

Global Safe Sequential Learning via Efficient Knowledge Transfer

Anonymous authors

Paper under double-blind review

Abstract

Sequential learning methods, such as active learning and Bayesian optimization, aim to select the most informative data to learn about a task for task learning. In many medical or engineering applications, the applications, however, data selection is constrained by a priori-unknown safety conditions, motivating the development of safe learning approaches. A promising line of safe learning methods utilize Gaussian processes (GPs) to model the safety probability and perform data selection in-uses Gaussian processes to model safety conditions, restricting data selection to areas with high safety confidence. However, accurate safety modeling requires prior knowledge or consumes data. In addition, these methods are limited to local exploration around an initial seed dataset, as safety confidence centers around the given observations, which leads to local exploration. As transferable source knowledge is often available in safety-critical experiments observed data points. As a consequence, task exploration is slowed down and safe regions disconnected from the initial seed dataset remain unexplored. In this paper, we propose to consider transfer-safe safe transfer sequential learning to accelerate the learning of safety. We further consider a pre-computation of source components to reduce the additional computational load that is introduced by incorporating source data. In this paper, we theoretically analyze the maximum-explorable safe regionsof conventional safe learning methods. Furthermore, we empirically demonstrate that our approach1) learns a task task learning and to expand the explorable safe region. By leveraging abundant offline data from a related source task, our approach guides exploration in the target task more effectively. We also provide a theoretical analysis to explain why single-task method cannot cope with disconnected regions. Finally, we introduce a computationally efficient approximation of our method that reduces runtime through pre-computations. Our experiments demonstrate that this approach, compared to state-of-the-art methods, learns tasks with lower data consumption, 2) globally explores and enhances global exploration across multiple disjoint safe regionsunder guidance of the source knowledge, and 3) operates with computation comparable to conventional safe learning methods, while maintaining comparable computational efficiency.

1 Introduction

Despite the great success of machine learning, accessing data is a non-trivial task, acquiring data remains a significant challenge. One prominent approach is to consider experimental design (Lindley, 1956; Chaloner & Verdinelli, 1995; Brochu et al., 2010). In particular, active learning (AL) (Krause et al., 2008; Kumar & Gupta, 2020) and Bayesian optimization (BO) (Brochu et al., 2010; Snoek et al., 2012) resort to a sequential data selection process in which the most informative data points are incrementally added to the dataset. The methods initiate begin with a small amount of data dataset, iteratively compute an acquisition function to prioritize data points for querying, select new data based on this information, query new data according to the acquisition score, receive observations from the oracle, and update the belief. This process is repeated until the learning goal is achieved, or until the acquisition budget is exhausted. These learning algorithms often utilize Gaussian processes (GPs, Rasmussen & Williams (2006)) as surrogate models for the acquisition computation (Krause et al., 2008; Brochu et al., 2010).

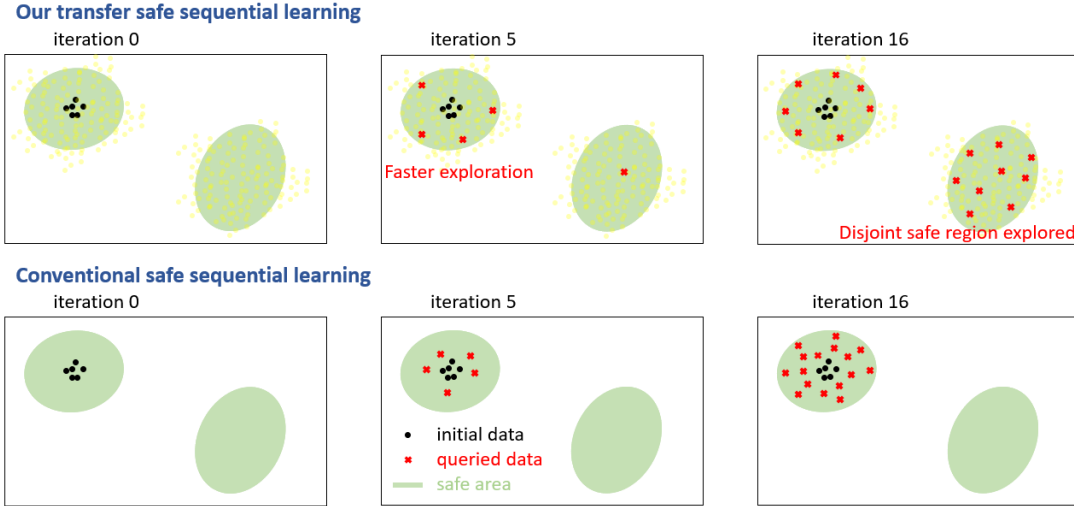


Figure 1: Illustration: ~~safe~~ Safe sequential learning with transfer (top) and conventional (bottom) learning. The light yellow data points represent source data. The main benefit of transfer learning is to accelerate exploration and identify larger and potentially disjoint safe regions by leveraging the source data.

In many applications, such as spinal cord stimulation (Harkema et al., 2011) and robotic learning (Berkenkamp et al., 2016; Baumann et al., 2021), ~~the algorithms must respect some a-priori unknown safety concerns.~~ data acquisition can introduce safety risks due to unknown safety constraints in the input space. For instance, tuning a robot controller requires testing various controller parameters; however, certain parameter settings may lead to unsafe behaviors, such as a drone flying at high speed toward a human—an issue only observed after executing the controller (Berkenkamp et al., 2016). This scenario highlights the need for a safe learning approach that selects data points being safe and maximally informative within safety limits. One effective approach ~~of performing~~ to safe learning is to model ~~the safety constraints with~~ safety constraints using additional GPs (Sui et al., 2015; Schreiter et al., 2015; Zimmer et al., 2018; Sui et al., 2018; Turchetta et al., 2019; Berkenkamp et al., 2020; Sergeyev et al., 2020; Baumann et al., 2021; Li et al., 2022). ~~The algorithms initiate with given safe observations. A safe set is then defined to restrict the~~ These algorithms begin with a small set of safe observations, and define a safe set to restrict exploration to regions with high safety confidence. ~~The~~ As learning progresses, this safe set expands ~~as the learning proceeds, and thus,~~ allowing the explorable area ~~grows to grow over time~~. Safe learning ~~is also considered in related domains approaches have also been explored in related fields,~~ such as Markov Decision Processes (Turchetta et al., 2019) and reinforcement learning (García et al., 2015).

~~In this paper, we focus on GPs as they are often considered the gold-standard when it comes to calibrated uncertainties. While such~~

~~While~~ safe learning methods have ~~achieved a huge impact, few demonstrated significant impact,~~ several challenges remain. ~~Firstly, GP priors need to be given prior to the exploration. First, the GP hyperparameters must be specified before exploration begins~~ (Sui et al., 2015; Berkenkamp et al., 2016; 2020) or ~~fitted with initial data (note that accessing the data is expensive)~~ be fitted using an initially small dataset (Schreiter et al., 2015; Zimmer et al., 2018; Li et al., 2022). In addition, safe learning algorithms ~~often~~ suffer from local exploration. ~~GPs: GP models are typically smooth and the uncertainty increases beyond the,~~ with uncertainty increasing beyond the boundaries of the reachable safe set boundary. ~~Disconnected safe regions will be.~~ This results in slow convergence, and disconnected safe regions are often classified as unsafe and ~~will remain unexplored~~. We provide a detailed analysis and ~~illustration of explorable regions in ??.~~ ~~In reality~~ visual illustration of this issue in Section 5. In practice, local exploration ~~increases the effort of deploying complicates the deployment of~~ safe learning algorithms ~~because the domain experts need to provide~~ as domain experts must supply safe data from multiple ~~distinct~~ safe regions.

