An Analysis of Safety Mechanisms for Constrained Reinforcement Learning with Gradient Boosted Regression Trees

Anonymous authors
Paper under double-blind review

Abstract

One aspect of trustworthy autonomous systems is the property of transparent decision-making, one that neural network architectures struggle with. Gradient-Boosted Regression Trees offer an alternative to neural networks as function approximators for Reinforcement Learning (RL), but their integration with safety-constrained policy optimization methods has not been studied systematically. We show that the structural properties of boosted ensembles, in particular frozen learners and function-space rather than parameter-space optimization, invalidate assumptions on which existing Safe RL algorithms rely. We identify five architectural adaptations that are necessary for correct integration: explicit critic orchestration, scale-consistent projection coefficients, independent ensemble allocation, an undiscounted violation signal, and a bounded ensemble combined with joint freezing of continuous policy parameters. We complement this analysis with a closed-form characterization of the gradient rotation induced by safety projection and establish its monotonicity in the projection strength and in the angle between reward and cost gradients. Finally, we compare gradient projection and Lagrangian relaxation across continuous control Safety-Gymnasium environments empirically.

1 Introduction

Reinforcement learning has produced policies of impressive empirical performance, but the dominant choice of function approximator, the deep neural network, carries two structural liabilities in safety-critical applications. Verification of properties of neural networks with rectified-linear activations is NP-complete (Katz et al., 2017). Independently, the decision-making of a trained neural policy is not inspectable: the function it computes is implicit in millions of weights rather than encoded in a structure a human can read. Post-hoc safety guarantees for such policies are mostly either absent or computationally intractable.

Gradient-boosted regression trees offer a different combination of properties. The function they compute is a sum of shallow decision trees, each of which maps observations to outputs through a small number of axis-aligned comparisons. Compared to deep neural networks, gradient-boosted ensembles are directly inspectable, exhibit competitive sample efficiency on tabular and low-dimensional regression benchmarks (Grinsztajn et al., 2022; Chen & Guestrin, 2016), and are deterministic at the level of tree fitting. The GBRL framework (Fuhrer et al., 2025) extends the actor-critic paradigm to represent both policy and value functions as gradient-boosted tree ensembles and has demonstrated competitive performance against neural-network policy gradient baselines on several RL benchmarks.

Despite this combination of properties, the integration of safety-constrained RL algorithms with gradient-boosted function approximators has not been studied. The two principal mechanisms used in Safe RL, the Lyapunov-based gradient projection of Chow et al. (2018) and the Lagrangian relaxation in the line of Tessler et al. (2019) and Stooke et al. (2020), are both developed and validated with neural network function approximators. Each mechanism therefore embeds architectural assumptions that hold implicitly in the neural setting but fail under the mechanics of functional gradient descent with boosted ensembles. Naive substitution of one approximator class for the other produces incorrect behavior on every Safety-Gymnasium environment we tested.

This paper makes the following contributions.

We identify five architectural requirements that are necessary for correct integration of safe reinforcement learning with gradient-boosted function approximation (Section 3). Four of the five apply equally to projection-based and Lagrangian-based safety; one is specific to projection. Each requirement is derived from a precise mismatch between the assumptions of existing safe reinforcement learning algorithms and the properties of functional gradient descent.

We give a closed-form characterization of the gradient rotation induced by the safety projection in the GBRL setting and prove its monotonicity in the projection strength and in the angle between reward and cost gradients (Section 4, Lemma 1). The lemma localizes the source of value-function non-stationarity under projection and distinguishes it from the Lagrangian, which produces no analogous rotation.

We conduct an empirical comparison of gradient projection and Lagrangian relaxation across three Safety-Gymnasium environments (Section 5). Under the five requirements, both mechanisms achieve comparable constraint satisfaction, but neither dominates the other on reward acquisition. The trade-off is environment-dependent, and the binding factor on the viability of either mechanism is the density of the reward signal, not the choice of safety mechanism.

The remainder of the paper is organized as follows. Section 2 reviews the necessary background on constrained Markov decision processes, gradient boosting, value estimation via TD(λ) and generalized advantage estimation, the GBRL framework, and the two safety mechanisms studied. Section 3 states and motivates the five architectural requirements. Section 4 develops the gradient rotation lemma. Section 5 reports the experimental setup, the cross-environment comparison, and a finer-grained analysis of how projection intensity affects value-function quality. Section 6 discusses the implications and identifies the conditions under which each constraint mechanism is preferable. Sections 7 and 8 state the limitations of the present work and summarize the contribution. Appendix A contains the proof of Lemma 1, and Appendix B provides the full experimental configuration.

2 Background

This section introduces the mathematical background for our contribution.

2.1 Constrained Markov Decision Processes

Reinforcement Learning problems are usually formalized as Markov Decision Process (MDP). An MDP $(\mathcal{S}, \mathcal{A}, p, r, \gamma)$ is a tuple that contains the state space $\mathcal{S}$, the action space $\mathcal{A}$, a transition probability function p, a reward function r and a discount factor γ. Safe RL is formalized within the framework of Constrained Markov Decision Processes (Altman, 1999). A CMDP extends a standard MDP with a cost function $c : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and a cost threshold κ. The agent seeks a policy π^* that maximizes expected discounted reward subject to the constraint that the expected discounted cumulative cost $J_C(\pi) = \mathbb{E}_{\tau\sim\pi}[\sum_{t=0}^{\infty} \gamma^t c(s_t, a_t)]$ does not exceed κ.

2.2 Gradient Boosting

Boosting originated as an ensemble paradigm in which a sequence of weak learners is aggregated into a strong predictor (Schapire, 1990; Freund & Schapire, 1997). Friedman et al. (2000) showed that AdaBoost is equivalent to stage-wise additive modeling under the exponential loss, and Mason et al. (1999) and Friedman (2001) generalized this view to arbitrary differentiable losses, which recast boosting as gradient descent in function space. Under the gradient boosting formulation, the prediction function is constructed additively as

$$F_M(x) \;=\; F_0(x) \;+\; \sum_{m=1}^{M} \eta_m \, h_m(x), \tag{1}$$

where each weak learner h_m is drawn from a base hypothesis class, most commonly the class of shallow regression trees, and is fitted to approximate the negative functional gradient $-[\partial L(y, F(x))/\partial F(x)]_{F=F_{m-1}}$ of the loss at the current model. The shrinkage parameter $\eta_m \in (0, 1]$ controls the contribution of each new

learner and provides implicit regularization against overfitting (Friedman, 2001). Two structural properties of this procedure are central to this work. First, the optimization is stage-wise and greedy: once a learner has been added to the ensemble, subsequent boosting rounds do not modify it. Second, the model resides in function space rather than parameter space, so the customary tools of parametric optimization, such as joint gradient updates, momentum, or learning rate schedules over weights, have no direct analogue. Modern implementations including XGBoost (Chen & Guestrin, 2016), LightGBM (Ke et al., 2017), and CatBoost (Prokhorenkova et al., 2018) augment this framework with regularized objectives and scalable split finding.

