Graphical Abstract

Hybrid Safe Reinforcement Learning: Tackling Distribution Shift and Outliers with the Student-t's Process

Xavier Hickman, Yang Lu, Daniel Prince

Highlights

Hybrid Safe Reinforcement Learning: Tackling Distribution Shift and Outliers with the Student-t's Process

Xavier Hickman, Yang Lu, Daniel Prince

- Tail decay rates of Gaussian and Student-t processes' densities are explicitly derived.
- A novel TP-based robust hybrid offline-online SRL algorithm maximizes adaptability.
- Using t-posterior variance to infer safety under covariate shift and outliers is novel in SRL.
- A novel criterion evaluates two SRL agents, identifying the 'safer' w.r.t safety constraints

Hybrid Safe Reinforcement Learning: Tackling Distribution Shift and Outliers with the Student-t's Process

Xavier Hickman[a], Yang Lu[a,*], Daniel Prince[a]

[a]*School of Computing, InfoLab21, South Dr, Lancaster University, Bailrigg, Lancaster, LA1 4YW, UK*

Abstract

Safe reinforcement learning (SRL) aims to optimize control policies that maximise long-term reward, while adhering to safety constraints. SRL has many real-world applications such as, autonomous vehicles, industrial robotics, and healthcare. Recent advances in offline reinforcement learning (RL) - where agents learn policies from static datasets without interacting with the environment - have made it a promising approach to derive safe control policies. However, offline RL faces significant challenges, such as covariate shift and outliers in the data, which can lead to suboptimal policies. Similarly, online SRL, which derives safe policies through real-time environment interaction, struggles with outliers and often relies on unrealistic regularity assumptions, limiting its practicality. This paper addresses these challenges by proposing a hybrid-offline-online approach. First, prior knowledge from offline learning guides online exploration. Then, during online learning, we replace the popular Gaussian Process (GP) with the Student-t's Process (TP) to enhance robustness to covariate shift and outliers.

Keywords: Safe Reinforcement Learning, Bayesian non-parametric Models, Constrained Markov Decision Processes, Robustness to Outliers, Covariate Shift, Uncertainty, Gaussian Process, Student-t's Process

*y.lu44@lancaster.ac.uk

1. Introduction

Safe reinforcement learning (SRL) is a branch of reinforcement learning (RL) that considers safety constraints (e.g. cost functions) as the primary objective, followed by long-term cumulative reward. SRL finds applications in safety-critical contexts such as autonomous driving, industrial robotics, wireless security, and biomedical science Gu (2022). Hybrid training schemes are achievable in may real-world SRL application. Using autonomous driving as an example, agents can be trained offline with existing trajectories collected using either human drivers or previous AV missions. Online training can then be leveraged to fine tune the system model and learn more mission specific behaviour. Healthcare is another example where an agent can be trained on existing system trajectories collected using human physicians. Then agents can then fine tune online in a live clinical setting, based on real-time patient data. Industrial robotics is a further example where an agent can be trained offline using trajectories from previous missions or historical data. Then the agent can fine tune its policy online by performing in a live industrial environment. SRL could greatly benefit from ideas used in offline RL which derives policies from offline/static datasets in contexts where prior knowledge of the problem is available and models of the environment do not deviate too much from the deployment environment. Much of the current focus in SRL literature is geared towards online methods, where the agent starts with very little knowledge of the problem and derives a safe control policy while deployed in a live environment, often without violating any safety constraints. These methods often come with strong formal guarantees for safety however, these guarantees are derived using sometimes unrealistic regularity assumptions which if violated could lead to unsafe behaviour Levine (2020).

1.1. Common Regularity Assumptions

Regularity assumptions such as L-Lipschitz continuity are commonly used in SRL problem formulations as they guarantee that similar states have similar cost and reward behaviours Turchetta (2016) Budd (2020) Wachi (2018) Bottero (2022) Sui (2015). The fundamental conditions for establishing regularity assumptions, such as L-Lipschitz continuity, include function smoothness, differentiability, and continuity. However, this assumption can be restrictive or may not hold in many practical scenarios such as autonomous driving, healthcare applications, robotics in unstructured environments, air traffic control and energy management in smart grids/microgrids. Using the

autonomous driving use case as an example, regularity assumptions often do not hold primarily because the environment is likely to change rapidly and unpredictably. For instance, a pedestrian suddenly appearing in the path of the vehicle, or a sensing error like a stationary pedestrian being misclassified as a static object. These scenarios represent sudden and significant changes in the cost/safety function resulting in discontinuities and violating the base regularity assumption(s). Taking medical treatment as a further example, a rapid and unpredictable change in this class of environments could be a patient's state (possibly described by vital signs) rapidly changing without discernible reason, again causing a discontinuity in the cost/safety function and violating regularity assumption(s).

1.2. Covariate Shift

In such situations where the regularity conditions do not hold, the problem known as covariate shift becomes extremely critical. In conventional machine learning, covariate shift occurs when the distribution of input features changes between the training and testing phases, while the conditional distribution of the target given the inputs remains unchanged Candela (2008) (Section 7.2.2, P.110). In the context of model-free reinforcement learning (RL) or safe reinforcement learning (SRL), the input features correspond to state-action pairs, and the targets represent evaluations of either cost or reward functions. The distribution of input features is the distribution of state-action pairs across the state space. In our case, this specifically pertains to the distribution of unsafe states, which form a subset of the state space. The conditional distribution of the target given the inputs—i.e., the distribution of costs given state-action pairs—remains constant. Intuitively, this conditional distribution can be interpreted as the rule or constraint that designates states as safe or unsafe.

Consider a problem where states in the state space are characterized by a safety property, represented as a scalar value. States with safety property values above a predefined threshold are deemed unsafe, while all others are considered safe. Under a covariate shift, the states designated as unsafe during training may differ from those during deployment. In other words, the specific states that exceed the safety threshold during training may not be the same as those that exceed it during deployment. However, the rule for designating states as unsafe—i.e., the safety evaluation exceeding the threshold—remains unchanged. To summarize: the probability that a state-action pair associated with an unsafe transition will yield a positive cost evaluation is always 1. This

principle holds indefinitely, but the specific states labelled as unsafe may shift between training and deployment.

In practice, this phenomenon often arises due to sudden changes in state distributions, particularly in the distribution of unsafe states. Such shifts can cause policies learned in an offline setting to exhibit unsafe behaviour. This occurs because offline policies are trained on a specific distribution of unsafe states and are then tasked with making predictions about future costs when that distribution has shifted. Furthermore, covariate shifts can disrupt the performance of online SRL methods that rely on regularity assumptions. Changes in the cost function during learning can delay or even prevent convergence, leading to unsafe behaviour.

Another significant challenge in both offline and online SRL is the presence of outliers or extreme events. In the offline setting, static datasets may contain outlying observations, which can cause the algorithm to converge to a suboptimal or unsafe policy. In the online setting, SRL algorithms may be adversely affected if the environment exhibits extreme or outlying events. In contexts where regularity assumptions—such as L-Lipschitz continuity—would typically hold, the presence of outliers can violate the foundational conditions that enable the use of such assumptions. As a result, the formal guarantees provided by online methods—such as upper bounds on cost and safety violations—no longer hold, as the predictability assumptions are negated by the presence of outliers.

1.3. Student-t Process vs Gaussian Process

We address the problem of SRL in environments where outliers and covariate shifts are likely. Our work takes a hybrid learning approach by combining ideas from both offline RL/SRL and online SRL. We propose a hybrid offline-online method capable of handling both outliers in offline datasets and covariate shifts between offline datasets and online deployment environments. Our algorithm makes use of robust models and both offline and continued online training to address these challenges. Much of the existing online Safe Reinforcement Learning (SRL) literature relies on probabilistic models known as Gaussian Processes (GPs). However, GPs are built on the Gaussian distribution, which has very light, fixed tails, making them highly susceptible to outliers Shah (2014). Additionally, GPs assume *homoscedastic noise* Rasmussen (2005), meaning they presume all random variables have the same finite variance. This results in a rigid method for computing predictive variance. To address these limitations, we leverage the Student-t Process

(TP), a probabilistic model similar to the GP but built on the heavier-tailed Student-t distribution. The tail weight of the TP is controlled by a dedicated parameter, ν, providing greater flexibility in modelling data with outliers. Unlike the GP, the TP assumes *heteroscedastic noise*, meaning it allows the variance of random variables to vary. This difference in noise assumptions is reflected in the TP's formula for computing posterior predictive variance (uncertainty), which theoretically makes it more robust to shifts in covariate distributions (see Section 3.1 for details). These properties—greater flexibility in tail behaviour and the ability to model heteroscedastic noise—make the TP a far more promising choice than the GP for SRL in environments with outliers and distribution shifts

1.4. Related Work

1.4.1. Offline SRL

Offline SRL has proven a promising way to mitigate the risk of online interactions when searching for a safe optimal control policy. Authors in Zheng (2024) Guan (2023) Cao (2024) Li (2023) all approach the problem of deriving safe control policies from offline datasets. Authors in Guan (2023) introduce non-parametric variational distributions using probabilistic inference to replace parametrized policies, improving the flexibility of learned policies. While this approach offers a solution towards more flexible policies, they do not explicitly consider covariate shifts or outliers in the dataset. Authors in Zheng (2024) introduce a safe offline approach using safe-control theory. The solution addresses the problem of hard safety constraints in offline datasets using a decoupled learning process, however, the method is entirely confined to an offline setting, with no way to facilitate hybrid or online learning, further the issue of outliers in the offline datasets is not addressed. Authors in Cao (2024) study the offline SRL problem using goal-conditioned policies for safety-critical tasks. The solution leverages imitation learning, by training an offline policy with out-of-distribution (OOD) action detection to learn goal-reaching tasks. The work also trains a recovery policy to ensure the agent's trajectory does not enter unsafe regions of the state space. The paper discusses the importance of the quality of the offline dataset and explains that the recovery policy actively filters out unsafe trajectories from the dataset. While this approach has merit, the paper makes no mention of outliers or covariate shifts, implying the method would still be susceptible to such occurrences.

1.4.2. Online SRL and Safe Exploration

Online SRL has been widely studied in the literature and authors in Turchetta (2016) Budd (2020) Wachi (2018) Bottero (2022) Sui (2015) all offer strong approaches to tackle the safe exploration problem using GPs. Safe exploration is one of the more popular open challenges explored by online SRL literature, and GPs have been the favoured model in many works due to their sample efficiency and natural notion of uncertainty. These works all assume the safety/cost function to be L-Lipschitz continuous w.r.t some metric $d(\cdot, \cdot)$. As previously mentioned this assumption is reasonable in so much as it makes the problem of safe exploration tractable. However, if outliers are present during the online learning process may cause the online SRL approaches to converge to suboptimal policies because the L-Lipschitz assumption would be violated.

An alternative view is offered by Schreiter (2015) where *safe exploration* for active learning is studied using GPs without the L-Lipschitz continuity assumption. To clarify; works addressing *safe exploration* typically cannot make any constraint violations during online learning whereas *safe exploration* for active learning can violate constraints, in the pursuit of improved learning Schreiter (2015). Works such as Turchetta (2016) make no mention of updating kernel parameters during the online learning process. The authors appear to choose some appropriate values for kernel parameters and keep them fixed during learning, relying on aforementioned regularity assumptions and new observations to condition future predictions on. The kernel used in Turchetta (2016) was a Matern kernel with $\nu = 5/2$ Rasmussen (2005), this kernel is parameterized with a length scale parameter l and an amplitude parameter σ^2, the authors state they chose $l = 14.5, \sigma^2 = 10$. These parameters were chosen based on limited prior knowledge of the function. This strategy is perfectly acceptable under the regularity assumption and relies on the principle that a GP can approximate smooth functions arbitrarily well if infinitely many observations are provided, even with fixed sub-optimal, kernel parameters Rasmussen (2005). However the role of the kernel parameters defines the prior smoothness of the functions being modelled and if the function changes in an abrupt and unpredictable way, the formal guarantees and empirical performances offered by such works do not hold, which motivates the need to consider dropping such regularity assumptions in addition to updating kernel parameters during learning/deployment when using probabilistic models such

as GPs or TPs. Moreover, this raises the question of why a strategy to continuously re-evaluate kernel parameters during learning or deployment is not utilised, as it could theoretically result in a more optimal model.