Our contribution: As ~~safe learning~~ (Schreiter et al., 2015; Sui et al., 2015) is ~~always initialized~~ Safe learning generally begins with prior knowledge, ~~we fairly assume~~ (Schreiter et al., 2015; Sui et al., 2015; Berkenkamp et al., 2020). We assume that correlated experiments have ~~been performed and the results are available.~~ already been performed, and their results are readily available. This assumption enables transfer learning, ~~where the benefit is twofold:~~, offering two key benefits (see also Figure 1): (1) ~~exploration as well as~~ Exploration and expansion of safe regions are significantly accelerated, and (2) ~~disconnected safe regions can be explored allowing to discover larger safe regions.~~ Both advantages are made possible by guidance from the source task ~~may provide guidance on safe regions disconnected from the initial target data and thus helps us to explore globally.~~ Concrete applications. We empirically demonstrate both of the benefits and provide a theoretical analysis showing that conventional single-task approaches cannot identify unconnected safe regions. Real-world applications of this approach are ubiquitous, including ~~simulation to reality~~ simulation-to-reality transfer (Marco et al., 2017), serial production, and multi-fidelity modeling (Li et al., 2020).

Transfer learning can be ~~achieved by considering~~ implemented by jointly modeling the source and target tasks ~~jointly~~ as multi-output GPs (Journel & Huijbregts, 1976; Álvarez et al., 2012). However, GPs are notorious for ~~the their~~ cubic time complexity due to the inversion of Gram matrices. ~~Large amount~~ (Section 3.1). Consequently, large volumes of source data ~~thus introduce pronounced significantly increase~~ computational time, which is often a bottleneck in ~~real experiments.~~ We further ~~real-world experiments.~~ To address this, we modularize the multi-output GPs ~~such that the source relevant components can be pre-computed and fixed.~~ This ~~alleviates the complexity of multi-output GPs while the benefit is retained,~~ allowing source-related components to be precomputed and fixed, which reduces the computational complexity while retaining the benefits of transfer learning.

In summary, we 1) introduce the idea of ~~transfer safe sequential learning supported by a thorough mathematical formulation~~ safe transfer sequential learning, 2) derive that conventional ~~no-transfer approaches have an upper bound of explorable regions~~ single-task approaches cannot discover disjoint safe regions, 3) provide a modularized approach to multi-output GPs that ~~can alleviate~~ alleviates the computational burden of ~~incorporating the~~ source data, ~~with our technique being more general than the previous method in~~ Tighineanu et al. (2022), and 4) demonstrate the empirical efficacy ~~on safe AL problems.~~

Related work: Safe learning is considered in many ~~problems such as applications including~~ AL, BO, Markov Decision Processes (Turchetta et al., 2019) and reinforcement learning (García et al., 2015). In this paper, we focus on GP learning problems. ~~In~~ Gelbart et al. (2014); Hernandez-Lobato et al. (2015); Hernández-Lobato et al. (2016), ~~the authors~~ as GPs are considered the gold-standard when it comes to calibrated uncertainties which is particularly important for safe learning under uncertainty. Previous works (Gelbart et al., 2014; Hernandez-Lobato et al., 2015; Hernández-Lobato et al., 2016) investigated constrained learning with GPs. ~~The authors integrated by incorporating~~ constraints directly into the acquisition function (e.g., discounting the acquisition score by the probability of constraint violation). ~~These works~~ However, these approaches do not exclude unsafe data from the search pool, and ~~the experimenting examples are mostly not safety-critical~~ generally address non-safety-critical applications. A safe set concept was introduced for safe BO (Sui et al., 2015) and safe AL (Schreiter et al., 2015). ~~The concept was then~~ and later extended to BO with multiple safety constraints (Berkenkamp et al., 2020), to AL for time series modeling (Zimmer et al., 2018), and to AL for multi-output problems (Li et al., 2022). For safe BO, Sui et al. (2018) proposed ~~to conduct the a two-stage approach, separating~~ safe set exploration and BO ~~in two distinguished stages.~~ All. However, all of these methods suffer from local exploration. ~~Sergeyev et al. (2020) considered~~ (Section 5). Some recent methods address disjoint safe regions, ~~assuming regions separated only by a small gap.~~ For example, Sergeyev et al. (2020) considered regions separated by small gaps where the constraint function(s), with the noise, shortly goes beneath (but still close to) functions briefly fall below, but remain near, the safety threshold. Baumann et al. (2021) proposed a global safe BO method ~~on for~~ dynamical systems, assuming that unsafe ~~areas are regions can be~~ approached slowly enough and that there exists ~~such that~~ an intervention mechanism ~~which stops the system quickly enough.~~ None of these methods exploits transfer safe learning exists to stop the system in time. Despite these advances, none

Table 1: Key Notation

<u>Symbols</u>	<u>Meaning</u>
N_{init}	number of initial target data points
N_{query}	number of target data points added by AL
$N = N_{\text{init}} + \dots + N_{\text{init}} + N_{\text{query}}$	number of total target points
N_{source}	number of source data points
$\mathcal{D}_N = \{\mathbf{x}_{1:N}, \mathbf{y}_{1:N}, \mathbf{z}_{1:N}\}$	dataset of the target task
$\mathcal{D}_{\text{source}} = \{\mathbf{x}_{s,1:N_{\text{source}}}, \mathbf{y}_{s,1:N_{\text{source}}}, \mathbf{z}_{s,1:N_{\text{source}}}\}$	dataset of the source task
$\mathbf{z} = (z^1, \dots, z^J)$	safety variables of the target task
$\mathbf{z}_s = (z_s^1, \dots, z_s^J)$	safety variables of the source task
$y = f(\mathbf{x}) + \epsilon_f$	model of the target observation y
$z^j = q^j(\mathbf{x}) + \epsilon_{q^j}$	model of the target safety observation z^j
$y_s = f_s(\mathbf{x}) + \epsilon_{f_s}$	model of the source observation y_s
$z_s^j = q_s^j(\mathbf{x}) + \epsilon_{q_s^j}$	model of the source safety observation z_s^j
$f \sim \mathcal{GP}(\mathbf{0}, k_f)$	single-output GP prior over target main function f
$q^j \sim \mathcal{GP}(\mathbf{0}, k_{q^j})$	single-output GP prior over target safety function q^j
$\mathbf{f} \sim \mathcal{GP}(\mathbf{0}, k_f)$	multi-output GP prior over main functions f_s and f
$\mathbf{q}^j \sim \mathcal{GP}(\mathbf{0}, k_{q^j})$	multi-output GP prior over safety functions q_s^j and q^j

of these approaches leverages safe transfer learning, which can allow for global exploration ~~on any systems given prior source knowledge~~ by utilizing prior knowledge from source tasks for a wide range of scenarios.

Transfer learning and ~~multi-task learning have caught~~ multitask learning have gained increasing attention. In particular, multi-output GP methods have been developed for ~~multi-task~~ multitask BO (Swersky et al., 2013; Poloczek et al., 2017), sim-to-real transfer for BO (Marco et al., 2017), and ~~multi-task~~ multitask AL (Zhang et al., 2016). However, GPs ~~have time complexity cubic~~ face cubic time complexity with respect to the number of observations, ~~competed by a challenge that grows with~~ multiple outputs. In Tighineanu et al. (2022), the authors assume a specific structure of the multi-output kernel, ~~and which allows to~~ factorize the computation with an ensembling technique. This eases the computational ~~burdens~~ burden for transfer sequential learning. In our paper, we propose a modularized ~~transfer safe learning to facilitate real experiments while avoiding safe transfer learning that avoids the~~ cubic complexity. ~~Our modularization technique can be generalized to arbitrary multi-output kernels.~~

Paper structure: The remaining of this paper is structured as follows: ~~we provide the goal of safe sequential learning in ??; in ??, we introduce the background and analyze the local exploration problem~~. We provide the setup and problem statement in Section 2, background and assumptions of GPs and multitask GPs in Section 3. Section 4 introduces the state-of-the-art safe AL algorithm. Section 5 discusses theoretical perspective of safe learning; ~~?? elaborates our approach under a transfer learning scenario; ?? and demonstrate that safe learning approaches based on standard GPs suffer from local exploration.~~ Section 6 elaborates our safe transfer learning approach and our modular computation scheme. Section 7 is the experimental study; ~~finally~~. Finally, we conclude our paper in ~~??~~ Section 8.