2.3 Value Estimation with TD(λ) and GAE

The actor-critic algorithms studied in this paper depend on estimates of the state-value function $V^\pi(s) = \mathbb{E}_\pi[\sum_{t\geq 0}\gamma^t r_t \mid s_0 = s]$ and on advantages derived from V^π. Monte Carlo estimation replaces $V^\pi(s)$ with the sample return obtained from a completed rollout: unbiased but with variance that grows with the trajectory horizon. Temporal-difference learning, introduced by Sutton (1988), bootstraps from a learned value function: the one-step target $r_t + \gamma V(s_{t+1})$ replaces a long stochastic sum with a single reward and a single bootstrap term, which reduces variance sharply but inherits the bias of V.

The two extremes are unified by the n-step return, $G_t^{(n)} = \sum_{l=0}^{n-1}\gamma^l r_{t+l} + \gamma^n V(s_{t+n})$, which recovers the one-step TD target at $n = 1$ and the Monte Carlo return as $n \to \infty$. TD(λ) (Sutton, 1988; Sutton & Barto, 2018) generalizes this construction through an exponentially weighted average over all n:

$$G_t^\lambda \;=\; (1-\lambda)\sum_{n=1}^{\infty}\lambda^{n-1}\,G_t^{(n)}. \tag{2}$$

The forward view applied to advantage residuals rather than to value targets is the Generalized Advantage Estimator (GAE) of Schulman et al. (2016):

$$\hat{A}_t^{\mathrm{GAE}(\gamma,\lambda)} \;=\; \sum_{l=0}^{\infty}(\gamma\lambda)^l\,\delta_{t+l}, \qquad \delta_t = r_t + \gamma V(s_{t+1}) - V(s_t). \tag{3}$$

This is consequential in the following ways: First, the quality of the bootstrap term V determines the operating regime: when V is reliable, a low λ produces low-variance estimates; when V is unreliable, λ must approach 1 and the estimator behaves more like Monte Carlo with the accompanying variance cost. The explained variance metric reported throughout Section 5 measures this reliability of V. Second, when the value function fails to learn at all, every bootstrap term is zero and the advantages reduce to a discounted empirical sum of rewards, a Monte Carlo estimate without baseline subtraction.

2.4 Gradient-Boosted Reinforcement Learning

The GBRL framework (Fuhrer et al., 2025) represents both policy and value functions as gradient-boosted regression tree ensembles and updates each ensemble through stage-wise functional gradient descent on the appropriate loss. At each training iteration, the framework computes pseudo-residuals that act as gradient targets at every collected sample, fits a new tree to approximate those targets, and appends the tree to the ensemble.

Trees once added are permanently part of the ensemble, and each new tree is fitted to the gradient residual of all previous trees. A further structural property is that each new tree contributes a fraction $1/N$ of the total ensemble output once N trees have accumulated, which produces an implicit learning rate decay of order $\mathcal{O}(1/N)$. These properties are benign in standard RL but become problematic under the dual-objective structure of Safe RL, as Section 3 details.

2.5 Safety Mechanisms: Projection and Lagrangian relaxation

Gradient projection. Chow et al. (2018) propose a Lyapunov-based safety approach that projects the policy gradient into the safe set defined by state-dependent linearized constraints. The projection coefficient

is

$$\lambda^* = \max\left(0,\ \frac{\langle \mathbf{g},\, \mathbf{b}\rangle + e_k}{\|\mathbf{b}\|^2}\right), \tag{4}$$

where $\mathbf{g} = \nabla J_R$ is the reward gradient, $\mathbf{b} = \nabla J_C$ is the cost gradient, and $e_k = J_C(\pi_k) - \kappa$ is the current constraint violation. The projected gradient is $\mathbf{g}_{\text{proj}} = \mathbf{g} - \lambda^* \mathbf{b}$. In the GBRL setting, the projection operates on per-sample functional gradient targets rather than on parameter-space gradients: each fitted tree encodes a safety-corrected direction at the sample level.

Lagrangian relaxation. An alternative is Lagrangian relaxation (Stooke et al., 2020; Tessler et al., 2019; Ray et al., 2019), which replaces the hard constraint with a soft penalty whose weight is learned by gradient ascent. The combined objective is

$$J_{\text{Lag}}(\pi, \lambda) \ = \ J_R(\pi) - \lambda\big(J_C(\pi) - \kappa\big), \tag{5}$$

with the scalar dual variable $\lambda \geq 0$ updated by $\lambda \leftarrow [\lambda + \eta_\lambda (J_C(\pi) - \kappa)]_+$. In the GBRL setting the per-sample advantage $A_i^{\text{Lag}} = A_i^R - \lambda \cdot A_i^C$ is supplied as the gradient target for tree fitting. The dual variable acts as a scalar weighting on the cost advantage and produces no rotation of the gradient direction in function space.

The two mechanisms differ structurally. The projection enforces a per-update constraint and computes λ^* from current quantities, so a fresh correction applies at every training step. The Lagrangian enforces a trajectory-level constraint asymptotically, with λ accumulated across updates. These differences have implications for the GBRL setting that are quantified in Section 5.

3 Architectural Requirements for Safe GBRL

Safe RL algorithms, such as Proximal Policy Optimization (PPO) Schulman et al. (2017), are developed and validated with neural network function approximators, which embeds architectural assumptions that hold implicitly in that setting but fail under functional gradient descent with boosted ensembles. This section identifies five such assumptions and derives the corresponding requirements. Requirements 1, 3, 4, and 5 apply to both projection-based and Lagrangian-based safe GBRL; Requirement 2 is specific to projection.