In the wider online SRL literature, researchers have explored diverse approaches to ensuring safety while maintaining learning efficiency. Yu (2019) proposed a convergent policy optimization framework that integrates safety constraints directly into the policy update process, ensuring both safe exploration and convergence guarantees. This method provides strong theoretical assurances regarding safety, making it a significant contribution to constrained policy optimization. Shao (2021) introduced Reachability-based Trajectory Safeguard (RTS), a safety layer for continuous control that utilizes reachability analysis to enforce safety constraints in real time. This approach effectively balances safety and resource efficiency. Complementing these approaches, Zhou (2024) focused on system-level safety in mixed-autonomy platoons, leveraging safe RL techniques to enhance coordination and safety in multi-agent autonomous vehicle systems. Zhang (2024) proposed a Barrier Lyapunov Function (BLF)-based safe RL framework combined with optimized backstepping. This approach constrains the system's state within predefined safety boundaries, ensuring safe operation while maximizing performance. While these works provide valuable insights into safe policy learning and enforcement mechanisms, we note that they do not explicitly address robustness to outliers or adaptation to distribution shifts–two critical challenges that our work specifically targets.

1.5. Contribution

This paper studies covariate shift in the presence of outliers in constrained Markov decision processes (CMDPs) using a hybrid offline-online SRL approach. The contributions of this work are fourfold:

(1) The tail decay rates of the GP and TP's probability densities are for the first time explicitly derived. Our analysis demonstrates that the Student's t-distribution exhibits a polynomially decaying tail, controlled by a degrees of freedom (DOF) parameter. We also demonstrate that the Gaussian distribution has an exponential tail decay rate. This key theoretical insight shows that the TP is better equipped to capture heavy tailed behaviour (provided a small enough DOF), making it a more robust choice in contexts where outliers or extreme events are more likely. Particularly, TPs, with their polynomial tail decay, offer greater resilience to outliers, better uncertainty quantification, and improved adaptation to distribution shifts compared to GPs. These

properties enhance safe policy learning, trustworthy decision-making, and generalization in deployment. This contribution not only strengthens the mathematical intuition behind the choice of the TP but also provides a principled justification for its application in SRL settings where robustness and flexibility are very important.

(2) A novel TP-based robust hybrid offline-online SRL algorithm is developed. The novelty of our algorithm comes in part from the use of a Student's t-Process trained both offline and online to maximise adaptability. Secondly, the use of the Student's t-posterior predictive variance to infer the safety of unseen states, under covariate shift and outliers is novel in an SRL context. Finally, the use of the heavier-tailed Student's t-distribution to enhance this algorithm's robustness to outliers, both in offline data and online data is novel in an SRL context.

(3) Comprehensive experiments are conducted to evaluate the performance of the proposed TP-based approach under covariate shift in the presence of outliers. The performance of our approach is then compared with the already popular GP-based approach, and we demonstrate the superiority of our method.

(4) A novel preference criterion is developed and used to evaluate two safety constraint-satisfying agents solving the same CMDP, determining the "safer" agent.

2. Problem Formulation

We consider a deterministic finite-horizon constrained Markov decision process (CMDP) with distribution-dependant reward and cost functions Altman (1999), this is expressed as follows:

$$CMDP = \langle S, A, P, R, C, \gamma, d, \rho \rangle \tag{1}$$

In (1) S and A are the state and action spaces. The deterministic transition probability function $P(s' \mid s, a)$ describes the probability of transitioning to state s' using action a while in state s. ρ is the distribution of states that we expect to change under a covariate shift. The distribution-dependant immediate reward and cost for taking action $a \in A$ in state $s \in S$ are unknown functions $R_\rho(s, a) \in \mathbb{R}$ and $C_\rho(s, a) \in \mathbb{R}_+$, respectively. The distribution dependency of the reward and cost functions means that if the distribution of unsafe states ρ remains constant, evaluations of $R_\rho(s, a)$ and $C_\rho(s, a)$ also

remain constant. However in the event ρ shifts to ρ', those same evaluations of $R_\rho(s,a)$ and $C_\rho(s,a)$ will also shift under the new distribution of unsafe states. Formally, $C_\rho(s,a) \neq C_{\rho'}(s,a)$. This represents the covariate shift in our problem setup. We also note that outliers are assumed to exist in the target space of the reward and cost functions. $\gamma \in [0,1)$ is a scalar discount factor. d represents a safety tolerance threshold, this is used to bound safety constraint violations. In a real world context d is set based on domain specific and problem specific knowledge.

Building on the work by Ray (2019) we formulate the following optimization problem.

$$\begin{aligned} \max_{\pi_\theta} \quad & \mathop{\mathbb{E}}_{\tau\sim\pi_\theta}\Big[\sum_{t=0}^{T} r_t\Big] \\ \text{s.t.} \quad & \mathop{\mathbb{E}}_{\tau\sim\pi_\theta}\Big[\sum_{t=0}^{T} c_t\Big] \leq d, \end{aligned} \tag{2}$$

Eq.(2) formulates an optimal control problem. The agent must maximize the finite horizon undiscounted reward $G_T = r_1 + r_2 + \cdots + r_T$ subject to the cumulative cost $C_T = c_1 + c_2 + \cdots + c_T \leq d$ where d is some problem specific safety threshold. π_θ denotes the control policy parametrized by θ. τ denotes the trajectory (s, a, r, s') that the agent follows using the policy π_θ.

2.1. Preliminaries on the GP and TP

A GP is a stochastic process defined as a collection of random variables any finite number of which have a joint Gaussian distribution Rasmussen (2005). In the context of optimization a GP can be considered a prior over a space of all possible objective functions Tracey (2018) parametrized by a mean function $\mu(x)$ and a kernel function $k(x, x')$. The kernel expresses the covariance between any two input points $x, x' \in D$ where D is the set of all evaluations of the objective function $D = \{(x_i, y_i)\}_{i=0}^{n}$.

Similarly a TP is a stochastic process defined as a collection of random variables, any finite number of which have a multivariate Student's t-distribution Shah (2014). Like the GP, the TP is characterised by a mean function $\mu(x)$ and a kernel function $k(x, x')$, with an additional *tail weight parameter* $\nu \geq 2, \nu \in \mathbb{R}$. This parameter controls the heaviness of the tails in the distribution. Like the GP, the TP expresses a distribution over functions

where each function is a possible realization of the underlying process. Both models are Bayesian regression models used to approximate functions, with their Bayesian nature naturally providing a predictive uncertainty measure. Predictions from either a GP or TP include both a mean estimate $y = \hat{f}(x_*)$ and an associated uncertainty measure, which corresponds to the variance of the prediction at point x_* In the one-dimensional case, the posterior predictive distribution of a GP is a univariate Gaussian defined by mean $\mu(x_*)$ and variance $\sigma^2(x_*)$. The mean function sample $\mu(x_*)$ quantifies the evaluation of the mean function at x_*, while $\sigma^2(x_*)$ represents the associated uncertainty, with lower variance indicating higher confidence in the prediction.

To summarize, the key differences between the two models are as follows. The GP is built on the Gaussian distribution, which has lighter tails, making it more susceptible to outliers. In contrast, the TP is based on the Student-t distribution, which includes an adjustable tail-weight parameter, ν. This parameter allows the TP to model both heavy-tailed and light-tailed distributions, making it more robust to outliers. Additionally, the GP assumes homoscedastic noise (constant variance across input points), which limits its flexibility in scenarios involving distribution shifts. On the other hand, the TP accommodates heteroscedastic noise (variance that depends on input points), making it better suited for handling such shifts. Formal definitions of both the TP and GP are provided in the appendices.

2.2. Technical Assumptions:

Similarly to Turchetta (2016) we assume by Mercer's theorem (1) Ingo (2008) the existence of a continuous, symmetric, and positive semi-definite kernel function.

Theorem 1. ***Mercer's Theorem:*** *Let $\mathcal{X}$ be a compact metric space and $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a continuous kernel. Furthermore, let μ be finite Borel measure with supp$(\mu) = \mathcal{X}$. Then, for $\phi_i(.)_{i\in\mathcal{I}}$ and $(\lambda_i)_{i\in\mathcal{I}}$ we have:*

$$k(x, x') = \sum_{i\in\mathcal{I}} \lambda_i \phi_i(x)\phi_i(x'), \qquad x, x' \in \mathcal{X} \tag{3}$$

where the series convergence is absolute and uniform.

The kernel function is defined as $k : X \times X \to \mathbb{R}$ for any finite set of points $x_1, x_2, \cdots, x_n \in X$, the kernel evaluates similarity between pairs of points.

Specifically, each entry Σ_{ij} in the covariance matrix is computed as $k(x_i, x_j)$, which quantifies the similarity between states x_i and x_j. We assume that the cost C and reward R functions lie within a reproducing kernel Hilbert space (RKHS) induced by the kernel function k. The RKHS associated with k must satisfy the reproducing property, which is a fundamental characteristic of such vector spaces. This property ensures that the evaluation of any function f in the RKHS at a point x can be expressed as an inner product between f and the kernel function centred at x. Formally, for any f in the RKHS and $x \in S$, we have $f(x) = \langle f, k(x, \cdot)\rangle$. These assumptions allow us to treat the cost and reward functions as stochastic processes more specifically either as GPs or TPs.

2.3. The Effects of Outliers on Mercer's Conditions

Mercer's Theorem expresses the conditions required to treat the reward and cost functions as stochastic processes. The first condition is symmetry in the kernel function $k(x, x') = k(x', x), \forall x, x' \in S$. The second is boundedness of the kernel function $|k(x, x')| < M, \forall x, x' \in S$ where M is some constant. The third is continuity on the kernel where the kernel function must be continuous on the square $S \times S$. Finally, the kernel must be positive semi-definite for any finite set of points Ingo (2008). The presence of outliers can violate conditions such as boundedness, continuity, or positive semi-definiteness of kernels. For instance, outliers can lead to unbounded kernels, dominating their behaviour. Boundedness refers to the idea that the kernel doesn't produce arbitrarily large values, so their should be a limit to how large the kernel functions range can become. Outliers can violate continuity and smoothness, by causing abrupt changes or even discontinuities in the function. Continuity refers to the idea that small changes in the input should lead to small changes in the kernels output. Such violations can lead to a failure in the eigenfunction expansion. In this case, Mercer's theorem cannot guarantee the existence of valid eigenfunctions and eigenvalues, as it otherwise does under satisfied conditions.

2.4. Practical Example

Consider an autonomous robot operating in a large warehouse, tasked with transporting goods from a starting location to a goal location. The warehouse is structured as a grid-like environment, where the robot must navigate through aisles, shelves, and workstations while avoiding unsafe states. In this setting, unsafe states include obstacles such as forklifts, fallen pallets,

workers, maintenance zones, and no-entry areas. The robot is first trained offline using historical data from sub-optimal trajectories collected during previous missions. These trajectories may contain outliers, representing rare events (e.g., a forklift breaking down in an unusual location) or sensor errors (e.g., misclassified obstacles). Once offline training is complete, the robot is deployed in a live environment, where it encounters a distribution shift. This shift can result from changes in the warehouse layout, the introduction of new obstacles, or evolving worker traffic patterns. While we assume that unsafe states remain fixed within a single mission, they may evolve over time due to operational changes. To ensure safe and efficient operation, the robot must adapt to these environmental changes between offline training and deployment. When the robot encounters previously unseen scenarios where it cannot reliably predict safety constraints, it must perform online re-training to update its model and improve decision-making in real time. This reflects real-world warehouse conditions, where environments remain temporarily stable within a task but gradually evolve over extended periods.