2 ~~Problem statement~~ Safe Transfer Active Learning Setup

Preliminary: Transfer Learning aims to transfer knowledge from previous, *source*, systems to a new, *target*, system. Usually, there exist a lot of data from one or more source systems and only few or no data from the target system. Safe Transfer Active Learning will exploit the knowledge from the source systems' data and allows for safe and active data collection on the target system. Throughout this paper, we inspect regression output and safety values. ~~Each problems.~~

Target and Safety – Notation: Each D -dimensional input $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^D$ has a corresponding noisy-regression output $y \in \mathbb{R}$ and the corresponding noisy safety values jointly expressed as a vector $\mathbf{z} = (z^1, \dots, z^J) \in \mathbb{R}^J$. $y = f(\mathbf{x}) + \epsilon_f, z^j = q^j(\mathbf{x}) + \epsilon_{q^j}$, where $\epsilon_f \sim \mathcal{N}(0, \sigma_f^2)$, $\epsilon_{q^j} \sim \mathcal{N}(0, \sigma_{q^j}^2)$. In addition, $y_s = f_s(\mathbf{x}_s) + \epsilon_{f_s}, z_s^j = q_s^j(\mathbf{x}_s) + \epsilon_{q_s^j}$, where $\epsilon_{f_s} \sim \mathcal{N}(0, \sigma_{f_s}^2)$, $\epsilon_{q_s^j} \sim \mathcal{N}(0, \sigma_{q_s^j}^2)$. $\{f, q^j\}$ are our target black-box function and safety functions; $\{f_s, q_s^j\}$ are our source main and safety functions, where J is the number of safety variables. There are J thresholds $T_j \in \mathbb{R}, j = 1, \dots, J$, and an input \mathbf{x} is safe if the corresponding safety values $z^j > T_j$ for all $j = 1, \dots, J$. It is assumed that the underlying functions of y, z^1, \dots, z^J are all unknown.

Source and Safety – Notation: Similarly, there exist output and safety values of one or more source tasks, again from unknown underlying functions. The source output value is denoted by $y_s \in \mathbb{R}$ and source safety values by $\mathbf{z}_s = (z_s^1, \dots, z_s^J) \in \mathbb{R}^J$, s is the index of source task(s). The source tasks are defined on the same domain \mathcal{X} . The source and target tasks may have different numbers of safety conditions constraint variables, but we can add trivial constraints (e.g. $1 \geq -\infty$) to any of the tasks in order to have the same number of constraints J . The notation is summarized in Table 1. Furthermore, the source data may or may not be measured with the same safety constraints as the target task. For example, in a simulation-to-reality transfer (Marco et al., 2017), the source dataset can be obtained unconstrained.

Safe learning problem statement Datasets – Notation: We are given a small number of safe observations $\mathcal{D}_{N_{init}} = \{\mathbf{x}_{1:N_{init}}, y_{1:N_{init}}, \mathbf{z}_{1:N_{init}}\}$, $\mathbf{x}_{1:N_{init}} = \{\mathbf{x}_1, \dots, \mathbf{x}_{N_{init}}\} \subseteq \mathcal{X}$, $y_{1:N_{init}} = \{y_1, \dots, y_{N_{init}}\} \subseteq \mathbb{R}$ and safety observations $\mathbf{z}_{1:N_{init}} := (z^1, \dots, z^J)_{1:N_{init}} := (z_{1:N_{init}}^1, \dots, z_{1:N_{init}}^J) = \{\mathbf{z}_n = (z_n^1, \dots, z_n^J)\}_{n=1}^{N_{init}}$. In practice, the initial data usually meet the safety constraints, i.e. $z_n^j \geq T_j$ for all observation index n and constraint index j . We are further given source data $\mathcal{D}_{N_{source}}^{source} = \{\mathbf{x}_{s,1:N_{source}}, y_{s,1:N_{source}}, \mathbf{z}_{s,1:N_{source}}\}$, $\mathbf{x}_{s,1:N_{source}} = \{\mathbf{x}_{s,1}, \dots, \mathbf{x}_{s,N_{source}}\} \subseteq \mathcal{X}$, $y_{s,1:N_{source}} = \{y_{s,1}, \dots, y_{s,N_{source}}\} \subseteq \mathbb{R}$, $\mathbf{z}_{s,1:N_{source}} = \{\mathbf{z}_n = (z_{s,n}^1, \dots, z_{s,n}^J) | n = 1, \dots, N_{source}\} \subseteq \mathbb{R}^J$, and N_{source} . A dataset over the target task is denoted by $\mathcal{D}_N = \{\mathbf{x}_{1:N}, y_{1:N}, \mathbf{z}_{1:N}\}$, $\mathbf{x}_{1:N} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\} \subseteq \mathcal{X}$, $y_{1:N} = \{y_1, \dots, y_N\} \subseteq \mathbb{R}$, safety observations $\mathbf{z}_{1:N} := \{\mathbf{z}_n = (z_n^1, \dots, z_n^J)\}_{n=1}^N \subseteq \mathbb{R}^J$, and N is the number of observed data. In this paper, N is not fixed, as we may actively add new labeled data. We denote the source data by $\mathcal{D}_{N_{source}}^{source} = \{\mathbf{x}_{s,1:N_{source}}, y_{s,1:N_{source}}, \mathbf{z}_{s,1:N_{source}}\} \subseteq \mathcal{X} \times \mathbb{R} \times \mathbb{R}^J$, s is the index of source task and N_{source} is the number of all source data points. In our main paper, we consider only one source task for simplicity, while ?? provides formulation and ablation studies Appendix D.2 provides formulation on more source tasks. We assume N_{source} is large enough, and we do not need to explore for the source task. This is often the case when there is plenty of data from previous versions of systems or prototypes. Notably, the source data do not need to be measured with the same safety constraints as the target task. For example, in a sim-to-real transfer (Marco et al., 2017), the source dataset can be obtained unconstrained. Please also see Table 1 for a summary of our notation.

Safe Active Learning Procedure: The goal of safe AL is to collect data actively and safely on the target system, such that the final dataset helps to model the regression output y on the safe region of input space \mathcal{X} , i.e. subset of \mathcal{X} corresponding to $z^1 \geq T_1, \dots, z^J \geq T_J$.

Concretely speaking, we are given a small amount of data on the target task, i.e. \mathcal{D}_N where the initial size $N = N_{init}$ is small. The initial data are typically given by a domain expert and are safe, i.e. for $\mathcal{D}_{N_{init}} = \{\mathbf{x}_{1:N_{init}}, y_{1:N_{init}}, \mathbf{z}_{1:N_{init}}\}, \forall n = 1, \dots, N_{init}, \mathbf{z}_n = (z_n^1, \dots, z_n^J)$ satisfy the safety constraints $z_n^1 \geq T_1, \dots, z_n^J \geq T_J$.

The goal is to evaluate the function $f: \mathcal{X} \rightarrow \mathbb{R}$ where each evaluation is expensive. In each iteration, we select a point $\mathbf{x}_* \in \mathcal{X}_{pool} \subseteq \mathcal{X}$ to evaluate. $\mathcal{X}_{pool} \subseteq \mathcal{X}$. At each N , one seeks the next point $\mathbf{x}_* \in \mathcal{X}_{pool} \subseteq \mathcal{X}$ to be evaluated. $\mathcal{X}_{pool} \subseteq \mathcal{X}$ is the search pool which can be the entire space \mathcal{X} or a predefined subspace of \mathcal{X} , depending on the applications. This selection should respect the a priori unknown safety constraints $\forall j = 1, \dots, J, q^j(\mathbf{x}_*) \geq T_j$, where true q^j are inaccessible. Then, a budget-consuming labeling process occurs, and we obtain The evaluation is budget consuming and safety critical, and it will return a noisy y_* and noisy safety values \mathbf{z}_* . The labeled points are then added to $\mathcal{D}_{N_{init}}$. Ideally, we need

to make sure that $\mathbf{z}_* = (z_*^1, \dots, z_*^J)$ respect the safety constraints $z_*^j \geq T_j$ for all $j = 1, \dots, J$ and that y_* is informative for the modeling of target y . As the safety outputs are unknown when an \mathbf{x}_* is selected, guaranteeing safety is challenging. Safe learning methods resort to allowing queries that are safe only with high probability (Sui et al., 2015; 2018; Zimmer et al., 2018; Li et al., 2022).

