training details can be found in Appendix C.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] All our benchmarking results include standard deviation over seeds with details provided in Appendix C.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] The underlying compute resources are provided in Appendix C.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] Authors of RL libraries used here were cited in Section 5 and Appendix B.
   (b) Did you mention the license of the assets? [Yes] Licenses are mentioned in Appendix B
   (c) Did you include any new assets either in the supplemental material or as a URL? [No] Our code will be made publicly available upon publication.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] The data used in this project was generated by us.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] We did not process personal data for this project.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]