3.1 Explicit critic orchestration

In neural-network actor-critic algorithms, a shared optimizer or computational graph ensures that both actor and critic receive gradient updates at every training step. In GBRL, actor and critic are independent tree ensembles. This separation eliminates implicit gradient coupling: if the critic boosting step is not explicitly issued, the value ensemble remains at its zero initialization, the explained variance is identically zero, and as observed in Section 2.3 the advantages reduce to raw Monte Carlo returns.

Requirement 1 (Explicit critic orchestration)**.** Each training iteration must include an explicit boosting step for every value ensemble (reward and cost) in addition to the actor boosting step.

3.2 Scale-consistent projection

The projection coefficient (4) requires that e_k and $\langle \mathbf{g}, \mathbf{b}\rangle$ be expressed in commensurable units. In neural network parameterizations, both quantities derive from the same parameter space and this commensurability holds implicitly. In GBRL, the inner product is computed over normalized per-sample advantage gradients, whereas e_k is an accumulated cost return. The units are incommensurable because tree leaf values are decoupled from the advantage function. Dimensional consistency is restored when e_k is rescaled by the root-mean-square norm of the cost gradient, $b_{\text{rms}} = \|\mathbf{b}\|_2/\sqrt{N}$:

$$\lambda^* = \max\left(0,\ \frac{\langle \mathbf{g},\, \mathbf{b}\rangle + e_k \cdot b_{\text{rms}}}{\|\mathbf{b}\|^2}\right). \tag{6}$$

Requirement 2 (Scale-consistent projection)**.** When the policy parameterization induces a scale separation between the constraint violation term and the advantage gradients, the projection coefficient must include an explicit rescaling factor.

3.3 Independent ensemble allocation

When actor and critic share a single tree ensemble with a single learning rate, the 10^2 to 10^3 times larger policy gradients dominate every tree-fitting step and leave the value function with negligible updates. When reward returns are normalized but cost returns are not, the safety projection (or the Lagrangian advantage) operates on gradients of incompatible magnitude and over-corrects systematically.

Requirement 3 (Independent ensembles and symmetric normalization)**.** Each objective must be assigned an independent tree ensemble with an independent learning rate, and any normalization applied to one return stream must be applied symmetrically to the other.

3.4 Undiscounted violation signal

The cost threshold κ is conventionally defined in terms of undiscounted episode cost. Both the projection and the Lagrangian, however, naturally compute the violation e_k from the discounted cost return $\sum_t \gamma^t c_t$.

The practical consequence is that the constraint mechanism fires less frequently than intended or not at all. In our experiments on SafetyPointCircle1 with $\kappa = 50$, the projection activation rate was 0 % despite undiscounted episode costs of 99 ± 22, because the discounted returns remained below the threshold. Replacing the discounted violation signal with the undiscounted episode cost mean restored projection activation to 80 % and reduced episode cost by 26 %. The same correction is required for the Lagrangian dual update, where the dual variable otherwise saturates near zero.

Requirement 4 (Undiscounted violation signal)**.** The violation term e_k in the projection coefficient and in the Lagrangian dual update must be computed from the undiscounted episode cost rather than from discounted cost returns, to ensure consistency with the threshold κ.

3.5 Bounded ensemble capacity and joint parameter freezing

Under safety constraints, the optimization target alternates between reward-seeking and cost-minimizing regimes. Since trees fitted during one regime cannot be removed or downweighted, the ensemble accumulates contributions from both regimes and produces a progressive drift in the output statistics. The following proposition formalizes this drift.

Proposition 1 (Worst-case ensemble drift)**.** *Let $F_M(x) = F_0(x) + \sum_{m=1}^{M} \eta_m h_m(x)$ denote the ensemble output after M boosting rounds, with shrinkage $\eta_m \leq \eta$ and leaf values uniformly bounded by $\|h_m\|_\infty \leq B$. For any $M_0 < M$,*

$$\sup_x |F_M(x) - F_{M_0}(x)| \;\leq\; \eta\,(M - M_0)\,B. \tag{7}$$

Proof. By the triangle inequality, $|F_M(x) - F_{M_0}(x)| \leq \sum_{m=M_0+1}^{M} \eta_m |h_m(x)| \leq \eta(M - M_0)B$, uniformly in x. □

Equation (7) is conservative under stationary optimization, where trees from successive rounds tend to oppose one another and observed drift is sublinear in $M - M_0$. The bound becomes informative under regime change: trees fitted in the new regime no longer cancel earlier contributions, and the drift approaches the linear worst case. A naive translation of (7) into a tree cap yields $N_{\max} \leq (L - |F_0|)/(\eta B)$, where L is the output range within which the Gaussian policy is well-defined. With representative values $\eta = 0.1$, $B = 3$, and $L = 1$ this evaluates to $N_{\max} \approx 3$, which is uninformative. A practical calibration ties $N_{\max}$ to the training budget through the per-iteration tree-addition rate ρ, so that the cap activates at the end of training rather than during it: $N_{\max} = \rho T$. With the configuration used in Section 5, the cap value $N_{\max} = 2.5 \times 10^4$ corresponds to approximately 1.28×10^6 environment steps. The experiments reported here use a 10^6-step budget and therefore do not exhaust $N_{\max}$; the divergence described below was observed in extended development runs at 1.5×10^6 steps and constitutes the empirical justification for retaining the cap.

A second structural problem arises once the cap activates. The Gaussian policy's log-standard-deviation $\log \sigma$ is typically updated by an independent optimizer through the policy gradient and entropy terms of the PPO objective. Both contributions depend on the actor output, and once the ensemble is frozen those outputs no

longer respond to changes in $\log \sigma$. The feedback that regulates the standard deviation is broken, and $\log \sigma$ drifts under residual gradient noise. In our extended-budget experiments the action standard deviation grew from 0.3 to 109 within 2×10^5 steps of cap activation, which effectively randomized the policy.

Requirement 5 (Bounded ensemble with joint freezing)**.** The ensemble size must be bounded at $N_{\max}$, and when the bound is reached, all policy parameters, including continuous parameters such as the Gaussian log-standard-deviation, must be frozen jointly.

4 Theoretical Analysis of Gradient Rotation under Projection

The safety projection rotates the effective gradient direction away from the pure reward gradient. The following lemma characterizes this rotation.