Afterward, the labeled point is added to \mathcal{D}_N (observed dataset becomes $\mathcal{D}_{N_{init}+1}$), and we proceed to the next iterations. In the following, N will be the size of observed dataset of the target task, and it varies from N_{init} to $N_{init} + \text{num_steps}$ (number of AL steps, i.e. AL budget). The notation is summarized in Table 1. N_{init} is initially N_{init} and grows to $N_{init} + N_{query}$. N_{query} is the number of the learning iterations, i.e. the number of data points actively added.

This problem formulation applies to both AL and BO. In this paper, we focus on AL problems. The goal is using the evaluations to make accurate predictions $f(\mathcal{X})$, and the points we select would favor general understanding over space.

Safe Transfer Active Learning Aim: In particular, this paper aims to build a new safe transfer AL, a safe AL algorithm with multi-output GPs, so that we leverage the information of the source data $\mathcal{D}_{N_{source}}^{\text{source}}$ to explore a larger safe area. Our algorithm aims to

- (i) collect as few (small N_{query}) data as possible for building an accurate regression model of y (in the safe part of the input domain \mathcal{X} , up to the safety constraints).
- (ii) collect the data in a safe way and hereby explore the safe region including its boundaries.
- (iii) in particular explore larger safe areas than benchmarks in a faster way.

3 Background & local exploration: Gaussian Processes of safe learning methods Single and Multiple Tasks

In this section, we introduce GPs, safe learning algorithms for GPs, and then provide detailed analysis and illustration of the local exploration problem. Gaussian Processes (GPs) and their multitask variant. GPs are the workhorse of safe AL in which they are routinely applied to select safe and informative data points (Schreiter et al., 2015; Zimmer et al., 2018; Li et al., 2022). Multitask GPs extend this role to safe transfer AL by leveraging information across related tasks, enabling more efficient exploration of the target system.

Gaussian processes (GPs):

3.1 Gaussian Processes (GPs)

Suppose we aim to model the output y and the safety observations z^1, \dots, z^J with GPs. Here, we introduce the modeling scheme and the underlying assumptions. The first assumption is that the data represent functional values blurred with i.i.d. Gaussian noises.

Assumption 3.1 (Data: target task). Assume $y = f(\mathbf{x}) + \epsilon_f$, where $\epsilon_f \sim \mathcal{N}(0, \sigma_f^2)$, for our target observations. We further assume that $z^j = q^j(\mathbf{x}) + \epsilon_{q^j}$, where $\epsilon_{q^j} \sim \mathcal{N}(0, \sigma_{q^j}^2)$, and $j = 1, \dots, J$ indexes the safety constraints. All of the noise variances $\{\sigma_f^2, \sigma_{q^1}^2, \dots, \sigma_{q^J}^2\}$ are positive.

We then place a GP assumption on each of the underlying functions f, q^1, \dots, q^J . A GP is a stochastic process specified by a mean and a kernel function (Rasmussen & Williams, 2006; Kanagawa et al., 2018; Schoelkopf & Smola, 2002). Without loss of generality, we assume the GPs have zero mean. In addition, without prior knowledge to In this work, we set the mean to zero — a common practice, as normalized data typically justifies this assumption. The kernel function, $\mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$, specifies the covariance of function values at different input points. Without prior knowledge of the data, it is common to assume the we make

the standard assumption that the governing kernels are stationary. The GP assumption is then formulated as the following.

Assumption 3.2 (Model: single task). For $g = f, q^1, \dots, q^J$, each function, $g \in \{f, q^1, \dots, q^J\}$, we assume that $g \sim \mathcal{GP}(0, k_g)$ and with a stationary kernel with bounded variance, $k_g(\mathbf{x}, \mathbf{x}') := k_g(\mathbf{x} - \mathbf{x}') \leq 1$ are stationary.

Bounding the kernels by 1 provides advantages in theoretical analysis (Srinivas et al., 2012) and is not restrictive because the data are since the data is usually normalized to zero-mean and unit variance.

The GP assumptions (Assumption 3.1 and Assumption 3.2) indicate that each of $\{f, q^1, \dots, q^J\}$ has a predictive distribution given as the following provide the predictive distribution of the functions f, q^1, \dots, q^J . We write down the distribution for the function f at a test point \mathbf{x}_* , while the distributions of q^j :

$$p(f(\mathbf{x}_*) | \mathbf{x}_{1:N}, y_{1:N}) = \mathcal{N}(\mu_{f,N}(\mathbf{x}_*), \sigma_{f,N}^2(\mathbf{x}_*)), \quad (1)$$

where

$$\begin{aligned} \mu_{f,N}(\mathbf{x}_*) &= k_f(\mathbf{x}_{1:N}, \mathbf{x}_*)^T (\mathbf{K}_f + \sigma_f^2 \mathbf{I})^{-1} y_{1:N}, \\ \sigma_{f,N}^2(\mathbf{x}_*) &= k_f(\mathbf{x}_*, \mathbf{x}_*) - k_f(\mathbf{x}_{1:N}, \mathbf{x}_*)^T (\mathbf{K}_f + \sigma_f^2 \mathbf{I})^{-1} k_f(\mathbf{x}_{1:N}, \mathbf{x}_*). \end{aligned} \quad (2)$$

We use the notation $k_f(\mathbf{x}_{1:N}, \mathbf{x}_*) = (k_f(\mathbf{x}_1, \mathbf{x}_*), \dots, k_f(\mathbf{x}_N, \mathbf{x}_*)) \in \mathbb{R}^{N \times 1}$ to denote the kernel vector between the training points $\mathbf{x}_{1:N}$ and the test point \mathbf{x}_* . The kernel matrix $\mathbf{K}_f \in \mathbb{R}^{N \times N}$ contains the covariances between the training points $\mathbf{x}_{1:N}$ with $[\mathbf{K}_f]_{m,n} = k_f(\mathbf{x}_m, \mathbf{x}_n)$, $m, n = 1, \dots, N$.

Typically, k_f is parameterized and can be jointly fitted with the noise variance σ_f^2 . Common fitting techniques involve computing the marginal likelihood, $\mathcal{N}(y_{1:N} | \mathbf{0}, \mathbf{K}_f + \sigma_f^2 \mathbf{I})$, where the runtime complexity is $\mathcal{O}(N^3)$, dominated by the inversion of the Gram matrix $(\mathbf{K}_f + \sigma_f^2 \mathbf{I})^{-1}$.

The predictive distributions of the safety functions q^1, \dots, q^J can be obtained by replacing f with q^j and q^1, \dots, q^J and the outputs $y_{1:N}$ with $z_{1:N}^j$: $p(f(\mathbf{x}_*) | \mathbf{x}_{1:N}, y_{1:N}) = \mathcal{N}(\mu_{f,N}(\mathbf{x}_*), \sigma_{f,N}^2(\mathbf{x}_*))$.

$$\begin{aligned} \mu_{f,N}(\mathbf{x}_*) &= \mu_{f,N} = k_f(\mathbf{x}_{1:N}, \mathbf{x}_*)^T (\mathbf{K}_f + \sigma_f^2 \mathbf{I})^{-1} y_{1:N}, \\ \sigma_{f,N}^2(\mathbf{x}_*) &= \sigma_{f,N}^2 = k_f(\mathbf{x}_*, \mathbf{x}_*) - k_f(\mathbf{x}_{1:N}, \mathbf{x}_*)^T (\mathbf{K}_f + \sigma_f^2 \mathbf{I})^{-1} k_f(\mathbf{x}_{1:N}, \mathbf{x}_*), \end{aligned}$$

where $k_f(\mathbf{x}_{1:N}, \mathbf{x}_*) = (k_f(\mathbf{x}_1, \mathbf{x}_*), \dots, k_f(\mathbf{x}_N, \mathbf{x}_*)) \in \mathbb{R}^{N \times 1}$, $z_{1:N}^j, j = 1, \dots, J$ in Equations (1) and (2). Similarly, the log-likelihood can be maximized for each q^j by jointly learning k_{q^j} and $\mathbf{K}_f \in \mathbb{R}^{N \times N}$ is a matrix with $[\mathbf{K}_f]_{ij} = k_f(\mathbf{x}_i, \mathbf{x}_j)$. Typically, k_f is parameterized and can be fitted together with σ_f^2 . $\sigma_{q^j}^2$ in the same manner, $j = 1, \dots, J$.