3. Theoretical foundations

This section lays the theoretical groundwork for our study. First, we discuss how the TP and GP compute posterior predictive variance (uncertainty), we highlight key differences between the two equations and the implications of those differences. Second, we explain and discuss the theoretical robustness of the TP to outlying or extreme observations and the implications of this property. Finally (our contribution), we analyze the tail behaviour of both the Student's t-distribution's density function and the standard Gaussian density function which cannot be found in relevant literature to our best knowledge.

3.1. Computing Posterior Predictive Variance (Uncertainty)

Both the GP and TP use Eq.(4) to compute the mean sample for a function evaluation around the point x_* (for the same kernel).

$$\mu(x_*) = k(x_*, X)[K(X, X) + \sigma^2 I]^{-1} y \tag{4}$$

$$\sigma^2(x_*) = k(x_*, x_*) - k(x_*, X)[K(X, X) + \sigma^2 I]^{-1} k(X, x_*) \tag{5}$$

$$\hat{K} = \frac{(\nu + \tilde{y}^T [K(X_*, X_*)]^{-1} \tilde{y} - 2)}{\nu + |D| - 2} \cdot \left(k(x_*, x_*) - k(x_*, X)[K(X, X) + \sigma^2 I]^{-1} k(X, x_*)\right) \tag{6}$$

$k(x_*, X)$ and $k(X, x_*)$ are kernel vectors expressing the covariance between the point x_* and each of the training points in the observation space. $K(X, X) + \sigma^2 I$ denotes the kernel matrix scaled by a noise variance term σ^2 along the diagonal of the matrix I. y denotes the vector of observed (training) outputs. Eq.(5) computes the GP's posterior predictive variance, reflecting uncertainty in predictions for x_*. In contrast, the TP computes uncertainty differently Li (2023). While (6) shares similarities with the GP's variance equation, it has an explicit dependence on training observations $\tilde{y}$ Tracey (2018), scaling the posterior variance. $[K(X_*, X_*)]^{-1}\tilde{y}$ is the inverse of the covariance matrix of the test points scaled by the training outputs, and if this is larger than $|D|$, then the posterior predictive variance will be greater than that of the GP taken for the same training and test points. Eq.(6) also has a dependence on the tail weight parameter ν. Smaller values for ν lead to increased uncertainty estimates, which can be advantageous to SRL when outliers are present. A smaller ν allows the TP to assign higher probabilities to outliers and informs the computation of predictive variance, providing a more realistic estimation of uncertainty Li (2023). The differences in the posterior predictive variance Eq.(6) make the TP better suited for handling the covariate shift problem compared to the GP. Additionally, these enhancements address outliers in offline datasets.

In summary, while both the GP and TP share a common framework for computing mean predictions, their approaches to compute predictive uncertainty differ significantly. The GP's Gaussian assumption makes it efficient and well-suited for smooth, well-behaved data, but it struggles with outliers and distribution shifts. The TP on the other hand, leverages its heavier-tailed distribution and data dependant variance scaling to provide more robust uncertainty estimates, making it a better choice for challenging real-world applications.

3.2. Robustness of the TP to Outliers

The TP's tolerance of outliers stems from the weight of its density's tails. Put simply, this is the rate at which the tails of the Student's t-PDF decay Shah (2014). The TP's tail weight parameter ν determines the tail decay rate, i.e. when ν is small, the tails are heavier, conversely as ν gets larger, the tails get lighter. In fact as $\nu \to \infty$ the TP converges to a GP. Contrast this with the GP which has no such tail weight control, the process will naturally assign lower probabilities to outliers, thus making them less likely.

In principle, optimizing ν online, allowing the model to dynamically adjust its robustness to outliers as new data arrives is an ideal approach. However, in practice, this is highly challenging and computationally expensive. Prior works in the statistics and machine learning literature, such as Shah (2014), typically fix ν rather than optimizing it dynamically. Studies exploring methods for estimating ν, such as Jylanki (2011), found that while various optimization algorithms can be employed, the process is often slow and computationally inefficient. Their recommendation was to fix $\nu = 4$, as it provides a reasonable balance between robustness to outliers and model flexibility when outliers are present. Another study Rover (2011) explored estimating ν in a Student-t noise model and found that empirical quantile-based estimation can yield consistent results. However, they noted practical ambiguities, such as the choice of quantiles and the time intervals over which estimates should be computed, making it difficult to generalize a robust online estimation strategy. For these reasons we follow the work of Shah (2014) and Jylanki (2011) and fix $\nu = 3$ prior to training, this ensures the TP can robustly tolerate outliers while making accurate and meaningful inferences.

3.3. Theoretical Analysis of Probability Densities

We will demonstrate how Karamata's theory of regular variation (Definition (2)) Karamata (1930) examines functions with specific asymptotic behaviours. This analysis will illustrate the growth and decay of the Gaussian and Student's t-PDFs as their input tends towards infinity. Intuition about tail weight and tail decay rate is crucial to understanding why the TP is more robust to outliers than the GP.

Definition 2. *A function f is said to be regularly varying at infinity if it is real-valued, positive and measurable on $[A, \infty)$, for some $A > 0$, and if for each $\lambda > 0$*

$$\lim_{x \to \infty} \frac{f(\lambda x)}{f(x)} = \lambda^p \tag{7}$$

for some p in the interval $-\infty < p < \infty$. (p is called the index of regular variation)

Definition (2) formalises Karamata's theory of regular variation (RV) Karamata (1930). Karamata's RV can be used to study the behaviour of positive real functions ($f(x) \in \mathbb{R}_+$) that exhibit certain asymptotic properties. We

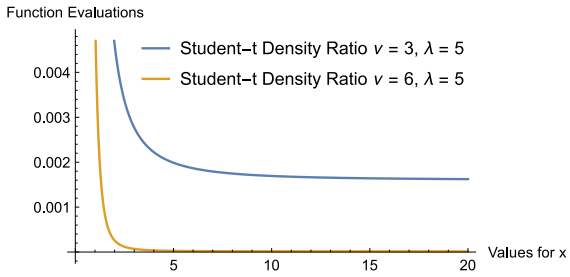
use Karamata's RV to understand how the Gaussian PDF and the Student's t-PDF grows and decays as their inputs tend towards infinity.

$$\lim_{x\to\infty} \frac{\frac{\Gamma(\frac{\nu+1}{2})}{\sqrt{\nu\pi}\Gamma(\frac{\nu}{2})}\left(1+\frac{\lambda x^2}{\nu}\right)^{-\frac{\nu+1}{2}}}{\frac{\Gamma(\frac{\nu+1}{2})}{\sqrt{\nu\pi}\Gamma(\frac{\nu}{2})}\left(1+\frac{x^2}{\nu}\right)^{-\frac{\nu+1}{2}}} = \lambda^{-1-\nu}, \lambda > 1 \tag{8}$$

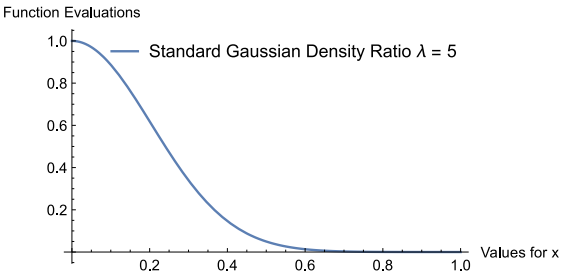
The result in (8) was obtained using the **Limit** function of *Mathematica v13.2.1.0*, the *PDF-Tail-Analysis-Karamatas-RV.nb Mathematica* file can be found in the associated code repository (see Appendix E), demonstrating the derivation Hickman (2024). Eq.(8) shows, using Karamata's RV, the Student's t-density is $f(x) \in RV(p)$, $p = -1 - \nu$. We then apply Karamata's RV to the Gaussian density to obtain its tail decay rate as $x \to \infty$. We use the standard Gaussian density, $\mathcal{N}(0, 1)$.

$$\lim_{x\to\infty} \frac{\frac{1}{\sqrt{2\pi}}\exp^{-\frac{\lambda x^2}{2}}}{\frac{1}{\sqrt{2\pi}}\exp^{-\frac{x^2}{2}}} = 0 \tag{9}$$

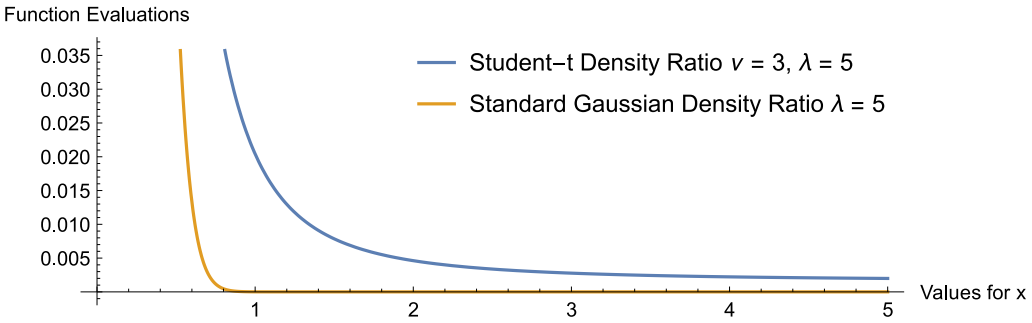
The limit given in (9) shows Karamata's treatment (7) of the Gaussian density and we see it approaches 0 as $x \to \infty$. This result demonstrates that the tails of the Gaussian density decay at a faster rate than that of the Student's t-density (when ν is sufficiently small), therefore the Gaussian density is $f(x) \notin RV(p)$. Figure 1 demonstrates empirically the decay rate of the ratios given in (9) and (8). All graphed functions are parameterized with $\lambda = 5$. Figure 1(a) demonstrates the decay of the ratio in (8) graphed over $x \in [0, 20]$ with two variations of ν. In the case of $\nu = 3$ the function decays at a slower rate than the case where $\nu = 6$, demonstrating that as $\nu \to \infty$ the Student's t-distributions approaches a Gaussian. Figure 1(b) shows the ratio in (9) graphed over the interval $x \in [0, 1]$. By comparing figures 1(a) and 1(b) it is clear to see that the Gaussian density converges to 0 much faster than the Student's t-density. The Gaussian ratio approaches 0 when $x \sim= 0.6$, compared with the Student's t-ratio, which in the case of $\nu = 3$ moves in the direction of 0 but ultimately plateaus as x grows larger. In the case of $\nu = 6$ the function approaches 0 as $x \sim= 3$. Figure 1(c) demonstrates the decay rates of both ratios graphed on the same interval, as we might expect the Gaussian ratio decays at a much faster rate than the Student's t, thus demonstrating empirically the results in (8) and (9). Using the result in (8) of $\lambda^{-1-\nu}$, we can check our graphs by substituting in the values for λ and ν.



(a) Student's t-Density Decay ($\nu = (3, 6), \lambda = 5$)



(b) Gaussian Density Decay ($\lambda = 5$)



(c) Density Decay Comparison

Figure 1: Density Tail Decay Rates Comparison

For example take the case of $\nu = 3$ in 1(a), if we evaluate $5^{-1-3}, \lambda = 5, \nu = 3$ we get 0.0016, which is where the blue curve approximately converges to in figure 1(a).