Lemma 1 (Gradient rotation)**.** *Let* $\mathbf{g} = \nabla_F J_R$ *and* $\mathbf{b} = \nabla_F J_C$ *with* $\varphi = \angle(\mathbf{g}, \mathbf{b}) \in (0, \pi/2)$*. Under safety projection with coefficient* $\lambda^* > 0$*, the rotation angle* $\alpha = \angle(\mathbf{g}, \mathbf{g}_{\text{proj}})$ *satisfies*

$$\cos(\alpha) = \frac{1 - \rho \cos \varphi}{\sqrt{1 - 2\rho \cos \varphi + \rho^2}}\,, \tag{8}$$

where $\rho = \lambda^* \|\mathbf{b}\| / \|\mathbf{g}\|$ *is the dimensionless projection strength. Moreover, in the operating regime* $\cos \varphi > \rho$*:* (i) α *is monotonically increasing in* φ*;* (ii) α *is monotonically increasing in* ρ*;* (iii) $\alpha = \pi/2$ *if and only if* $\rho \cos \varphi = 1$*.*

The proof is given in Appendix A. Two implications of the lemma are relevant for the present work.

The lemma provides a closed-form relationship between projection strength, objective geometry, and the magnitude of the gradient rotation. In environments where the projection fires frequently with high ρ, the effective gradient direction changes substantially at each projection event, which creates a non-stationary optimization target for the value function. Frozen GBRL trees cannot adapt to such shifts, and value function quality degrades with increasing projection intensity. Our experiments in Section 5 report values of ρ between 12 and 32 across environments, confirming that the non-stationary regime is the operating point of practical GBRL training under projection.

The Lagrangian relaxation incurs no analogous rotation. The combined objective (5) defines a single fixed convex combination of reward and cost objectives at any moment, parameterized by the scalar dual variable. The advantage target $A^R - \lambda A^C$ alters only the magnitude and sign of the combined gradient, not its direction relative to the cost gradient at the current update. The dual variable evolves slowly under gradient ascent and introduces non-stationarity at a much lower frequency than per-step projection. Figure 1 visualizes the consequence: under a rotating target gradient, a neural network re-aligns at every step, while a GBRL ensemble accumulates frozen contributions whose sum lags the current target by an angle that grows with each successive snapshot.

5 Experiments

This section presents the results of our empirical evaluation.

5.1 Setup

All experiments use the GBRL framework on top of Stable-Baselines3 with the following configuration: 10^6 total environment steps; $n_{\text{steps}} = 512$; $n_{\text{envs}} = 16$; batch_size = 512; $n_{\text{epochs}} = 5$; $\gamma = 0.99$; GAE $\lambda = 0.98$; clip_range = 0.2; ent_coef = 0.015; grow_policy = oblivious; $N_{\max} = 25{,}000$. For the projection, $\lambda^* \leq 1.0$ is enforced as a hard cap. For the Lagrangian, the dual variable is capped at $\lambda \leq 2.0$ with learning rate $\eta_\lambda = 5 \times 10^{-3}$. All architectural requirements from Section 3 are applied to both methods. Each configuration is evaluated over 10 random seeds with PPO as the base algorithm.

Three Safety-Gymnasium (Ji et al., 2023) environments are used, each with cost threshold $\kappa = 25$. *SafetySwimmerVelocity* requires a Swimmer agent to maximize forward velocity subject to a velocity limit. The

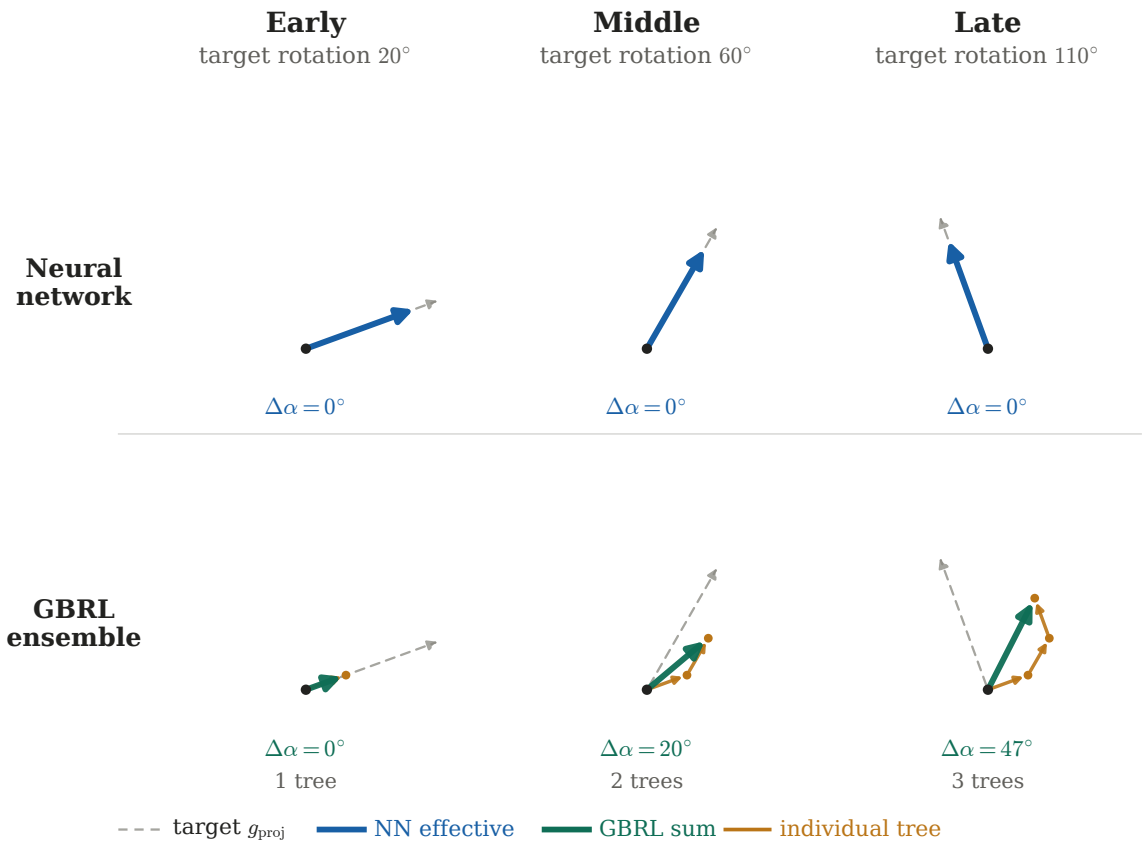

Figure 1: Effect of gradient rotation on neural network and GBRL function approximation at three training snapshots. The dashed grey arrow is the current projected gradient g_{proj}. Top row: a neural network re-aligns its parameter vector with the current target at every step ($\Delta\alpha = 0°$). Bottom row: GBRL adds one tree per snapshot (amber, head-to-tail) and cannot remove earlier trees. The sum (teal) lags the rotating target by 0°, 20°, and 47° respectively.

reward is dense (proportional to forward velocity) and the constraint is a threshold on instantaneous velocity. *SafetyPointCircle1* requires a Point agent to maintain forward velocity along a circular path while it remains within a boundary. The reward is dense (proportional to tangential velocity) and the constraint penalizes boundary crossings. *SafetyPointGoal1* requires a Point agent to navigate to randomly placed goals while it avoids hazards. The reward is sparse (granted upon goal arrival) and the constraint penalizes hazard contact; SafetyPointGoal1 serves as a diagnostic case for the sparse-reward limitation.