Key notation **Symbols Meaning** $\mathcal{D}_N = \{\mathbf{x}_{1:N}, y_{1:N}, z_{1:N}^j\}$ dataset of the target task

3.2 Transfer Task GPs

In the presence of a source task, one can model the source task and the target task jointly with multi-output GPs (Journal & Huijbregts, 1976; Álvarez et al., 2012; Tighineanu et al., 2022). The key idea is to augment the input with a task index variable, allowing the model to distinguish between tasks while sharing information across them. Leveraging a source task in this way improves data efficiency on the target system, as relevant information can flow from the source to the target task. To proceed, it is necessary to first make a hypothesis on the source data, similar to Assumption 3.1.

Assumption 3.3 (Data: source task). The source data are modeled as $y_s = f_s(\mathbf{x}) + \epsilon_{f_s}$, $z_s^j = q_s^j(\mathbf{x}) + \epsilon_{q_s^j}$, where $\{f_s, q_s^1, \dots, q_s^J\}$ are unknown source main and safety functions, and s indexes the source task. We assume additive noise distributed as $\epsilon_{f_s} \sim \mathcal{N}(0, \sigma_{f_s}^2)$, $N = N_{init}, \dots, N_{init} + \text{num_steps}$, $\epsilon_{q_s^j} \sim \mathcal{N}(0, \sigma_{q_s^j}^2)$ with all noise variances $\{\sigma_{f_s}^2, \sigma_{q_s^1}^2, \dots, \sigma_{q_s^J}^2\}$ being positive.

~~$z_{1:N}^j = \{z_1^j, \dots, z_N^j\}$ safety observations of the j -th constraint (unknown function q^j)~~ Next, we introduce task indices, with s for the source task, and t for the target task. These indices allow us to describe the source and target functions jointly as multitask functions. The data are then concatenated with the task indices, and, based on Assumption 3.1 and Assumption 3.3, we define the multitask functions $\mathbf{f}, \mathbf{q}^1, \dots, \mathbf{q}^J : \mathcal{X} \times \{\text{task indices}\} \rightarrow \mathbb{R}$, where $\mathbf{f}(\cdot, s) = f_s(\cdot)$ corresponds to the source main function, $\mathbf{f}(\cdot, t) = f(\cdot)$ to the target main function, $\mathbf{q}^j(\cdot, s) = q_s^j(\cdot)$ to the source safety constraint and $\mathbf{q}^j(\cdot, t) = q^j(\cdot)$ to the target safety constraint. We use bold symbols to indicate the multitask functions. Subsequently, we assign GP priors to the multitask functions.

Assumption 3.4 (Model: multitask). For each multitask function $\mathbf{g} \in \{\mathbf{f}, \mathbf{q}^1, \dots, \mathbf{q}^J\}$, we assume $\mathbf{g} \sim \mathcal{GP}(\mathbf{0}, k_{\mathbf{g}})$ with kernel $k_{\mathbf{g}} : (\mathcal{X} \times \{\text{task index}\}) \times (\mathcal{X} \times \{\text{task index}\}) \rightarrow \mathbb{R}$.

~~$z_{1:N} = (z_{1:N}^1, \dots, z_{1:N}^J)$ safety observations of all constraints jointly~~ ~~$\mathcal{D}_{N_{source}}^{\text{source}}$ dataset of the source task~~ ~~$\{\mathbf{x}_{s,1:N_{source}}, y_{s,1:N_{source}}, z_{s,1:N_{source}}\}$ $y = f(\mathbf{x}) + \epsilon_f$ observation of unknown function $f \sim \mathcal{GP}(0, k_f)$, $\epsilon_f \sim \mathcal{N}(0, \sigma_f^2)$~~ Note that the structure of our new assumption resembles Assumption 3.2. However, the GP is now defined jointly over the source and target task, allowing information to flow between them. Example kernels are provided in Section 6.3. We proceed by presenting the predictive distribution for the main target task, leveraging source and target data. To incorporate task indices into the given input data, we use a hat notation: We denote the source input, $\mathbf{x}_{s,1:N_{source}}$, paired with the source index s , with $\hat{\mathbf{x}}_{s,1:N_{source}} := \{(\mathbf{x}_{s,i}, s) | \mathbf{x}_{s,i} \in \mathbf{x}_{s,1:N_{source}}\}$. Analogously, the target training and test points, $\mathbf{x}_{1:N}$ and \mathbf{x}_* , paired with the target index t , are represented as $\hat{\mathbf{x}}_{1:N} := \{(\mathbf{x}_i, t) | \mathbf{x}_i \in \mathbf{x}_{1:N}\}$ and $\hat{\mathbf{x}}_* := (\mathbf{x}_*, t)$. The predictive distribution is then given by:

$$p(\mathbf{f}(\mathbf{x}_*, t) | \mathbf{x}_{1:N}, y_{1:N}, \mathbf{x}_{s,1:N_{source}}, y_{s,1:N_{source}}) = \mathcal{N}(\mu_{\mathbf{f},N}(\mathbf{x}_*), \sigma_{\mathbf{f},N}^2(\mathbf{x}_*)),$$

where the predictive mean $\mu_{\mathbf{f},N}(\mathbf{x}_*)$ and variance $\sigma_{\mathbf{f},N}^2(\mathbf{x}_*)$ are given by

$$\begin{aligned} \mu_{\mathbf{f},N}(\mathbf{x}_*) &= \mathbf{v}_f^T \Omega_f^{-1} \begin{pmatrix} y_{s,1:N_{source}} \\ y_{1:N} \end{pmatrix}, \\ \sigma_{\mathbf{f},N}^2(\mathbf{x}_*) &= k_f(\hat{\mathbf{x}}_*, \hat{\mathbf{x}}_*) - \mathbf{v}_f^T \Omega_f^{-1} \mathbf{v}_f. \end{aligned} \quad (3)$$

~~$z^j = q^j(\mathbf{x}) + \epsilon_{q^j}$ observation of unknown constraint $q^j \sim \mathcal{GP}(0, k_{q^j})$, $\epsilon_{q^j} \sim \mathcal{N}(0, \sigma_{q^j}^2)$~~ The vector \mathbf{v}_f represents the covariances between the training points, aggregated over the source points $\hat{\mathbf{x}}_{s,1:N_{source}}$ and the target training points $\hat{\mathbf{x}}_{1:N}$, and the target test point $\hat{\mathbf{x}}_*$. It is defined as:

$$\mathbf{v}_f = \begin{pmatrix} k_f(\hat{\mathbf{x}}_{s,1:N_{source}}, \hat{\mathbf{x}}_*) \\ k_f(\hat{\mathbf{x}}_{1:N}, \hat{\mathbf{x}}_*) \end{pmatrix} \in \mathbb{R}^{(N_{source}+N) \times 1}. \quad (4)$$

The matrix Ω_f combines the kernel matrices and noise variances for both source and target tasks, forming a block structure:

$$\Omega_f = \begin{pmatrix} K_{f_s} + \sigma_{f_s}^2 I_{N_{source}} & K_{f_s, f} \\ K_{f_s, f}^T & K_f + \sigma_f^2 I_N \end{pmatrix} \in \mathbb{R}^{(N_{source}+N) \times (N_{source}+N)}, \quad (5)$$

where $K_{f_s} = k_f(\hat{\mathbf{x}}_{s,1:N_{source}}, \hat{\mathbf{x}}_{s,1:N_{source}})$ denotes the kernel matrix between the source data points, $K_{f_s, f} = k_f(\hat{\mathbf{x}}_{s,1:N_{source}}, \hat{\mathbf{x}}_{1:N})$ between source and target training points, and $K_f = k_f(\hat{\mathbf{x}}_{1:N}, \hat{\mathbf{x}}_{1:N})$ between

target training points. For brevity, we omitted the task index from the predictive terms $\mu_{f,N}(\mathbf{x}_*)$ and $\sigma_{f,N}^2(\mathbf{x}_*)$, as this paper focuses exclusively on predictions for the target task. For brevity, we present formulas only for the main function; the safety functions are analogous.