Definition 3. *Let f and g define density functions. Then f is said to have lighter, equivalent, or heavier tails than g if,*

$$\lim_{x \to \infty} \frac{f(x)}{g(x)} \tag{10}$$

is zero, a constant $0 < c < \infty$, or ∞ respectively

Alternatively, can also analyze the Student's t-density using the Gaussian density as a reference Rojo (2013). Definition (3) formalises a method to determine whether one function has lighter, equivalent, or heavier tails than another by taking its limit as $x \to \infty$. We leverage this as additional

theoretical evidence to illustrate the weight of the Student's t-density's tails.

$$\lim_{x\to\infty} \frac{\frac{1}{\sqrt{2\pi}}\exp^{-\frac{x^2}{2}}}{\frac{\Gamma(\frac{\nu+1}{2})}{\sqrt{\nu\pi}\Gamma(\frac{\nu}{2})}\left(1+\frac{x^2}{\nu}\right)^{-\frac{\nu+1}{2}}} = 0 \tag{11}$$

Eq.11 shows the limit at infinity of the ratio of the Gaussian density (numerator) and the Student's t-density (denominator). The Student's t-density has heavier tails than the reference Gaussian density because the limit at infinity of the ratio is 0. The result in (11) was obtained using the **Limit** function of *Mathematica v13.2.1.0*, the *PDF-Tail-Analysis-Karamatas-RV.nb Mathematica* file can be found in the associated code repository, demonstrating the derivation (see Appendix E).

3.4. Impact of Tail Decay Rate on Learning

Having established the different decay rates of both the Gaussian distribution and the Student's t-distribution, we must now explore the effects of these differing decay rates on the learning process for each model. As shown in equations (8) and (9) and illustrated in figure 1 the tails of the Student's t-distribution decay at a much slower rate than that of the Gaussian distribution. This means the tails of the t-distribution are heavy and the tails of the Gaussian are light, specifically the Student-t's tails are heavier than the Gaussian's as shown in Eq.(10). During training, we assume the data to contain outliers therefore the goal of each model will be to learn a mapping of state-action pairs to the cost or reward functions in the presence of outliers. The TP will assume noise follows a Student-t's distribution, this means it will assign higher probabilities to extreme observations or outliers, this naturally down-weights the influence of outliers as they are not considered drastically improbable events by the model. When fitting the model to the data the TP will learn to discount points in the observation space that deviate too far from the mean and due to their low influence on the overall likelihood, this allows the TP to learn a good representation of the underlying function Shah (2014), Andrade (2011). In contrast to this, the GP assumes that noise in observations follows a Gaussian distribution, which as we have shown has light tails. This means the GP will assign very low probabilities to observations that deviate drastically from the mean, treating outliers as highly unlikely events. During learning the GPs aim is to maximise the likelihood of the observed data including the outliers, and because outliers

are treated as highly improbable, the GP will adjust the learned mean and variance to accommodate the outliers, thus skewing the representation of the underlying function and resulting to a poor approximation. During inference, the outliers can result in erratic predictions and cause overfitting Rasmussen (2005), Andrade (2011).

4. Algorithm Design

We will now formally state the Safe and Robust Stochastic Process (SARSP) algorithm. The SARSP algorithm is a model-free SRL algorithm (similar to Q-learning and SARSA) that does not explicitly model the transition dynamics of the CMDP (1), instead learns the reward and cost functions using a stochastic process.

4.1. Preliminaries

Algorithm (1) formalises how an agent chooses safe and conducive actions online during deployment. As mentioned previously our approach is hybrid and makes use of offline training and online training. First two models are trained using a dataset of suboptimal trajectories, one TP is trained to approximate the reward function and a second TP is trained to approximate the cost function. The training scheme for the TPs/GPs follows the conventional approach of optimising the kernel hyperparameters using gradient based methods. Both TP models are passed to Algorithm (1) for online training and deployment. The dataset of suboptimal trajectories used during offline training are assumed to contain outliers, and the distribution of covariates are assumed to have shifted between offline training and online training and deployment.

4.2. Choosing Safe and Conducive Actions

A safe action is any action that satisfies the safety constraint with high probability formally, any $a \in A$ s.t $\hat{c}(\vec{S}_t, a; \theta, \chi_c) \geq d$. In this context, satisfying the safety constraint with high probability means the predicted cost of a state-action transition, has an associated predictive variance (uncertainty) that is $\leq \sigma_*^2$, a user set uncertainty threshold. A conducive action is one that is likely to result in the highest possible reward (in that state) and enter a state that puts the agent closer to completing its task - in this case closer to its goal state.

Algorithm 1: The Safe and Robust Stochastic Process Algorithm

Require: $\mathcal{D}, \vec{S}_t, A, d, S_u, k, n, \sigma_*^2$,
$\chi_c = \{(\vec{S}_i, c_i) \sim \mathcal{D}\}_{i=1}^{k} \cup \{(\vec{S}_i, c_i) \in \mathcal{D} \mid \vec{S}_i = \vec{S}_t\}_{i=1}^{n}$,
$\chi_r = \{(\vec{S}_i, r_i) \sim \mathcal{D}\}_{i=1}^{k} \cup \{(\vec{S}_i, r_i) \in \mathcal{D} \mid \vec{S}_i = \vec{S}_t\}_{i=1}^{n}$,
$\hat{c}(\vec{S}_t, a; \theta, \chi_c) = c(\vec{S}_t, a) \sim St_\nu(\mu, \Sigma))$,
$\hat{r}(\vec{S}_t, a; \theta, \chi_r) = r(\vec{S}_t, a) \sim St_\nu(\mu, \Sigma))$,
$\hat{\mu}_{cost} = \{a \in A \mid \hat{f} \sim \hat{c}(\vec{S}_t, a; \theta, \chi_c)\}$,
$\hat{K}_{cost} = \{a \in A \mid \hat{\sigma^2} \sim \hat{c}(\vec{S}_t, a; \theta, \chi_c)\}$,
$\hat{\mu}_{reward} = \{a \in A \mid \hat{f} \sim \hat{r}(\vec{S}_t, a; \theta, \chi_r)\}$
for $a \in A$ **do**
  if $(\vec{S}_t, a) \in S_u$ **then**
    $\hat{\mu}_{cost}[a] = -\infty$, $\hat{K}_{cost}[a] = \infty$, $\hat{\mu}_{reward}[a] = -\infty$
$r_* = -\infty, \quad a^*_{safe} =$ **NOP**
for $a \in A$ **do**
  if $\hat{K}_{cost}[a] \leq \sigma_*^2 \wedge \hat{\mu}_{cost}[a] \geq d$ **then**
    if $\hat{\mu}_{reward}[a] > r_*$ **then**
      $r_* = \hat{\mu}_{reward}[a], \quad a^*_{safe} = a$
if $a^*_{safe} =$ ***NOP*** **then**
  $a^*_{safe} = \arg\max_{a \in A} \hat{\mu}_{reward}[a]$
$\rightarrow$ Take action a^*_{safe}, observe s_{t+1}, r_{t+1}, and c_{t+1}
$\rightarrow$ Evaluate the result of taking action a^*_{safe}
if $\hat{\mu}_{cost}[a^*_{safe}] \geq d \vee \hat{K}_{cost}[a^*_{safe}] = \arg\min_{a \in A} \hat{K}_{cost}[a]$ **then**
  if $(\vec{S}_t, a^*_{safe}) \in \mathcal{D}_{cost}$ **then**
    $\forall \mathcal{D}_{cost}[(\vec{S}_t, a^*_{safe})] \leftarrow c_t$
  else
    $\mathcal{D}_{cost} = \mathcal{D}_{cost} \cup (\vec{S}_t, a^*_{safe}, c_t)$
  $\rightarrow$ Retrain Model
$\rightarrow$ Update S_u if $c_{t+1} \leq d$ and s_{t+1} is deemed unsafe
if $c_{t+1} \leq d$ **then**
  $S_u = S_u \cup (\vec{S}_t, a^*_{safe})$

4.3. Notation

Algorithm (1) requires 8 positional arguments. $\vec{S}_t$ is the current state. A is the action space. d is the safety threshold (1), this is set based on domain knowledge or expert opinion on the specific problem. S_u is the set of all observed unsafe state-action pairs - learned during the offline training phase. S_u is expanded when the agent encounters a new unsafe state. k is the number of randomly sampled observations to be added to the prior, in our case $k = 100$. n is the maximum number of already observed states that match the current state $\vec{S}_t$ to be added to the prior, in our case $n = 10$. σ_*^2 is an empirically estimated uncertainty threshold, used to minimize the number of constraint violations during online learning and deployment.

4.4. Design and Operation

Algorithm (1) assembles two priors for the cost χ_c and reward χ_r models respectively. The priors are the union of k uniformly random samples from the offline dataset $\mathcal{D}$ and n similar state observations. This allows the agent to build a strategic prior to condition the next cost/reward prediction on giving the model the best chance of making an accurate prediction about future cost/reward. Two functions are defined $\hat{c}(\vec{S}_t, a; \theta, \chi_c)$ and $\hat{r}(\vec{S}_t, a; \theta, \chi_r)$ given as the approximations of the cost and reward functions using TPs. Using $\hat{c}(\cdot)$ and $\hat{r}(\cdot)$ we sample the TP for an expected cost and reward for taking a given action a in state $\vec{S}_t$, conditioned on the two strategically chosen priors χ_c and χ_r. $\hat{\mu}_{cost}$ is the set of sampled cost predictions for each $a \in A$, $\hat{K}_{cost}$ is the set of uncertainties associated with each prediction. $\hat{\mu}_{reward}$ is the set of sampled expected rewards for taking each $a \in A$. The algorithm then iterates through each $a \in A$, checking if any of the pairs $(s, a) \in S_u$. If a match is found, it sets the predicted cost to $-\infty$, the predicted uncertainty to $+\infty$ and the predicted reward to $-\infty$. This heavily discourages the choice of this action. The algorithm then initialises the optimal reward r_* to $-\infty$ and the current safe action to $a^*_{safe} = \textbf{NOP}$ (no operation). Next, a loop iterates through each $a \in A$, if the $\hat{K}_{cost}[a]$ of current action is below σ_*^2, and the $\hat{\mu}_{cost}[a]$ of that action is above d, then the action is considered safe with high confidence. Next, the algorithm checks if the reward associated with a_{safe} is greater than r_*, if true then the action becomes the top candidate for a^*_{safe} and $r_* = \hat{\mu}_{reward}[a]$. Finally, if a^*_{safe} is equal to **NOP**, then the algorithm was unable to find safe and conducive action with high confidence. In this case, a backup strategy is employed, whereby we choose the action with the highest expected reward. While this may potentially result in a constraint violation, it will enable the model to learn the location of a new unsafe state

and potentially increase cumulative reward and reduce cumulative cost. After taking action a^*_{safe}, the agent observes a new state $\vec{S}_{t+1}$, reward r_{t+1} and cost c_{t+1}. The agent must then evaluate the result of its action. If the predicted cost for the 'safe' action violates the safety constraint, or if the predicted uncertainty is the lowest in the set $\hat{K}_{cost}$ then we check if the state-action pair $(\vec{S}_t, a^*_{safe}) \in \mathcal{D}_{cost}$. If the state-action pair is present in the set then it updates any matching state-action pair observation with the observed cost c_t. Otherwise, it adds the state-action-cost tuple to $\mathcal{D}_{cost}$. Finally, it triggers a retrain to incorporate the updated knowledge of the cost model.

4.5. The Role of TP/GP in SARSP

In Section 4.4 the term 'model' refers to either a GP or a TP. Algorithm (1) can be used with either model, in Section 5 we compare the algorithm's performance first using two GPs and then two TPs. In Algorithm (1) we define two functions, $\hat{c}(\vec{S}_t, a; \theta, \chi_c)$ approximates the cost function from Eq.(1) and $\hat{r}(\vec{S}_t, a; \theta, \chi_r)$ approximates the reward function from Eq.(1). These functions use a stochastic process (GP or TP) to approximate the next expected cost and reward. As discussed in Section 4.1, the models are trained offline using a set of suboptimal trajectories, they are then passed to Algorithm (1) for online training and deployment. The online training phase happens only if the a distribution shift is detected. This retraining process allows the model to minimize any higher variances/uncertainties about the locations of the new, unseen unsafe states and reduces the likelihood of incurring further cost from a new unsafe state. This approach addresses the covariate shift problem by improving both the detection of new unseen unsafe states and the prediction of future costs associated with these unsafe states. This online kernel optimisation is highly beneficial to the problem as the choice of kernel parameters is a key factor in the accuracy of a TP or GP.