5.2 Cross-environment results

Table 1 reports the principal metrics for both constraint mechanisms across all three environments. Figure 2 shows the corresponding training trajectories and Figure 3 summarises final performance. Three findings emerge.

(i) Both mechanisms produce comparable safety outcomes. Final episode costs are similar within standard deviation across all three environments: 13.0 vs 11.2 on SwimmerVelocity, 17.6 vs 11.9 on PointCircle1, and 2.9 vs 1.8 on PointGoal1. The fraction of evaluation periods below κ is also similar (22 vs 23 %, 19 vs 16 %, 79 vs 85 %). The Lagrangian achieves slightly lower final cost on every environment, yet the projection's stronger per-step correction does not translate into substantially better constraint satisfaction.

Table 1: Projection versus Lagrangian relaxation across three Safety-Gymnasium environments (10 seeds, $\kappa = 25$, 10^6 environment steps). EV is the explained variance of the reward value function. Final values are the last logged data point per seed; means are computed over the entire training run.

	SwimmerVelocity		PointCircle1		PointGoal1	
Metric	Proj.	Lag.	Proj.	Lag.	Proj.	Lag.
EV (mean)	0.68 ± 0.11	0.73 ± 0.06	0.68 ± 0.06	0.66 ± 0.07	0.35 ± 0.08	0.34 ± 0.09
EV (final)	0.65 ± 0.19	0.70 ± 0.12	0.67 ± 0.08	0.60 ± 0.12	0.11 ± 0.14	0.01 ± 0.34
Cost (mean)	63.4 ± 16.7	64.5 ± 14.6	72.2 ± 28.3	85.5 ± 29.8	16.2 ± 4.4	12.9 ± 3.5
Cost (final)	13.0 ± 3.8	11.2 ± 2.7	17.6 ± 6.0	11.9 ± 8.2	2.9 ± 1.2	1.8 ± 1.8
Reward (mean)	0.58 ± 2.65	1.26 ± 3.82	1.03 ± 1.35	0.09 ± 0.88	-0.23 ± 0.23	-0.21 ± 0.20
Reward (peak)	19.5 ± 4.3	18.5 ± 11.9	5.4 ± 2.9	5.6 ± 4.4	0.5 ± 0.3	0.5 ± 0.5
Periods $< \kappa$	22 %	23 %	19 %	16 %	79 %	85 %
Std (final)	0.39 ± 0.12	0.20 ± 0.03	0.45 ± 0.24	0.20 ± 0.01	1.14 ± 0.18	0.97 ± 0.36
Proj. activation	79 %	n/a	80 %	n/a	23 %	n/a
Proj. ρ (mean)	29.2 ± 4.7	n/a	32.4 ± 11.1	n/a	12.2 ± 2.6	n/a
Lag. λ final	n/a	1.88 ± 0.19	n/a	1.96 ± 0.05	n/a	0.00 ± 0.00

(ii) The two mechanisms differ in how they preserve reward learning. The differences are environment-dependent and do not favour either mechanism across the board. On SwimmerVelocity, the Lagrangian achieves higher mean reward (1.26 vs 0.58) and indistinguishable peak reward. On PointCircle1, the projection achieves higher mean reward (1.03 vs 0.09), although peak rewards are comparable. On PointGoal1, both mechanisms produce mean reward near zero. The Lagrangian dual saturates near its cap of 2 on the dense-reward environments ($\lambda = 1.88$ on SwimmerVelocity, $\lambda = 1.96$ on PointCircle1) and remains at zero on PointGoal1, where the cost stays below threshold without any dual pressure. The saturation (Figure 4) indicates that the trajectory- level constraint is binding on SwimmerVelocity and PointCircle1 and that the algorithm pushes λ to its cap to enforce satisfaction.

(iii) Return signal density is the dominant factor for value function viability. Explained variance on the dense-reward environments (0.66 to 0.73) is consistently high under both methods, whereas PointGoal1 (0.34 to 0.35) degrades throughout training. The relationship holds independent of the constraint mechanism: return variance, not safety method, determines whether the GBRL value function can support meaningful policy improvement. The return variance is $\mathrm{Var}(R) \approx 1$ to 4 on the dense- reward environments versus ≈ 0.06 on PointGoal1, a difference of nearly two orders of magnitude.

5.3 Projection intensity and value function quality

Table 2 summarizes the relationship between projection intensity and value function quality. Environments with higher projection activation rates and larger ρ exhibit lower explained variance than the baseline determined by reward density alone, consistent with Lemma 1: stronger projection produces larger gradient rotations and greater non-stationarity, which frozen GBRL trees cannot track.

Table 2: Relationship between projection intensity and value function quality. The first three rows show projection runs at $\kappa = 25$; the last row shows a PointCircle1 baseline in which the projection is inactive throughout training (achieved by use of the discounted violation signal, which never exceeds $\kappa = 50$).