$q^j(\mathbf{x}) \geq T_j$ ~~j -th safety condition~~

$y_s = f_s(\mathbf{x}_s) + \epsilon_{f_s}$ ~~source task observation prior $f_s \sim \mathcal{GP}(0, k_{f_s})$, $\epsilon_{f_s} \sim \mathcal{N}(0, \sigma_{f_s}^2)$~~ The time complexity of the predictive distribution is $\mathcal{O}((N_{\text{source}} + N)^3)$ due to the inversion of the Gram matrix Ω_f . Similarly, the runtime for estimating the likelihood, and consequently for training the hyperparameters, is also $\mathcal{O}((N_{\text{source}} + N)^3)$. While this higher time complexity introduces additional computational overhead, it is offset by the benefit of improved data efficiency through the joint modeling of source and target tasks.

$z_s^j = q_s^j(\mathbf{x}_s) + \epsilon_{q_s^j}$ ~~source task constraint prior $q_s^j \sim \mathcal{GP}(0, k_{q_s^j})$, $\epsilon_{q_s^j} \sim \mathcal{N}(0, \sigma_{q_s^j}^2)$~~

Remark 3.5. In our paper, all safety measurements z^1, \dots, z^J are modeled independently. If the variables are not independent, our methods and analysis still apply, as the dependent constraints can be grouped, and the problem reduces back to the independent case.

$f: \mathcal{X} \times \{\text{task indices}\} \rightarrow \mathbb{R}$ ~~f_s and f jointly as a multi-task function~~

4 Background: Safe Active Learning

$q^j: \mathcal{X} \times \{\text{task indices}\} \rightarrow \mathbb{R}$ ~~q_s^j and q^j jointly as a multi-task function~~

$f \sim \mathcal{GP}(\mathbf{0}, k_f)$ ~~multi-task GP prior, kernel k_f parameterized by $\boldsymbol{\theta}_f = (\theta_{f_s}, \theta_f)$~~ In this section, we introduce state-of-the-art safe Active Learning (safe AL). The state-of-the-art of Safe AL can not exploit knowledge in form of source data. Although source data is not considered in this section, we will still write *target task* to make the distinction between the two tasks clear.

$q^j \sim \mathcal{GP}(\mathbf{0}, k_{q^j})$ ~~multi-task GP prior, kernel k_{q^j} parameterized by $\boldsymbol{\theta}_{q^j} = (\theta_{q_s^j}, \theta_{q^j})$~~

Safe AL (Schreiter et al., 2015; Zimmer et al., 2018; Li et al., 2022) aims to select data actively and safely to learn about a target task. At a given number of available target data N , the goal is to select $\mathbf{x}_* \in \mathcal{X}_{\text{pool}} \subseteq \mathcal{X}$, that gives us a safe $\mathbf{z}_* = (z_*^1, \dots, z_*^J)$ and informative y_* (informative in the context of modeling).

Safe learning: The key of safe AL is the safety and data selection criteria. Commonly, these criteria employ GPs due to the well calibrated predictive uncertainty (Schreiter et al., 2015; Zimmer et al., 2018; Li et al., 2022). It is worth highlighting a closely related field, safe Bayesian optimization (BO) (Sui et al., 2015; 2018; Berkenkamp et al., 2020), which follows a similar procedure except that the goal is to search for an optimum y_* subject to safety constraints $z_*^1 \geq T_1, \dots, z_*^J \geq T_J$.

A core of safe learning methods (Sui et al., 2015; 2018; Berkenkamp et al., 2020; Baumann et al., 2021) is to

Algorithm 1 Safe AL

Require: Assumption 3.1, Assumption 3.2, $\mathcal{D}_{N_{\text{init}}}$, $\mathcal{X}_{\text{pool}}$, β or α , N_{query} , thresholds T_1, \dots, T_J

- 1: **for** $N = N_{\text{init}}, \dots, N_{\text{init}} + N_{\text{query}} - 1$ **do**
- 2: Fit GPs f, q_1, \dots, q_J with \mathcal{D}_N
- 3: $\mathcal{S}_N \leftarrow \bigcap_{j=1}^J \{\mathbf{x} \in \mathcal{X}_{\text{pool}} \mid \mu_{q^j,N}(\mathbf{x}) - \beta^{1/2} \sigma_{q^j,N}(\mathbf{x}) \geq T_j\}$ (Equations (2) and (6))
- 4: $\mathbf{x}_* \leftarrow \arg\max_{\mathbf{x} \in \mathcal{S}_N} a(\mathbf{x})$, $a(\mathbf{x}) = \sum_{g \in \{f, q^1, \dots, q^J\}} H_g[\mathbf{x} | \mathcal{D}_N]$
- 5: Query \mathbf{x}_* to get evaluations y_* and \mathbf{z}_*
- 6: $\mathcal{D}_{N+1} \leftarrow \mathcal{D}_N \cup \{\mathbf{x}_*, y_*, \mathbf{z}_*\}$, $\mathcal{X}_{\text{pool}} \leftarrow \mathcal{X}_{\text{pool}} \setminus \{\mathbf{x}_*\}$
- 7: **end for**
- 8: **Return** $\mathcal{D}_{N_{\text{init}} + N_{\text{query}}}$, trained models f, q_1, \dots, q_J

Safety Condition: The safety variables are modeled by GP functions q^1, \dots, q^J (Section 3.1), and the core is to compare the safety confidence bounds (Equation (2)) with the thresholds and define a safe set $\mathcal{S}_N \subseteq \mathcal{X}_{\text{pool}}$ as $T_1, \dots, T_J \in \mathbb{R}$. At each iteration N , we can compute the safety probability $p(q^j(\mathbf{x}) | \mathbf{x}_{1:N}, z_{1:N}^j) = \mathcal{N}(\mu_{q^j, N}(\mathbf{x}), \sigma_{q^j, N}^2(\mathbf{x}))$, for each safety constraint $\forall j = 1, \dots, J$. Sui et al. (2015) defines the safe set $\mathcal{S}_N \subseteq \mathcal{X}_{\text{pool}}$ as

$$\mathcal{S}_N = \cap_{j=1}^J \{\mathbf{x} \in \mathcal{X}_{\text{pool}} | \mu_{q^j, N}(\mathbf{x}) - \beta^{1/2} \sigma_{q^j, N}(\mathbf{x}) \geq T_j\}, \quad (6)$$

where $\beta \in \mathbb{R}^+$ is a parameter for probabilistic tolerance control (Sui et al., 2015; Berkenkamp et al., 2020). This definition is equivalent to $\forall \mathbf{x} \in \mathcal{S}_N, p(q^1(\mathbf{x}) \geq T_1, \dots, q^J(\mathbf{x}) \geq T_J) \geq (1 - \alpha)^J$ when $\alpha = 1 - \Phi(\beta^{1/2})$ ($\alpha \in [0, 1]$) where Φ is the standard Gaussian cumulative distribution function (CDF) (Schreiter et al., 2015; Zimmer et al., 2018; Li et al., 2022). Note that $\alpha > 0.5$ corresponds to $\beta \in \mathbb{C}, \beta^{1/2} < 0$ which is usually not considered because safe learning aims for high safety confidence while $\alpha > 0.5$ indicates a safety confidence of at most 50% per constraint - so at most a random guess.