5. Experiments

This section conducts comprehensive experiments to demonstrate effectiveness of our proposed method.

5.1. Experimental Setup

The experimental setup closely mirrors the practical real-world example described in Section 2.4. The alignment is established as follows:

- Offline Training (Refer to Section 5.2): The agent is first trained on sub-optimal trajectories, which represent historical mission data collected from past operations in a warehouse setting. These trajectories are generated using a stochastic policy, ensuring the dataset includes both safe and unsafe behaviours, similar to real-world historical warehouse operations.

- Incorporation of Outliers (Refer to Section 5.3): To simulate rare real-world anomalies, we introduce outliers into the dataset. These outliers model real-world disruptions such as sensor errors, misplaced objects, or unexpected forklift breakdowns, as described in Section 2.4.

- Covariate Shift (Refer to Section 5.4): After offline training, we modify the environment to introduce a covariate shift, mimicking changes in warehouse layout, obstacle placement, and worker traffic patterns. This shift forces the agent to adapt to new safety constraints, much like how warehouse conditions evolve over time.

- Online Learning and Adaptation (Refer to Section 5.5): The agent is deployed in the modified environment and tasked with navigating from a randomly assigned safe starting location to a goal location, reflecting real-world goods transportation. During deployment, if the agent encounters previously unseen unsafe states, it triggers online learning to re-optimize its kernel parameters, ensuring continued safe operation.

5.2. Offline Dataset Collection and Hyperparameter Selection

The experiments use Algorithm (1) to solve a safe navigation task under a covariate shift and in the presence of outliers. Each experiment was configured with varying degrees of outlier magnitude, outlier proportion, and degree of covariate shift - these serve as experiment parameters. This ensures we measure each model's performance under an appropriate variation in the experiment parameters. Next, the offline training data $\mathcal{D}$ is collected using a stochastic control policy $\pi_{uniform}$ interacting with the environment. To clarify; the offline training data $\mathcal{D}$ is collected uniformly at random meaning at every discrete timestep $t \in T$, the next action a is chosen uniformly at random, meaning the probability of choosing any given action $a \in A$ is $P(a) = \frac{1}{|A|}$. Next, outliers are injected into the offline data, and then the TP and GP are

trained on $\mathcal{D}$. The popular adaptive moment estimation (ADAM) algorithm (an extension of stochastic gradient descent (SGD)) is utilized to minimize the log probability of each model, to optimize the kernel parameters. This approach ensures a high-fidelity comparison between the two models.

We parametrize both the TP and GP with the radial basis function (RBF) kernel as this aligns with the experimental setup of much of the related literature Wachi (2018), Bottero (2022), Sui (2015). The RBF kernel also possesses some desirable properties that make it often a universally good choice for GP/TP regression Ingo (2008). The RBF kernel can theoretically approximate any continuous function given enough data, this universality makes it versatile for many tasks without requiring domain specific tuning. The RBF also exhibits exponential decay which ensures that points further apart have minimal influence on each other, which helps the kernel capture local patterns in the data. The RBF is also widely studied across various domains and problem types, making it a reliable and proven choice.

Remark 5.1. *While acknowledging that kernel choice can influence model performance, this study primarily focuses on demonstrating the advantages of TPs over GPs in handling outliers and distribution shifts. These core properties of TPs are independent of the specific kernel used. Future research will explore the impact of different kernel functions on robustness, smoothness, and computational efficiency.*

Each experiment was run for 250 episodes and collected the following metrics: mean reward J_r, mean cost μ_{cost}, total cost Σ_{cost}, mean steps μ_{steps}, cost rate p_c (12) and task success rate TSR. Code to reproduce the experiments can be found at Hickman (2024) along with the experiment data discussed in section 6, a URL can be found in Appendix E.

$$p_c = \frac{\sum_{i=1}^{T} c_i}{\sum_{i=1}^{T} i} \tag{12}$$

Cost rate (12) is the ratio of the total cost incurred during a given period divided by the total number of safe and conducive environment interactions. Safe and conducive environment interactions are interactions that result in a completed task with 0 cost. Cost rate disregards interactions where either condition is violated.

5.3. Generating Outliers

In the Bayesian context, Andrade (2011) defines statistical outliers as surprising events contradicting the majority of available information about a parameter of interest. More technically Andrade (2011) suggests that an outlier can arise when observed data conflicts with the prior distribution, indicating significantly different parameter values than anticipated. A similar idea comes from Hawkins (1980) where an outlier is defined as "an observation which deviates so much from the other observations so as to arouse suspicions that it was generated by a different mechanism". In the absence of a real-world mechanism/process that exhibits outliers by nature, we leverage these definitions to generate synthetic outliers. This approach allows for adequate control over the outliers, ensuring we test each model's robustness (in the SRL context) effectively. Let $\mathcal{D} = \{(\vec{S}_i, y_i)\}_{i=1}^{n}$ be the set of n observations collected through successive environment interactions. Let p be the outlier proportion and m be the outlier magnitude. We uniformly randomly generate k outlier indices $V = \{i_1, i_2, \cdots i_k\}$, $k = p \cdot n$. We then transform k samples from $\mathcal{D}$ by $\vec{S}'_i = \vec{S}_i \cdot m$ and $y'_i = y_i \cdot m$. Notice the outlier indices are uniformly randomly generated, this is not an assumption about the outliers themselves, but rather the simplest way to ensure they are evenly distributed. This also ensures that no assumption is made about their origin. This method allows us to generate samples that deviate significantly from the majority of observations Shah (2014) and contradict model parameter values implied by the majority of information Andrade (2011). While this method is sound from a theoretical perspective in terms of generating synthetic outliers, we recognize that in the context of a real-world application, outliers would have to be domain-specific to the physical/real process.

5.4. The RL Task

The agent faces a classic safe navigation problem in a grid world, with an $n \times n$ discrete 2-dimensional Euclidean space represented by (x, y) coordinates. The environment, created with u uniformly randomly positioned unsafe states, requires the agent, starting at S_{start}, to navigate to the goal state S_{goal} while avoiding unsafe states. Entering an unsafe state will result in a cost of $c_t = 1$. An unsafe state does not constitute a terminal state. The agent has a maximum of $n \times n$ steps to complete the task before a terminal state is entered. The agent's start state S_{start} and its goal state S_{goal}, are generated uniformly at random at the start of each episode. The states corresponding to

unsafe states will shift between the offline training phase described in Section 5, which constitutes a covariate shift. This difference means that states that were unsafe during offline learning, do not correspond to the same physical locations as the unsafe states in the deployment environment, meaning the distribution of unsafe states has shifted.

5.5. The RL Environment

The experiment involves two key components: the agent and the custom environment designed for a safe navigation task. The agent's objective is to navigate from a starting state to a goal state while avoiding unsafe states. The reward function $r(s, a)$ is defined as the negative $-\ell_2$ norm (13), to be maximized over timesteps $t = 1, 2, 3, ..., T$. The agent must also manage a cost function $c(s, a)$, representing the positive ℓ_2 (13) norm, ensuring it remains greater than or equal to the safety threshold $d = 1$. The cost function evaluation is based on the distance from the agent to the nearest unsafe state, providing feedback at each timestep. The environment has two terminal states $\mathcal{T} \subset S, |\mathcal{T}| = 2$: reaching the goal state S_{goal} or completing the task within the time horizon $T_{max} = n \cdot n$. In this case, n is the dimension of the grid world i.e. $n = 10$ would mean a 10×10 gridworld and maximum time horizon $T_{max} = 100$ steps. After each episode, the environment resets, with the agent's new starting location assigned uniformly at random. The agent's starting location is never an unsafe or goal state. After the covariate shift occurs (after offline training and online deployment) the unsafe state locations remain constant across episodes.

$$\pm\, d(P_1, P_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{13}$$

5.6. Evaluating Safety Constraint Satisfying RL Agents:

Definition 4 provides a revised formulation of the method presented in Ray (2019) which evaluates two safety-constraint satisfying agents. Our revised approach is as follows: let two constraint-satisfying agents be A_1 and A_2. Both agents have been trained for an equal number of environment interactions. A_1 dominates A_2 ($A_1 \succ A_2$) if it strictly improves on either mean return J_r, cost rate p_c (12) or task success rate TSR and performs at least as well on the others or to a task-specific performance threshold $a \in \mathbb{R}_+, a \in [1, 100]$.

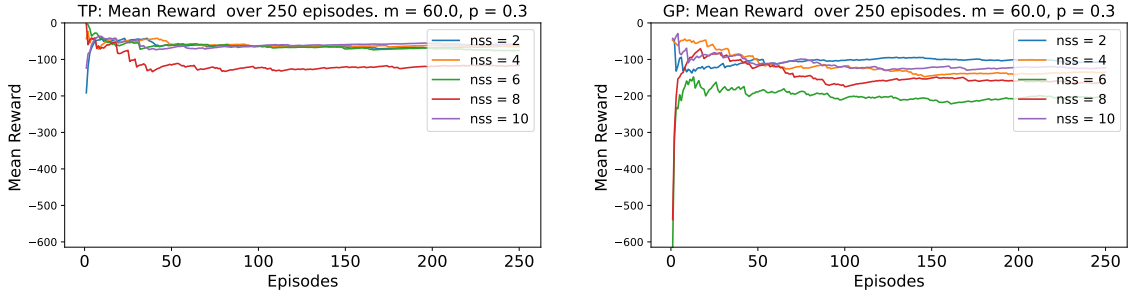
Definition 4. *Comparing 2 safety constraint satisfying agents.*

$$
\begin{aligned}
A_1 \succ A_2 \equiv \\
&(J_r(A_1) \geq J_r(A_2) \wedge p_c(A_1) < p_c(A_2)) \\
&\vee (TSR(A_1) \geq TSR(A_2) \wedge p_c(A_1) < p_c(A_2)) \\
&\vee (TSR(A_1) \geq a \wedge p_c(A_1) < p_c(A_2))
\end{aligned}
$$

6. Data Analysis

In this section, we will present the experimental results. 90 experiments were run under varying degrees of outlier extremity and covariate shift configurations, here we present our findings. The following subsections examine model performance metrics directly affected by outliers only. Subsections 6.1, 6.4, 6.5. The following subsections examine model performance metrics directly affected by covariate shift. Subsections 6.2, 6.3, 6.6. The following sections examine how well each model balances minimising cost while maximising reward under covariate shift and outliers. Subsections 6.7, 6.9.

6.1. Mean Reward Analysis



(a) TP Mean Reward

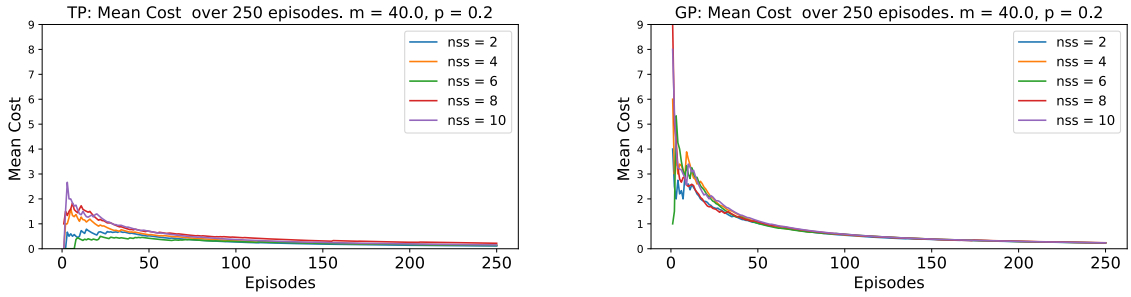
(b) GP Mean Reward

Figure 2: Mean Reward Comparison

Figure 2(a) shows the TP's mean reward from experiments where the outlier magnitude was $m = 60$, the outlier proportion was $p = 0.3$ and the number of switch states (degree of covariate shift) varies from $nss = 2 - 10$. The TP consistently maintains a mean reward of ~ -50 across the nss range, except for $nss = 8$, where it drops to ~ -150. The TP typically

maintains a mean reward of ~ -50 across the degrees of covariate shift, except for $nss = 8$ where μ_{reward} drops to around ~ -150. This drop in μ_{reward} is likely to be some form of anomaly as other recorded metrics from that experiment ($m = 60, p = 0.3, nss = 8$) do not appear to have unexpected readings. In contrast, the GP exhibits less consistent performance, with an overall lower mean reward as shown in figure 2(b). Figure 2(b) shows the worst performing experiment to be $nss = 6$ where the GP maintained a μ_{reward} of $\sim -200 \pm 20$, with more severe covariate shifts ($nss = 8, 10$) showing better performance. The GP's best performance was $nss = 2$, which would fit the general expectation that when the covariate shift was at its least severe, the model would be less perturbed. The GP's best consistent μ_{reward} was approximately $\sim -100 \pm 10$, which compared with the TP's best μ_{reward} of ~ -50, is a poor result. This can likely be attributed to the differences between each model's method for approximating posterior predictive variance (uncertainty), where the TP's more nuanced approach allows for more flexibility and adaptability.