	Projection rate	ρ	EV (mean)
PointGoal1 (sparse reward)	23 %	12.2	0.35
SwimmerVelocity	79 %	29.2	0.68
PointCircle1	80 %	32.4	0.68
PointCircle1 (no proj.)	0 %	0	0.78

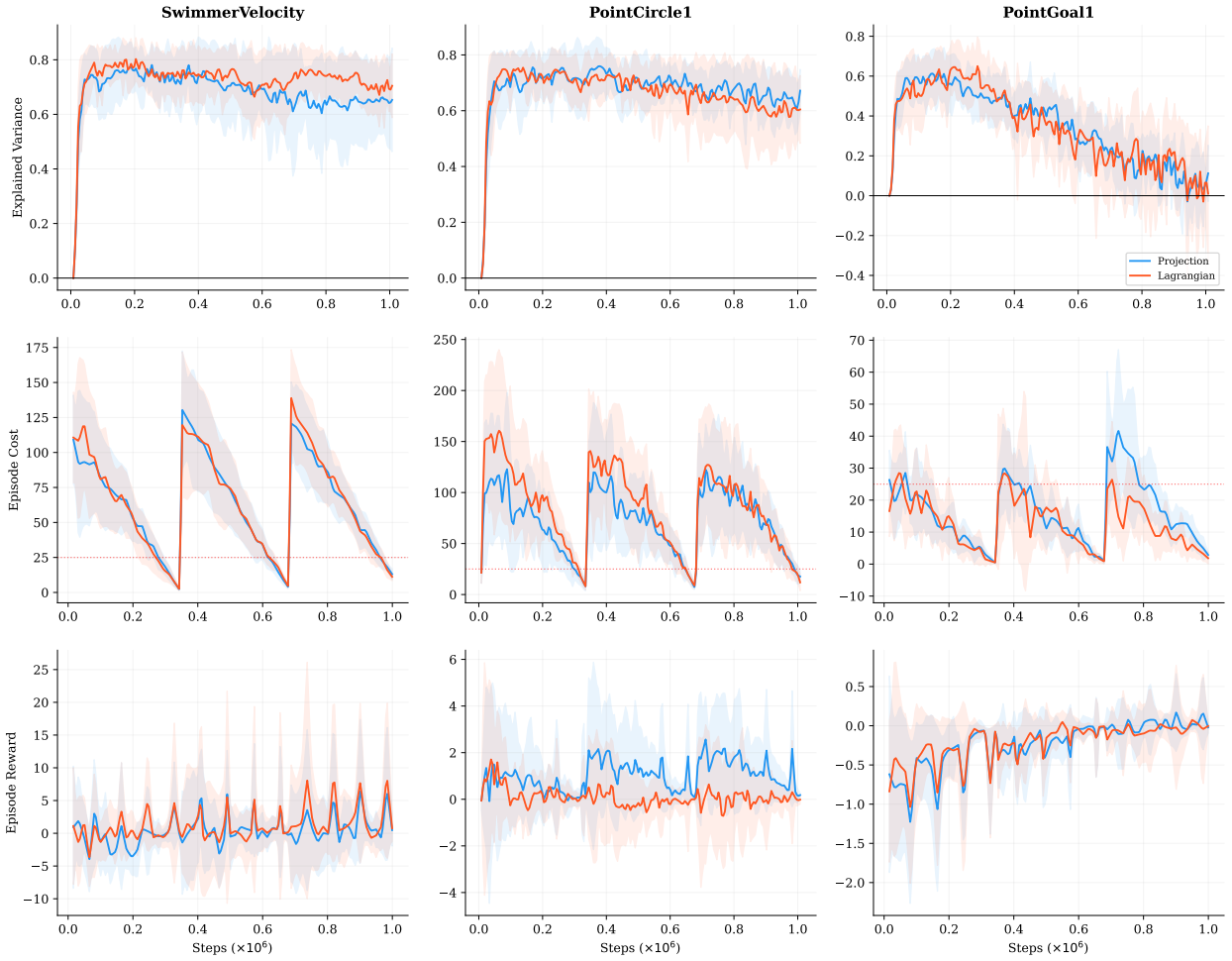

Figure 2: Training curves for projection (orange) and Lagrangian (blue) across three environments. Rows show reward, episode cost, and explained variance. Shaded bands denote ±1 standard deviation over 10 seeds. The dashed horizontal line indicates the cost threshold $\kappa = 25$.

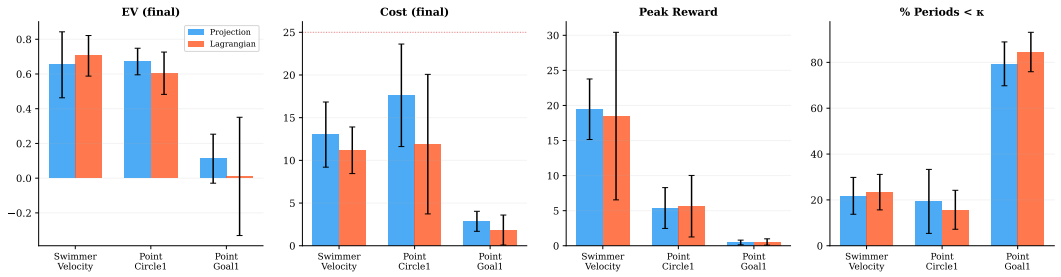

Figure 3: Final-performance summary (10 seeds, ±1 std). Both mechanisms reduce final episode cost below the threshold on SwimmerVelocity and achieve very low cost on PointGoal1; PointCircle1 hovers near κ under both mechanisms, with the Lagrangian achieving the lower final cost.

The PointCircle1 comparison with and without projection (0.68 vs 0.78 EV) provides the cleanest estimate of the projection's impact on value function quality, since all other variables are held constant. The 13 % reduction in EV represents the cost of safety enforcement under the projection mechanism. PointGoal1 has the lowest projection rate (23 %) and the lowest ρ (12.2) yet also the lowest EV (0.35), which breaks the

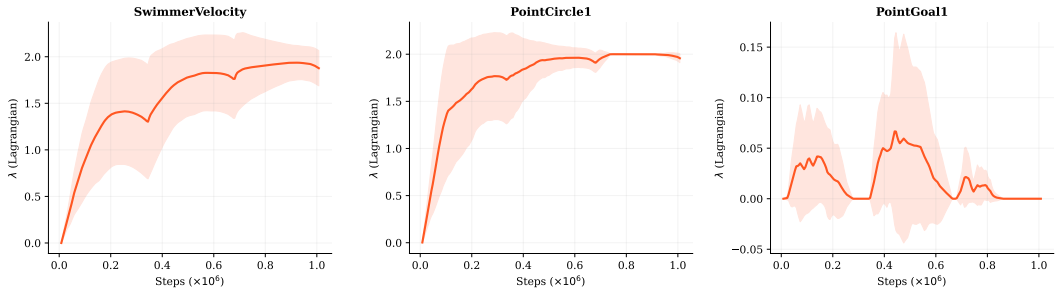

Figure 4: Lagrangian dual variable λ over training (10 seeds, ± 1 std). On SwimmerVelocity and PointCircle1, λ rises rapidly toward its cap of 2.0 and then plateaus. On PointGoal1, λ remains at zero because the undiscounted episode cost stays below $\kappa = 25$ from the outset.

monotonic relationship observed across the dense-reward environments. This confirms that EV is determined primarily by return signal density rather than projection intensity alone: the projection imposes an additional EV cost on top of the baseline determined by reward structure.