In each iteration,

Information Criterion: Safe AL queries a new point is queried by mapping safe candidate inputs to acquisition scores:

$$\mathbf{x}_* = \operatorname{argmax}_{\mathbf{x} \in \mathcal{S}_N} a(\mathbf{x} | \mathcal{D}_N), \quad (7)$$

where \mathcal{D}_N is the current observed dataset and $a(\cdot)$ is an acquisition function. Notice here that $a(\cdot)$ and \mathcal{S}_N both depend on the observed dataset \mathcal{D}_N . In AL problems, a prominent acquisition function is the predictive entropy: $a(\mathbf{x} | \mathcal{D}_N) = H_f[\mathbf{x} | \mathcal{D}_N] = \frac{1}{2} \log(2\pi e \sigma_{f, N}^2(\mathbf{x}))$ (Schreiter et al., 2015; Zimmer et al., 2018; Li et al., 2022). We use $a(\mathbf{x} | \mathcal{D}_N) = \sum_{g \in \{f, q^1, \dots, q^J\}} H_g[\mathbf{x} | \mathcal{D}_N]$ to $a(\mathbf{x}) = H_f[\mathbf{x} | \mathcal{D}_N] = \frac{1}{2} \log(2\pi e \sigma_{f, N}^2(\mathbf{x}))$, where $\sigma_{f, N}^2$ is defined in Equation (2). To accelerate the exploration of safety models. It is the safety functions, Berkenkamp et al. (2020) incorporate the information of the safety functions by using $a(\mathbf{x}) = \max_{g \in \{f, q^1, \dots, q^J\}} \{H_g[\mathbf{x} | \mathcal{D}_N]\}$. Our acquisition function is built upon this and is written in Algorithm 1.

Please note again the close connection to BO: it is possible to exchange the acquisition function by the SafeOpt criteria for if one wants to address safe BO problems (Sui et al., 2015; Berkenkamp et al., 2020; Rothfuss et al., 2022)).

Notably, solving such a constrained optimization

Constrained Acquisition Optimization: Solving Equation (7) is challenging. In the literature (Schreiter et al., 2015; Zimmer et al., 2018; Li et al., 2022; Sui et al., 2015; Berkenkamp et al., 2020) (Schreiter et al., 2015; Li et al., 2022; Sui et al., 2015; Berkenkamp et al., 2020), this is solved on a discrete pool with finite elements, i.e. $N_{\text{pool}} := |\mathcal{X}_{\text{pool}}| < \infty$. One would apply Equation (2) to the entire pool $\mathcal{X}_{\text{pool}}$ compute the GP posteriors on the entire pool $\mathcal{X}_{\text{pool}}$ to determine the safe set, then optimize the acquisition scores over the safe set. The time complexity of making GP inference is $\mathcal{O}(N_{\text{pool}} N^2) + \mathcal{O}(N^3)$. Note that we compute the pool only once to solve ???. Usually in our paper, we inherit this finite discrete pool setting.

Time Complexity: The complexity comprises $\mathcal{O}(N^3)$ for GP training and $\mathcal{O}(N_{\text{pool}} N^2)$ for GP inference, assuming the Gram matrices $(\mathbf{K}_g + \sigma_g^2 \mathbf{I})^{-1} \forall g \in \{f, q^1, \dots, q^J\}$ are already computed during the training (Equation (2)). Importantly, GP inference is only performed once per query, whereas GP training (or more specifically, its most computationally expensive step, the matrix inversion) is repeated multiple times during parameter learning. As a consequence, the size of the discretized pool is not the main computational bottleneck, e.g. N_{pool} can be up to perhaps tens or N_{pool} can be much larger than the training dataset N , e.g. $\mathcal{X}_{\text{pool}}$ can include up to tens or even hundreds of thousands. The main bottleneck

is still the training of GPs which will be described later. Training a GP takes $\mathcal{O}(N^3)$ for **multiple** times. This factor depends on the optimizer, the number of kernel parameters, and numerical stability. Often, the complexity of training is significantly larger than making a GP inference, despite N_{pool} much larger than N . In our paper, we inherit this finite discrete pool setting. The of samples.

The whole learning process is summarized in Algorithm 1. summarized in Algorithm 1.

Sequential Learning $\mathcal{D}_{N_{init}}, \mathcal{X}_{pool}, \beta$ or α Fit GPs $(k_f, k_{q^j}, \sigma_f^2, \sigma_{q^j}^2)$ $\mathbf{x}_* \leftarrow \operatorname{argmax}_{\mathbf{x} \in S_N} a(\mathbf{x} | \mathcal{D}_N)$ Evaluate at \mathbf{x}_* to get y_* and \mathbf{z}_* $\mathcal{D}_{N+1} \leftarrow \mathcal{D}_N \cup \{\mathbf{x}_*, y_*, \mathbf{z}_*\}, \mathcal{X}_{pool} \leftarrow \mathcal{X}_{pool} \setminus \{\mathbf{x}_*\}$

5 Safe Learning Solely on Target Task: Local Exploration

Safe learning suffer from local exploration: Before we introduce our safe transfer AL approach in the next Section 6, we analyze a shortcoming of the standard safe AL (Algorithm 1). We quantify the upper bound for an explorable safe region, and prove that safe AL is limited to local exploration within the given bound. Note, that the analysis will not involve the acquisition function, and therefore the result applies not only to safe AL but also to GP based safe BO settings with safe set as in Equation (6).

0.2in The safety function $q(x) = \sin(10x^3 - 5x - 10) + \frac{1}{3}x^2 - \frac{1}{2}$. The observations are with noise drawn from $\mathcal{N}(0, 0.1^2)$. -0.2in Given observations \mathcal{D}_N , we would like to know, until how far into the input space the safety confidence is sufficiently high.

In this section, we analyze the upper bound of explorable safe regions.

Correlation weakened with increasing distance: The conventional safe AL (Algorithm 1) builds models based on a standard GP assumption (Assumption 3.1, Assumption 3.2), and then the explorable region is obtained by quantifying the safety confidence, conditioned on observed data \mathcal{D}_N (Equation (6)). The safety confidence is calculated from the GP predictive distributions (Equation (2)), and it thus depends on the kernel to correlate input points of various locations. Commonly used stationary kernels ~~measure the difference of~~ measure the distance between a pair of points, while the actual ~~point output~~ values do not matter (for two points $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$, $\|\mathbf{x} - \mathbf{x}'\|$ determines the covariance). These kernels have the property that closer points ~~correlate strongly~~ have higher kernel values, indicating stronger correlation, while distant points result in small kernel values. We first formulate this property as the following ~~assumption. Given a kernel function,~~

Definition 5.1. We call a kernel k a kernel with *correlation weakened by distance* if $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$, ~~assume~~ fulfills the following property: $\forall \delta > 0, \exists r > 0$ s.t. $\|\mathbf{x} - \mathbf{x}'\| \geq r \Rightarrow k(\mathbf{x}, \mathbf{x}') \leq \delta$ under $L2$ norm. ~~We provide expression~~

This definition will later help us quantify the upper bound of an explorable region. We provide expressions of popular stationary kernels (RBF kernel and Matérn kernels), as well as their relations between input distance r and covariance δ in the ~~??~~. Appendix B.

In the following, we

Low correlation leading to small safety probability: With Definition 5.1, we can now derive a theorem showing that standard kernels only allow local exploration of safety regions measuring the explorable region. The main idea is: when ~~a~~ an unlabeled point \mathbf{x}_* is far away from the ~~observations, we can get very small~~ δ observed inputs, the value of δ can become very small (i.e. small covariance measured by the kernel). Thus ~~the prediction at \mathbf{x}_* is weakly correlated,~~ for each $j = 1, \dots, J$, the model weakly correlates $q^j(\mathbf{x}_*)$ to the observations. As a result, the predictive mean is close to zero (GP prior) and the predictive uncertainty is large, both of which imply that the method has small safety confidence, i.e. $p((q^j(\mathbf{x}_*) \geq T_j) | \mathbf{x}_{1:N}, \mathbf{z}_{1:N}^j)$ is small. Here we assume that $q^j \geq T_j$ is ¹ at least if $\forall j = 1, \dots, J, q^j \geq T_j$ is not a trivial condition. In other words, T_j is in sensitive domain of, e.g. a trivial condition would be if a function q^j has values majorly distributed in $[-1, 1]$ but thresholded at $T_j = -2$.

¹A small safety confidence indicates $\exists j = 1, \dots, J, p((q^j(\mathbf{x}_*) \geq T_j) | \mathbf{x}_{1:N}, \mathbf{z}_{1:N}^j)$ is not high enough.