6.2. Mean Cost Analysis



(a) TP Mean Cost

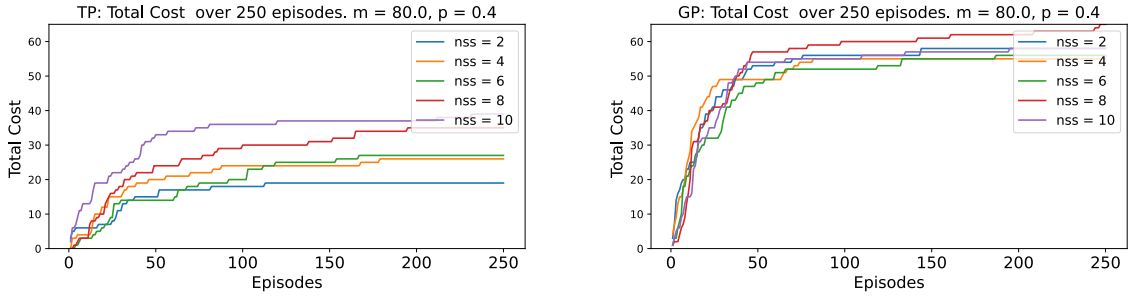
(b) GP Mean Cost

Figure 3: Mean Cost Comparison

Figure 3(a) shows the TP's mean cost from experiments where $m = 40$, $p = 0.2$ and $nss = 2 - 10$. The graph shows the TP's mean cost reaching a maximum of ~ 2.8 with $nss = 10$ and in all cases converging to $\sim 0.35 \pm 0.15$. The graph also shows the TP completing its online learning process within 50 episodes. During the start of each experiment, the TP incurs some initial cost as it adapts to the new distribution of covariates, but in all cases, the TP's mean cost converges. Looking at figure 3(b) we see the mean cost rising

substantially higher than that of the TP, reaching a maximum of 9 in some cases. While the GP's mean cost eventually converges to a reasonable number, this doesn't happen before the 100th episode, demonstrating the GP takes longer to learn the new distribution of unsafe states. This extra time taken to learn the new distribution of covariates can likely be attributed to 1) the outliers perturbing learning. 2) The GP's method of computing posterior predictive variance does not take into account the variability of uncertainty across the input space. As is expected the worst performing experiments for both the TP and GP were when $nss = 8, 10$, however, the TP was far better at minimizing average cost in these extreme cases.

6.3. Total Cost Analysis



(a) TP Total Cost

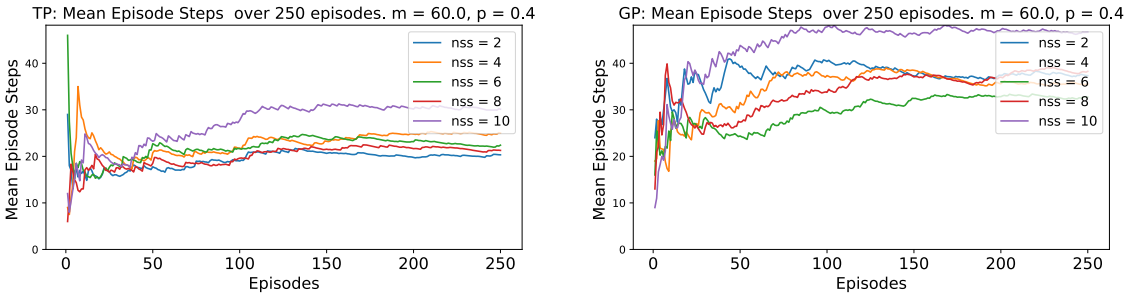
(b) GP Total Cost

Figure 4: Total Cost Comparison

Figure 4(a) shows the TP's Σ_{cost} for experiments where $m = 80, p = 0.4$ and $nss = 2 - 10$. The TP's total cost rises fairly gradually in line with the increases in nss, and remains < 40 across all experiments. This is an encouraging result, especially given the cost plateaus shown in all cases of nss, where the TP eventually learns the new distribution of covariates and ceases to enter unsafe states. The plateaus essentially represent convergence during online learning. Contrast this with the results shown in figure 4(b) where the GP's total cost rises far more rapidly and to higher levels with a maximum of ~ 65 achieved when $nss = 8$. The GP also exhibits fewer consistent cost plateaus with cost being incurred up until the final episodes, suggesting the new distribution of covariates is not effectively learned by the GP. This initial escalation to higher cost levels than the TP, coupled with

the lack of plateaus, demonstrates that the TP is far better suited to these sorts of problems.

6.4. Mean Steps Analysis



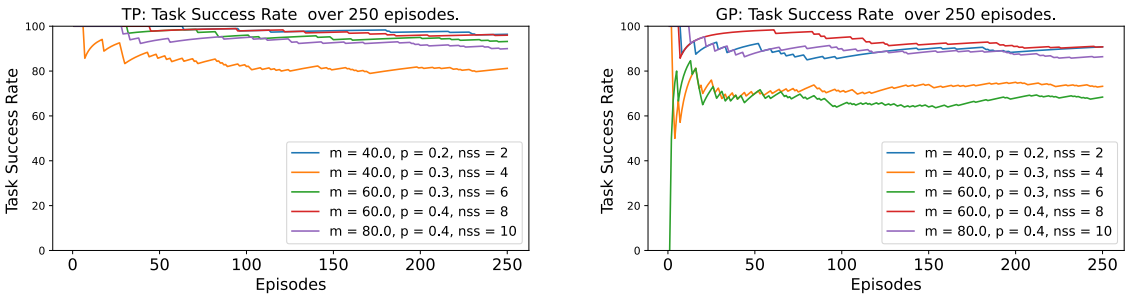
(a) TP Mean Steps

(b) GP Mean Steps

Figure 5: Mean Steps Comparison

Figure 5(a) shows the mean number of steps the TP-driven agent took to complete the task across all 250 episodes. The mean steps metric quantifies how efficiently the model can solve the task, i.e. the fewer the steps, the more efficient the solution. Comparing figure 5(a) to figure 5(b) we can see the TP is more efficient on average in every case. A small exception occurs with $nss = 6$ near the start of the experiment, but the spike lasted < 5 episodes. There is also a marginally higher mean-steps reading for the TP when $nss = 10$, with μ_{steps} reaching ~ 28. However, given the severity of the outlier magnitude and proportion ($m = 60, p = 0.4$) coupled with the most extreme degree of covariate shift ($nss = 10$), this small dip in task efficiency is to be expected. The results for the GP shown in figure 5(b) tell a very different story. The GP exhibits far more variability in its efficiency, with far less consistency than that of TP. For example when $nss = 2$ (least extreme) $\mu_{steps} \sim= 35 \pm 3$, and when $nss = 6$ (medium extremity) $\mu_{steps} \sim= 28 \pm 4$. As is expected the worst performing GP experiment was $nss = 10$. The biggest drawback of the GP's results are their lack of consistency, which in safety-critical settings, is a very desirable property. In contrast, the TP achieves far more consistent results, with less variability and requires fewer steps to complete tasks/episodes. This demonstrates that the outliers and shifts in covariates present significant challenges to GP-driven agents in this context.

6.5. Task Success Rate Analysis



(a) TP Task Success Rate

(b) GP Task Success Rate

Figure 6: Task Success Rate Comparison

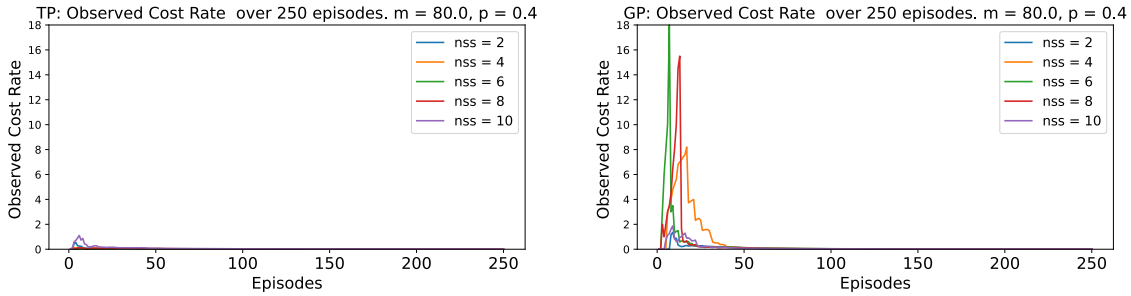
Figure 6(a) shows the TP's task success rate under varying degrees of outliers and covariate shifts. The TP's performance remains fairly consistent across the varying degree of outliers and covariate shift with a $TSR > 90\%$ for almost all experiments, except for the experiment where $m = 40, p = 0.3, nss = 4$ where the $TSR \sim= 80\%$. Figure 6(b) shows the GP's TSR across the same varying degree of outliers and covariate shift and the graph shows far more variation in performance with two experiments seeing a $TSR \sim= 70\%$.

6.6. Cost Rate Analysis

Figure 7 shows the comparison between the cost rates of the TP and GP. The cost rate equation was formalised in (12) and is essentially the ratio of the total cost incurred over the total number of safe and conducive environment interactions. Figure 7(a) shows the cost rate achieved by the TP under $m = 80, p = 0.4$ and $nss = 2 - 10$. Figures 7(a) and 7(b) show both models converging to a $p_c =\sim 0$, however, the TP maintains a cost rate of < 1 across all variations of nss, whereas the GP reaches a cost rate of 18 in the worst case. This demonstrates the superiority of the TP-based approach in minimising cost and maximising reward and task success under outliers and covariate shifts.