6 Discussion

When to prefer each mechanism. The projection enforces a per-step constraint and is therefore appropriate when transient violations are unacceptable and immediate correction is required. The Lagrangian enforces a trajectory- level constraint asymptotically and is preferable when a longer adjustment horizon is admissible and a smoother optimization landscape is desired. In the GBRL setting, the trade-off has an additional dimension: the projection induces a per-update gradient rotation that creates non-stationarity in the value function target, whereas the Lagrangian preserves gradient direction and only rescales the linear combination of objectives. Table 2 translates this into a quantitative cost: roughly a 13 % reduction in explained variance attributable to projection alone on PointCircle1. The Lagrangian avoids this cost by construction, although it incurs different costs when the dual variable saturates and effectively re- weights the objective into a near-pure cost minimisation problem, as observed on PointCircle1, where the Lagrangian reduces final cost to 11.9 but suppresses mean reward to 0.09.

Return signal density as a prerequisite. The experimental results indicate that the density of the return signal is the primary determinant of whether GBRL-based safe RL produces meaningful policies. On environments with dense rewards, the value function achieves and maintains explained variance of 0.66 to 0.73 under both constraint mechanisms, which provides reliable advantage estimates for the policy gradient. On PointGoal1 with its sparse reward structure, the return variance is two orders of magnitude lower and the value function degrades continuously. Either constraint mechanism still enforces constraints in this regime, but the resulting policy is trivially conservative rather than skilfully safe. Practitioners considering GBRL for safety-critical applications should verify that the reward structure provides sufficient return variance for the value function to learn from. Dense reward shaping, where available, significantly improves the viability of the approach.

The safety-reward trade-off in frozen ensembles. The asymmetry between neural network and GBRL representations is the core structural finding of this work. In neural network implementations, the cost of safety enforcement is mitigated by backpropagation through all parameters, which lets the value function adapt to the changed return distribution. In GBRL, trees fitted before a projection event are frozen and cannot adapt; stale predictions accumulate and degrade explained variance. The Lagrangian relaxation incurs no analogous explained-variance penalty across the dense-reward environments (0.73 vs 0.68 on SwimmerVelocity, 0.66 vs 0.68 on PointCircle1). Once the dual variable saturates, however, the Lagrangian effectively re-weights the objective into a near-pure cost minimisation problem, which produces tight constraint satisfaction at the cost of reward suppression. The two mechanisms expose two facets of the safety-reward trade-off: the

projection trades explained variance for per-step safety, while the Lagrangian trades reward for asymptotic constraint satisfaction once the dual saturates.

7 Limitations

The projection coefficient ρ observed in our experiments indicates that the safety correction is an order of magnitude larger than the reward gradient. At these values, the projected gradient is dominated by the cost objective, which leaves limited capacity for reward-directed learning. Methods that reduce ρ while constraint satisfaction is maintained are an important direction for future work but are not pursued here.

The Lagrangian dual variable saturates at its cap on two of three environments, indicating that the cap is binding and that a higher cap or a different dual update rule (such as PID Lagrangians (Stooke et al., 2020)) might shift the trade-off. Adaptive cap selection is not pursued.

All experiments use a single set of hyper-parameters across environments. This choice ensures a fair cross-environment comparison; per-environment tuning would likely improve absolute performance.

8 Conclusion

This paper has established conditions under which gradient-boosted tree ensembles can serve as function approximators for safety-constrained reinforcement learning. Five architectural requirements were identified, each addressing a mismatch between the assumptions of neural-network safe RL and the mechanics of functional gradient descent. Under these requirements, a controlled comparison of gradient projection and Lagrangian relaxation across three Safety-Gymnasium environments yields three practical conclusions. First, both mechanisms achieve comparable constraint satisfaction on the environments tested. Second, neither mechanism dominates on reward acquisition: the trade-offs are environment-dependent and the choice should be informed by application-specific priorities. Third, the central practical finding is that return signal density determines the viability of the approach: dense-reward environments support stable explained variance of 0.66 to 0.73 and effective constraint enforcement under either mechanism, whereas sparse-reward environments produce degrading value functions and trivially conservative policies regardless of the constraint mechanism applied. Together, these findings provide a principled basis for deciding when GBRL-based Safe RL is appropriate and for anticipating the trade-offs it entails.

References

Eitan Altman. *Constrained Markov Decision Processes*. Chapman and Hall/CRC, Boca Raton, FL, 1999.

Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, 2016.

Yinlam Chow, Ofir Nachum, Edgar A. Duéñez-Guzmán, and Mohammad Ghavamzadeh. A Lyapunov-based approach to safe reinforcement learning. In *Advances in Neural Information Processing Systems 31 (NeurIPS)*, pp. 8103–8112, 2018.

Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: A statistical view of boosting. *The Annals of Statistics*, 28(2):337–407, 2000.

Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1232, 2001.

Benjamin Fuhrer, Chen Tessler, and Gal Dalal. Gradient boosting reinforcement learning. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, pp. 17960–17985. PMLR, 2025.

Léo Grinsztajn, Edouard Oyallon, and Gaël Varoquaux. Why do tree-based models still outperform deep learning on typical tabular data? In *Advances in Neural Information Processing Systems 35 (NeurIPS), Datasets and Benchmarks Track*, 2022.

Jiaming Ji, Borong Zhang, Jiayi Zhou, Xuehai Pan, Weidong Huang, Ruiyang Sun, Yiran Geng, Yifan Zhong, Josef Dai, and Yaodong Yang. Safety gymnasium: A unified safe reinforcement learning benchmark. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL `https://openreview.net/forum?id=WZmlxIuIGR`.

Guy Katz, Clark Barrett, David L. Dill, Kyle Julian, and Mykel J. Kochenderfer. Reluplex: An efficient SMT solver for verifying deep neural networks. In *Computer Aided Verification (CAV)*, volume 10426 of *Lecture Notes in Computer Science*, pp. 97–117. Springer, 2017.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. LightGBM: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems 30 (NeurIPS)*, pp. 3146–3154, 2017.

Llew Mason, Jonathan Baxter, Peter L. Bartlett, and Marcus R. Frean. Boosting algorithms as gradient descent. In *Advances in Neural Information Processing Systems 12 (NeurIPS)*, pp. 512–518, 1999.

Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. CatBoost: Unbiased boosting with categorical features. In *Advances in Neural Information Processing Systems 31 (NeurIPS)*, pp. 6638–6648, 2018.

Alex Ray, Joshua Achiam, and Dario Amodei. Benchmarking safe exploration in deep reinforcement learning. Technical report, OpenAI, 2019. URL `https://cdn.openai.com/safexp-short.pdf`.

Robert E. Schapire. The strength of weak learnability. *Machine Learning*, 5(2):197–227, 1990.

John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations (ICLR)*, 2016.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Adam Stooke, Joshua Achiam, and Pieter Abbeel. Responsive safety in reinforcement learning by PID Lagrangian methods. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pp. 9133–9143. PMLR, 2020.

Richard S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3(1):9–44, 1988.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 2nd edition, 2018.

Chen Tessler, Daniel J. Mankowitz, and Shie Mannor. Reward constrained policy optimization. In *International Conference on Learning Representations (ICLR)*, 2019.

A Proof of Lemma 1

Proof. Let $\mathbf{g}_{\text{proj}} = \mathbf{g} - \lambda^* \mathbf{b}$ and write

$$\cos(\alpha) = \frac{\langle \mathbf{g}, \mathbf{g}_{\text{proj}} \rangle}{\|\mathbf{g}\| \, \|\mathbf{g}_{\text{proj}}\|}.$$

The numerator expands as $\langle \mathbf{g}, \mathbf{g} - \lambda^* \mathbf{b} \rangle = \|\mathbf{g}\|^2 - \lambda^* \langle \mathbf{g}, \mathbf{b} \rangle = \|\mathbf{g}\|^2 (1 - \rho \cos\varphi)$, using $\rho = \lambda^* \|\mathbf{b}\| / \|\mathbf{g}\|$ and $\langle \mathbf{g}, \mathbf{b} \rangle = \|\mathbf{g}\| \, \|\mathbf{b}\| \cos\varphi$. For the denominator,

$$\begin{aligned} \|\mathbf{g}_{\text{proj}}\|^2 &= \|\mathbf{g}\|^2 - 2\lambda^* \langle \mathbf{g}, \mathbf{b} \rangle + (\lambda^*)^2 \|\mathbf{b}\|^2 \\ &= \|\mathbf{g}\|^2 (1 - 2\rho \cos\varphi + \rho^2). \end{aligned}$$

Substituting, $\cos(\alpha) = (1 - \rho\cos\varphi)/\sqrt{1 - 2\rho\cos\varphi + \rho^2}$, establishing (8).

For (i), let $u = \cos\varphi$ and differentiate $\cos(\alpha) = N(u)/D(u)$ with $N(u) = 1 - \rho u$ and $D(u) = \sqrt{1 - 2\rho u + \rho^2}$. The quotient rule yields

$$\frac{d\cos(\alpha)}{du} = \frac{N'(u)D(u) - N(u)D'(u)}{D(u)^2} = \frac{-\rho\, D(u)^2 + \rho(1 - \rho u)^2}{D(u)^3} = \frac{\rho^2(u - \rho)}{D(u)^3}.$$

This derivative is positive when $u > \rho$. Since $u = \cos\varphi$ decreases as φ increases, $\cos(\alpha)$ decreases in φ on the operating regime $\cos\varphi > \rho$, hence α increases.

For (ii), differentiate with respect to ρ at fixed φ:

$$\frac{d\cos(\alpha)}{d\rho} = \frac{-\cos\varphi \cdot D(\rho)^2 - (1 - \rho\cos\varphi)(-\cos\varphi + \rho) \cdot D(\rho)^{-1} \cdot D(\rho)}{D(\rho)^3} \cdot D(\rho) = \frac{-\sin^2\varphi \cdot \rho}{D(\rho)^3} \leq 0,$$

with equality only when $\varphi = 0$. Therefore $\cos(\alpha)$ decreases in ρ and α increases.

For (iii), setting $\cos(\alpha) = 0$ in (8) requires the numerator to vanish: $1 - \rho\cos\varphi = 0$, i.e. $\rho\cos\varphi = 1$. □

B Experimental configuration and additional details

Algorithmic configuration. Both projection-based and Lagrangian-based safe GBRL share the following core hyperparameters:

Hyperparameter	Value
Total environment steps	1,000,000
n_{steps} per environment	512
n_{envs}	16
Batch size	512
n_{epochs} per update	5
Discount factor γ	0.99
GAE λ	0.98
PPO clip range	0.2
Entropy coefficient	0.015
Grow policy	Oblivious trees
Tree cap $N_{\max}$	25,000 per ensemble
Cost threshold κ	25
Seeds per configuration	10

Projection-specific parameters. The projection coefficient is computed by (6) with the corrected violation signal $e_k = J_C^{\text{undisc}}(\pi_k) - \kappa$. A hard cap $\lambda^* \leq 1.0$ is enforced. The projection is computed once per minibatch.

Lagrangian-specific parameters. The dual variable is updated by $\lambda \leftarrow [\lambda + \eta_\lambda (J_C^{\text{undisc}}(\pi_k) - \kappa)]_+$ with $\eta_\lambda = 5 \times 10^{-3}$. A hard cap $\lambda \leq 2.0$ is enforced. The update is computed once per training iteration.

Environments. All three Safety-Gymnasium environments are used at their default configurations from the Safety-Gymnasium repository, with episode length 1000 steps and the cost threshold κ unified at 25. No modifications to the environment rewards or costs are made.

Computation of metrics. Explained variance is computed as $\mathrm{EV} = 1 - \mathrm{Var}(R - V)/\mathrm{Var}(R)$, where R is the empirical Monte Carlo return and V is the value-function prediction at the corresponding state. *Mean* statistics in Table 1 are averaged over the full training run (approximately 122 logged data points per seed); *final* statistics are the last logged data point per seed. *Periods* $< \kappa$ is the fraction of logged episodes with cost below the threshold, averaged across seeds.

Hardware. All experiments were run on a HPC cluster with 4x NVIDIA H100 GPUs per node. Computational resources were provided by XXXXX.