6.7. Macro Analysis of Mean Cost & Mean Reward

Figure 8(a) shows the average cost incurred by the TP and GP-driven agents across all variations of outlier magnitude and outlier proportion plotted

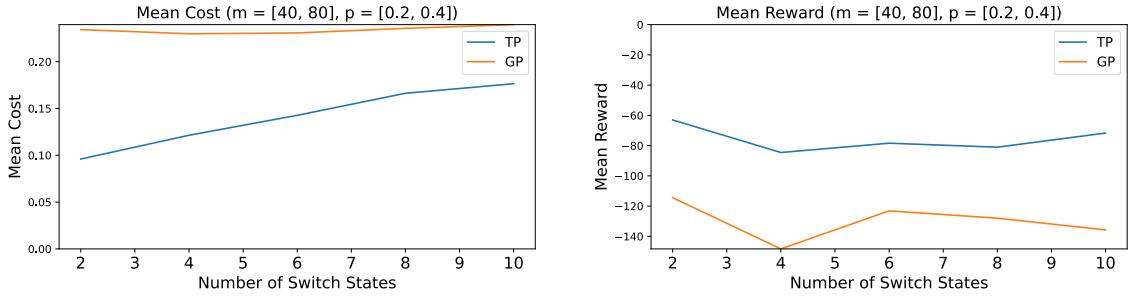


(a) TP Cost Rate

(b) GP Cost Rate

Figure 7: Cost Rate Comparison

against the number of switch states (nss). In the case of the TP, the graph shows a steady linear increase in mean cost inline with nss and a gradual slow down at $nss = 8 - 10$. The GP demonstrates a consistent average cost incurred with very little variation w.r.t the nss. Figure 8(b) shows the average reward achieved across the full range of outliers for both models plotted against the degree of covariate shift. The TP shows a fairly consistent mean reward between $-60, -80$ with a small dip when $nss = 4$. The GP shows marginally more variation in its mean reward with results being < -120 for the vast majority of measurements, however, of note is the corresponding dip in mean reward at $nss = 4$ which we can see in the TPs results also. Figure 9 shows the average total cost incurred by both models across all

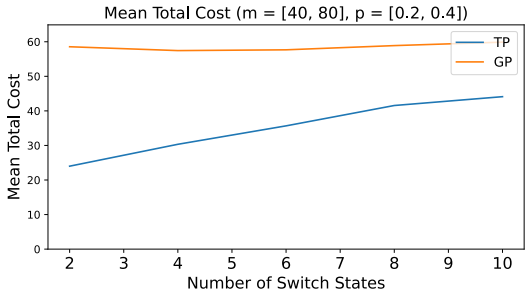


(a) TP vs GP Mean Cost Across all Experiments

(b) TP vs GP Mean Reward Across all Experiments

Figure 8: TP vs GP Mean Reward & Mean Cost

outlier variations, plotted against the number of switch states (nss). As expected the TP sees a gradual linear increase in average total cost with nss and a gradual slow down as $nss \rightarrow 10$. The TP's total cost is on average always below 40. The GP sees a consistently high mean total cost of ~ 56 with very little variation w.r.t increases in nss.



(a) TP Mean Cost

Figure 9: TP vs GP Mean Total Cost Across all Experiments

6.8. Macro Analysis of 6 Experiments

Table 1 shows results from 6 experiments comparing the TP and the GP's performance. The table details the mean cost, mean reward, total cost, mean steps, and a final result using the preference criterion described in definition (4) with % difference in the Σ_{cost}. Starting with mean cost the TP achieves a consistently lower μ_{cost} than the GP across all 6 listed experiments. As expected the lowest μ_{cost} from the TP was experiment 1 where it achieved $\mu_{cost} = 0.112$ when outliers and degree of covariate shift were at their least extreme. Interestingly, the GP achieves its lowest μ_{cost} of 0.224 when outliers were at $(m = 60, = 0.4)$ and covariate shift was $nss = 8$. The results for mean reward tell a similar story with the TP being consistently better across all 6 listed experiments. In the case of total cost the GP obtained some interesting results in that there was very little variation in the Σ_{cost} achieved by the GP with all 6 listed experiments reporting $\Sigma_{cost} = [56, 59]$. The task success rate results are fairly similar between the two models, with the TP being the clear winner, but the GP is not far behind save for experiment 3 where the TP achieved $TSR = 93.2\%$ and the GP $TSR = 68.4\%$. Mean steps are the efficiency metric quantifying how fast the model solved the episode/task. The TP had a consistently lower μ_{steps} score than the GP across all 6 listed

experiments, with very good performance in experiment 5 where the TP managed a $\mu_{steps} = 21.28$. In summary, the TP outperforms the GP across almost every metric in all 6 experiments, with at least a 29% reduction in total cost in all cases.

Model	μ_{cost}	μ_{reward}	Σ_{cost}	TSR	μ_{steps}	m	p	nss	Result
A_1	0.112	-66.418	28	96.4%	23.412	40.0	0.2	2	$A_1 \succ A_2$
A_2	0.232	-106.348	58	90.8%	34.312	40.0	0.2	2	69.8%
A_1	0.124	-104.342	31	81.2%	31.188	80.0	0.2	4	$A_1 \succ A_2$
A_2	0.232	-173.677	58	73.2%	47.488	80.0	0.2	4	60.7%
A_1	0.176	-75.311	44	93.2%	25.176	60.0	0.3	6	$A_1 \succ A_2$
A_2	0.236	-200.553	59	68.4%	49.728	60.0	0.3	6	29.1%
A_1	0.152	-116.432	38	82.8%	34.124	60.0	0.3	8	$A_1 \succ A_2$
A_2	0.224	-163.626	56	79.6%	41.408	60.0	0.3	8	38.3%
A_1	0.168	-55.973	42	96.0%	21.28	60.0	0.4	8	$A_1 \succ A_2$
A_2	0.236	-131.454	59	90.8%	38.264	60.0	0.4	8	33.7%
A_1	0.156	-76.408	39	90.0%	26.396	80.0	0.4	10	$A_1 \succ A_2$
A_2	0.232	-140.015	58	86.4%	37.772	80.0	0.4	10	39.2%

Table 1: Data from 6 of the experiments. **Note:** A_1 = TP, A_2 = GP

6.9. Macro Analysis of Central Tendency

We examine the results of all 90 experiments as a whole, providing a macro view of the differences in performance between the TP and the GP. This analysis will focus on the central tendency of the data, looking at the mean, median and mode of the results as a whole. The metrics cover all configurations of outlier magnitude, outlier proportion, and degree of covariate shift. In the case of the average metrics (Table 2) We also report the coefficient of variation (CV) which describes the relative variability that standardizes the dispersion of the data relative to its mean. This is a useful measure to understand how much variation there is in the average metrics w.r.t the central tendency of the average metrics. The CV is computed using $\left(\frac{\sigma}{\mu}\right) \times 100$ and is interpreted as standard deviation σ as a percentage of the mean μ. Table 2 details the average mean reward, mean cost, total cost, task success rate, mean steps and cost rate achieved by both models across all 90 experiments conducted (45 for the TP, 45 for the GP). Table 3 details the median metrics, and table 4 details the mode metrics. Table 2 shows the TP outperforming

Metric	TP	GP	% ↑↓	CV_{TP}	CV_{GP}
μ_{reward}	-75.755	-129.924	41.69% ↑	31.8%	25.99%
μ_{cost}	0.141	0.234	39.93% ↓	28.66%	3.99%
Σ_{cost}	35.133	58.489	39.93% ↓	28.66%	3.99%
$\mathcal{TSR}$	91.644%	86.062%	6.49% ↑	6.58%	9.61%
μ_{steps}	25.201	37.513	32.82% ↓	23.33%	17.6%
p_c	0.01	0.012	16.32% ↓	34.73%	15.02%

Table 2: Mean Metrics

the GP on average in every reported statistic. Starting with the μ_{reward} the TP achieves on average a 41.69% higher mean reward than the GP, and if we look at the TP and GP's CV there is very little difference in the level of variation between the TP and GP's μ_{reward}. Next, the TP's μ_{cost} is on average 39.93% lower across all experiments. The TP's CV is significantly higher at 28.66% than the GP's at 3.99%, which shows the TP responds to the fluctuating experiment parameters, producing more variation on average in its mean cost results than the GP. The reason for this is that the GP is poorly equipped to adapt to the changing experiment parameters meaning it will likely produce consistently poor results as is reflected in its CV. The total cost results reflect the same relative difference as is shown in the mean cost results, but the real-valued difference on average is stark. The TP on average makes 35 constraint violations per episode (including learning) and the GP makes 58 constraint violations on average per episode. There is little difference in the task success rate on average, with the TP coming in at 6.49% better at completing the task than the GP, but coupled with its cost and reward results, the GP's overall performance was poor. The TP's CV of 6.58% reflects the model being very consistent in its task success rate, with the GP's CV reflecting the same, but at a marginally higher 9.61%. The TP managed to achieve 32.82% fewer steps on average compared with the GP which achieved an average mean step count of 37.513. The TP's CV for the average mean steps was marginally higher at 23.33% than that of the GP at 17.6% but this is to be expected given that we would expect more adaptive behaviour from the TP compared with the GP. Finally, the average cost rate achieved by the TP was 16.32% lower than the GP's average cost rate, reflecting the results we saw in mean reward, mean cost and total cost.

The TP's CV is almost double that of the GP, implying the GP's behaviour was far more consistent (consistently poor) under outliers and covariate shifts than that of the TP.

Metric	TP	GP	% ↑↓
μ_{reward}	-68.396	-122.887	44.34% ↑
μ_{cost}	0.144	0.232	37.93% ↓
Σ_{cost}	36.0	58.0	37.93% ↓
$\mathcal{TSR}$	93.6%	90.4%	3.54% ↑
μ_{steps}	23.86	36.08	33.87% ↓
p_c	0.01	0.011	9.09% ↓

Table 3: Median Metrics

Table 3 reflects the median behaviour of both models across all experiments. The median behaviour can be interpreted as the central tendency of the data independent of any extreme results on either end of the scale. The median value provides a typical result without the influence of both extremely good performance or extremely bad performance. The TP achieves a 44.34% higher reward in the median with -68.396 when compared to the GP which achieved a median average reward of -122.887. The TP maintains a 37.93% lower median average cost of 0.144 when compared with the GP's median average cost of 0.232. The relative difference of the total cost and mean cost are the same, with the same improvement from the TP. The TP achieved a median total cost of 36 and the GP achieved a median total cost of 58. The median percentage increase of the TP's task success rate over the GP task success rate is almost half of the mean task success rate improvement, which suggests that either the GP had very good performance on a small number of experiments or the TP had poorer performance on a small number of experiments. In either case, the TP has a 3.54% better $\mathcal{TSR}$ in the median and a 6.58% higher $\mathcal{TSR}$ in the average metrics. The efficiency results in the median are similar to the mean, with the TP maintaining 33.87% fewer steps per episode than the GP. The TP achieves a 9.09% lower cost rate than the GP in the median metric.

Metric	TP	GP	% ↑↓
μ_{reward}	-145.273	-200.553	27.56% ↑
μ_{cost}	0.144	0.232	37.93% ↓
Σ_{cost}	36	58	37.93% ↓
$\mathcal{TSR}$	96.0%	90.4%	6.19% ↑
μ_{steps}	15.104	25.044	39.69% ↓
p_c	0.01	0.011	9.09% ↓

Table 4: Mode Metrics

Table 4 details the mode central tendency, i.e. the most frequently occurring behaviour from each model. Starting with the mode mean reward, we see a 27.56% higher μ_{reward} from the TP over the GP. When comparing the mean, median and mode μ_{reward} there is an obvious increase in the mode's results over the other two descriptors. This is because the vast majority of experiments were on the more extreme ends of the configuration spectrum (higher outlier proportion, magnitude and covariate shift). The TP achieves a 37.93% lower mean cost in the mode than the GP, in keeping with the results from the mean and median metrics. The TP and GP total cost results for the mode metric are very similar to that of the mean and median, implying consistency and consensus across the descriptors. The TP achieves a 6.19% better $\mathcal{TSR}$ than the GP in the mode. Interestingly the TP achieves its best improvement in efficiency over the GP in the mode metrics with the TP achieving a 39.69% lower μ_{steps} score than the GP. Finally, the TP maintained a 9.09% lower cost rate than the GP in keeping with results in the median metrics.

7. Conclusion

This study addresses the covariate shift problem in SRL in the presence of outliers. We discuss approaches in both offline and online SRL, and highlight their shortcomings w.r.t covariate shift and outliers. Leveraging Karamata's theory, we demonstrate the TP's superior ability to handle outliers while retaining the attractive mathematical properties of the GP. Findings reveal the susceptibility of GPs to outliers/extreme values, consistent with existing literature and supported by Ailton (2022). We analyze and discuss the differences in posterior predictive variance computation between the TP

and GP, concluding that the TP is far more suitable for this problem. We draw upon ideas from both offline and online SRL literature and propose a hybrid offline-online approach using TPs to address the covariate shift problem under outliers. We conduct extensive experiments to measure the empirical performance of our approach with the TP compared to the GP. Our findings demonstrate that on average the TP is a far better model when dealing with covariate shift and outliers in an SRL context, with a 41.69% improvement in mean reward, a 39.93% improvement in mean cost, and a 32.82% improvement in mean steps (efficiency). Finally, we offer a preference criterion (4) to evaluate two SRL agents solving the same CMDP under covariate shift and outliers. **Concluding Remark:** While this work focuses on covariate shift and outliers in SRL, the notion of using a TP applies to other open challenges in SRL such as the online safe exploration problem, therefore we highlight this as a future direction for this work.

Appendix A. Formal Definition of a Gaussian Process

Definition 5. *Let $\mathcal{T}$ be an abstract set and $\{X(t); t \in \mathcal{T}\}$ be a stochastic process. We call $\{X(t); t \in \mathcal{T}\}$ a Gaussian process if for every $n = 1, 2, 3..$ and every finite subset $\{t_1, ..., t_n\}$ of T, the random vector $(X(t_1, ..., X(t_n))$ has multivariate Gaussian distribution. Every Gaussian process is described uniquely by its two parameters, the mean and covariance function given respectively by:*

$$\mu(t) = \mathbb{E}[X(t)], \qquad t \in \mathcal{T} \tag{A.1}$$

and

$$\Gamma(t_1, t_2) = \mathbb{E}[\{X(t_1) - \mu(t_1)\}\{X(t_2) - \mu(t_2)\}], \qquad t_i \in \mathcal{T} \tag{A.2}$$

The covariance function is positive definite (PD), i.e. for every $n = 1, 2, 3, ...$ real numbers $\alpha_1, ..., \alpha_n$ and elements $t_1, ..., t_n$ in $\mathcal{T}$ the following holds,

$$\sum_{i,j=1}^{n} \alpha_i \alpha_j \Gamma(t_i, t_j) \geq 0. \tag{A.3}$$

Remark: The gamma function $\Gamma(\cdot, \cdot)$ given here refers to a positive definite kernel function. This definition follows Karlin (1968).

Appendix B. Formal Definition of a Student-t's Process

Definition 6. *Let $\mathcal{T}$ be an abstract set and $\{X(t); t \in \mathcal{T}\}$ be a stochastic process. We call $\{X(t); t \in \mathcal{T}\}$ a Student's t-process if for every $n = 1, 2, 3..$ and every finite subset $\{t_1, ..., t_n\}$ of T, the random vector $(X(t_1, ..., X(t_n))$ has multivariate Student's t-distribution. Every Student's t-process is described uniquely by its three parameters, the mean and covariance function, and the degrees of freedom parameter ν given respectively by:*

$$\mu(t) = \mathbb{E}[X(t)], \qquad t \in \mathcal{T} \tag{B.1}$$

and

$$\Gamma(t_1, t_2) = \mathbb{E}[\{X(t_1) - \mu(t_1)\}\{X(t_2) - \mu(t_2)\}], \qquad t_i \in \mathcal{T} \tag{B.2}$$

and the degrees of freedom parameter ν where,

$$\nu \geq 3, \qquad \nu \in \mathbb{R}_+ \tag{B.3}$$

The covariance function is positive definite (PD), i.e. for every $n = 1, 2, 3, ...$ real numbers $\alpha_1, ..., \alpha_n$ and elements $t_1, ..., t_n$ in $\mathcal{T}$ the following holds,

$$\sum_{i,j=1}^{n} \alpha_i \alpha_j \Gamma(t_i, t_j) \geq 0. \tag{B.4}$$

Remark: The gamma function $\Gamma(\cdot, \cdot)$ given here refers to a positive definite kernel function. This definition follows Karlin (1968).

Appendix C. Formal Definition of Student-t's Probability Density Function

Definition 7. *A random variable X follows a Student's t-distribution with ν degrees of freedom if its probability density is given by,*

$$f(x) = \frac{\Gamma(\frac{\nu+1}{2})}{\sqrt{\nu\pi}\Gamma(\frac{\nu}{2})} \left(1 + \frac{x^2}{\nu}\right)^{-\frac{\nu+1}{2}}, \qquad \forall x \in \mathbb{R} \tag{C.1}$$

Remark: The gamma function $\Gamma(\cdot)$ given here refers to the conventional gamma function which is a generalization of the factorial functions to complex and real numbers.

Appendix D. Formal Definition of Standard Gaussian Probability Density Function

Definition 8. *The standard Gaussian density,* $\mathcal{N}(0,1)$ *where the density is parameterized with mean* $\mu = 0$ *and variance* $\sigma = 1$ *and simplified.* (D.1)

$$f(x) = \frac{1}{\sqrt{2\pi}} \exp^{-\frac{x^2}{2}} \tag{D.1}$$

Appendix E. Supplementary Material Repository

https://github.com/onetimedev/safe-robust-srl-tp.git

References

Turchetta, M., Berkenkamp, F. & Krause, A. Safe Exploration in Finite Markov Decision Processes with Gaussian Processes. *Advances In Neural Information Processing Systems.* **29** (2016)

Shah, A., Wilson, A. & Ghahramani, Z. Student-t Processes as Alternatives to Gaussian Processes. *Proceedings Of The Seventeenth International Conference On Artificial Intelligence And Statistics.* **33** pp. 877-885 (2014,4), https://proceedings.mlr.press/v33/shah14.html

Tracey, B. & Wolpert, D. Upgrading from Gaussian Processes to Student's-T Processes. *2018 AIAA Non-Deterministic Approaches Conference.* (2018), https://doi.org/10.2514

Sui, Y., Gotovos, A., Burdick, J. & Krause, A. Safe Exploration for Optimization with Gaussian Processes. *Proceedings Of The 32nd International Conference On Machine Learning.* **37** pp. 997-1005 (2015), https://proceedings.mlr.press/v37/sui15.html

Sutton, R. & Barto, A. Reinforcement learning: An introduction. (MIT press,2018)

Andrade, J. On the robustness to outliers of the Student-t process. *Scandinavian Journal Of Statistics.* **n/a** (2022), https://onlinelibrary.wiley.com/doi/abs/10.1111/sjos.12611

Karamata, J. Sur un mode de croissance régulière des fonctions. (Inst. de arte graf. Ardealul,1930), https://books.google.co.uk/books?id=5FC8XwAACAAJ

Rojo, J. Heavy-tailed densities. *WIREs Computational Statistics*. **5**, 30-40 (2013)

Andrade, J. & O'Hagan, A. Bayesian Robustness Modelling of Location and Scale Parameters. *Scandinavian Journal Of Statistics*. (2011)

Altman, E. Constrained Markov Decision Processes. (1999), https://api.semanticscholar.org/CorpusID:14906227

Ray. A, A. Benchmarking Safe Exploration in Deep Reinforcement Learning. (2019)

Steinwart, I. & Christmann, A. Support Vector Machines. (Springer Publishing Company, Incorporated,2008)

Rasmussen, C. & Williams, C. Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning). (The MIT Press,2005)

Gu, S., Yang, L., Du, Y., Chen, G., Walter, F., Wang, J., Yang, Y. & Knoll, A. A Review of Safe Reinforcement Learning: Methods, Theory and Applications. (2022,5)

Hawkins, D. Identification of Outliers. (Chapman,1980), https://books.google.co.uk/books?id=fb0OAAAAQAAJ

Quionero-Candela, J., Sugiyama, M., Schwaighofer, A. & Lawrence, N. Dataset Shift in Machine Learning. (The MIT Press,2009)

Li, Q., Meng, F., Tian, Y. & Wang, X. Bayesian Optimization Based on Student's T Process for Microstrip Antenna Design. *The Applied Computational Electromagnetics Society Journal (ACES)*. **37**, 856-866 (2023), https://journals.riverpublishers.com/index.php/ACES/article/view/15047

Wachi, A., Kajino, H. & Munawar, A. Safe Exploration in Markov Decision Processes with Time-Variant Safety using Spatio-Temporal Gaussian Process. (2018)

Schreiter, J., Nguyen-Tuong, D., Eberts, M., Bischoff, B., Markert, H. & Toussaint, M. Safe Exploration for Active Learning with Gaussian Processes. *Machine Learning And Knowledge Discovery In Databases*. (2015)

Budd, M., Lacerda, B., Duckworth, P., West, A., Lennox, B. & Hawes, N. Markov Decision Processes with Unknown State Feature Values for Safe Exploration using Gaussian Processes. *2020 IEEE/RSJ International Conference On Intelligent Robots And Systems (IROS)*. (2020)

Bottero, A., Luis, C., Vinogradska, J., Berkenkamp, F. & Peters, J. Information-Theoretic Safe Exploration with Gaussian Processes. *Advances In Neural Information Processing Systems*. pp. 30707-30719 (2022)

Zheng, Y., Li, J., Yu, D., Yang, Y., Li, S., Zhan, X. & Liu, J. Safe Offline Reinforcement Learning with Feasibility-Guided Diffusion Model. (2024)

Guan, J., Chen, G., Ji, J., Yang, L., Zhou, A., Li, Z. & Jiang, C. VOCE: Variational Optimization with Conservative Estimation for Offline Safe Reinforcement Learning. *Advances In Neural Information Processing Systems*. pp. 33758-33780 (2023)

Cao, C., Yan, Z., Lu, R., Tan, J. & Wang, X. Offline Goal-Conditioned Reinforcement Learning for Safety-Critical Tasks with Recovery Policy. (2024)

Li, A., Misra, D., Kolobov, A. & Cheng, C. Survival Instinct in Offline Reinforcement Learning. *Advances In Neural Information Processing Systems*. pp. 62062-62120 (2023)

Levine, S., Kumar, A., Tucker, G. & Fu, J. Offline Reinforcement Learning: Tutorial, Review, and Perspectives on Open Problems. *ArXiv*. **abs/2005.01643** (2020), https://api.semanticscholar.org/CorpusID:218486979

X. Hickman, Y. Lu, and D. Prince, “Supplementary Material for "Hybrid Safe Reinforcement Learning: Tackling Distribution Shift and Outliers with the Student-t’s Process",” Mendeley Data, V1, 2024, doi: 10.17632/hxtnfmwkyr.1.

Jylänki, P., Vanhatalo, J. & Vehtari, A. Robust Gaussian Process Regression with a Student-t Likelihood. *Journal Of Machine Learning Research*. **12**, 3227-3257 (2011), http://jmlr.org/papers/v12/jylanki11a.html

Yu, M., Yang, Z., Kolar, M. & Wang, Z. Convergent Policy Optimization for Safe Reinforcement Learning. (2019), https://arxiv.org/abs/1910.12156

Zhou, J., Yan, L. & Yang, K. Enhancing System-Level Safety in Mixed-Autonomy Platoon via Safe Reinforcement Learning. *IEEE Transactions On Intelligent Vehicles*. pp. 1-13 (2024), http://dx.doi.org/10.1109/TIV.2024.3373512

Shao, Y., Chen, C., Kousik, S. & Vasudevan, R. Reachability-based Trajectory Safeguard (RTS): A Safe and Fast Reinforcement Learning Safety Layer for Continuous Control. (2021), https://arxiv.org/abs/2011.08421

Y.Zhang, X.Liang, D.Li, S.S.Ge, B.Gao, H.Chen, and T.H.Lee, "Barrier Lyapunov Function-Based Safe Reinforcement Learning for Autonomous Vehicles With Optimized Backstepping," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 2, pp. 2066–2080, Feb. 2024, doi: 10.1109/TNNLS.2022.3186528.

Samuel Karlin and Howard M. Taylor, *A First Course on Stochastic Processes*, 1968. Available at: https://api.semanticscholar.org/CorpusID:116197521.

Christian Rover, *Degrees-of-freedom estimation in the Student-t noise model*, 2011. https://dcc.ligo.org/public/0070/T1100497/001/DofEstimation.pdf.