Theorem 5.2 (Local exploration of single-output GPs). We are given $\forall \mathbf{x}_* \in \mathcal{X}, \mathbf{x}_{1:N} \subseteq \mathcal{X}$, a kernel k_{q^j} satisfying ?? (distant points result in weak correlation) and $k_{q^j}(\cdot, \cdot) \leq 1$. Denote $k_{scale}^j := \max k_{q^j}(\cdot, \cdot)$. $q^j \sim \mathcal{GP}(0, k_{q^j})$ is a GP. For any safety constraint indexed by $j = 1, \dots, J$, let $\mathbf{z}_{1:N}^j := (z_1^j, \dots, z_N^j)$ is a set of observed noisy values and be the observed noisy safety values and let $\|(\mathbf{z}_1^j, \dots, \mathbf{z}_N^j)\| \leq \sqrt{N}$. The safety value $z^j = q^j(\mathbf{x}) + \epsilon_{q^j}$ satisfies the GP assumptions (Assumption 3.1, Assumption 3.2): $q^j \sim \mathcal{GP}(0, k_{q^j}), k_{q^j}(\cdot, \cdot) \leq 1, \epsilon_{q^j} \sim \mathcal{N}(0, \sigma_{q^j}^2)$. The kernel k_{q^j} is a kernel with correlation weakened by distance (Definition 5.1). Denote $k_{scale}^j := \max k_{q^j}(\cdot, \cdot)$. Then $\forall \delta \in (0, \sqrt{k_{scale}^j \sigma_{q^j}^2 / \sqrt{N}}), \exists r > 0$ s.t. when $\forall \mathbf{x}_* \in \mathcal{X}$ that fulfill $\min_{\mathbf{x}_i \in \mathbf{x}_{1:N}} \|\mathbf{x}_* - \mathbf{x}_i\| \geq r$, the probability thresholded on a constant T_j is bounded by $p\left((q^j(\mathbf{x}_*) \geq T_j) | \mathbf{x}_{1:N}, \mathbf{z}_{1:N}^j\right) \leq \Phi\left(\frac{N\delta/\sigma_{q^j}^2 - T_j}{\sqrt{k_{scale}^j - (\sqrt{N}\delta/\sigma_{q^j})^2}}\right)$. Φ is the CDF of standard Gaussian.

To prove this theorem, we need Equation (2) to compute the safety likelihood, and then we use the eigenvalues of the kernel Gram matrix together with Definition 5.1 to derive the final bound (see Appendix C for the detailed proof).

Our theorem (proof in the ??) provides the maximum safety probability of a point as a function of its distance to the observed data in \mathcal{X} . Therefore, it The safe set tolerance parameter β or α (Equation (6)) can be used to compute the covariance bound δ . For example, when $J = 1$ and $\beta^{1/2} = 2$, which means $p(q(\mathbf{x}) \geq T) \geq \Phi(2)$ is safe (Equation (6)), we choose a δ such that $\Phi\left(\frac{N\delta/\sigma_q^2 - T}{\sqrt{k_{scale} - (\sqrt{N}\delta/\sigma_q)^2}}\right) \leq \Phi(2)$ (j omitted when $J = 1$). Such a δ exists in all situation of our interest, as we will soon discuss. Given a δ , we can then determine a corresponding radius r (see e.g. Appendix B). Interpreting r as the radius around the observed data, the safety confidence outside this region always remains low. Since safety confidence decides the explorable regions (Equations (6) and (7)), this theorem measures an upper bound of explorable safe area. Notice that $\|\mathbf{z}_{1:N}^j\| \leq \sqrt{N}$. The upper bound is given for one safety constraint, and we can see from Equation (6) that the final bound of safety confidence is the product of the Φ term over different j . In other words, the more safety constraints, the smaller the explorable regions may be, which is intuitive.

Notice that $\|(\mathbf{z}_1^j, \dots, \mathbf{z}_N^j)\| \leq \sqrt{N}$ ($\forall j = 1, \dots, J$) is not very restrictive because an unit-variance dataset has $\|\mathbf{z}_{1:N}^j\| = \sqrt{N}$. $\|(\mathbf{z}_1^j, \dots, \mathbf{z}_N^j)\| = \sqrt{N}$. Note further that $\delta \in (0, \sqrt{k_{scale}^j \sigma_{q^j}^2 / \sqrt{N}})$ implies $k_{scale}^j > (\sqrt{N}\delta/\sigma_{q^j})^2$, which means the bound $\Phi\left(\frac{N\delta/\sigma_{q^j}^2 - T_j}{\sqrt{k_{scale}^j - (\sqrt{N}\delta/\sigma_{q^j})^2}}\right)$ is always valid.

This theorem indicates that a standard GP with commonly used kernels explores only neighboring regions of the initial $\mathbf{x}_{1:N}$. In practice, we consider safe AL on a discrete pool $\mathcal{X}_{pool} \subseteq \mathcal{X}$, which means the GP explores only neighboring discretized points. In ??, we will see that our new transfer safe sequential learning framework may explore beyond the neighborhood of target $\mathbf{x}_{1:N}$, taken the source data into consideration. The theorem is independent of the acquisition functions, and thus the local exploration problems present in all safe learning methods based on standard GPs.

Existence of δ for common safe learning situations: We would like to illustrate an example of using our theorem to compute an explorable bound. Before that, we will make a statement relating the safety level β to the quantity δ used in Theorem 5.2. This shows that a δ and, therefore, local exploration is present in all but some (at least) uncommon scenarios, which are in fact out of interest for the sake of safe exploration.

Corollary 5.3 (Existence of δ). We are given the assumptions in Theorem 5.2. For each $j = 1, \dots, J$, if either (1) $T_j \geq 0, \beta^{1/2} > 0$ or (2) $T_j < 0, \beta^{1/2} > \frac{|T_j|}{\sqrt{k_{scale}^j}}$, then $\exists \delta \in (0, \sqrt{k_{scale}^j \sigma_{q^j}^2 / \sqrt{N}})$ s.t.

$$\Phi\left(\frac{N\delta/\sigma_{q^j}^2 - T_j}{\sqrt{k_{scale}^j - (\sqrt{N}\delta/\sigma_{q^j})^2}}\right) \leq \Phi(\beta^{1/2}).$$

This corollary can be proved by inspecting the boundary of each constant (detailed in Appendix C).

0.2in

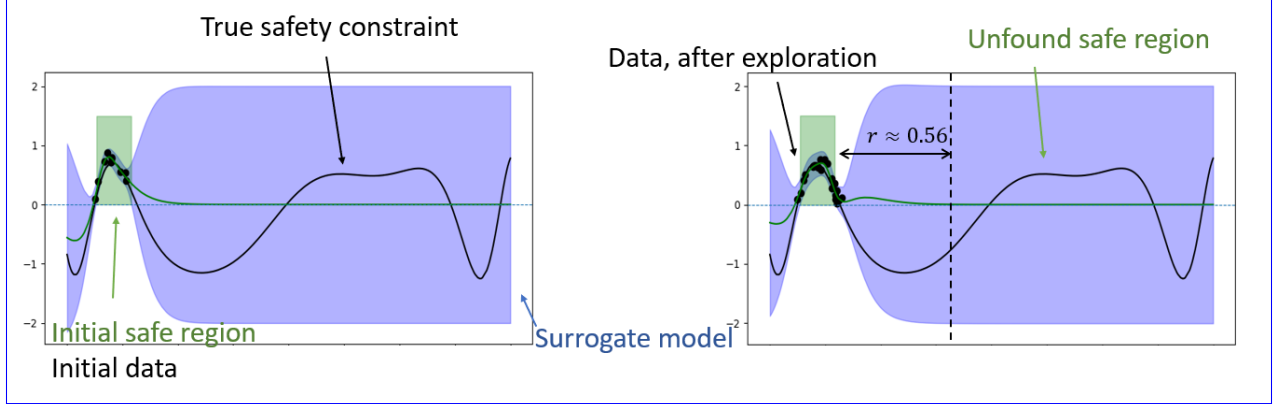


Figure 2: A safety constraint (shown in black) with two safe regions above threshold zero. Left graphics: Based on the initial data within one of the safe regions, a GP surrogate is trained. The blue line represents the mean prediction, while the blue shaded area indicates the uncertainty (e.g., confidence interval) around the mean. The green area indicates the learned safe area. Right graphics: After exploration, more points are sampled within the first safe region. However, the gap to the second safe region exceeds r , preventing the discovery of the second region, rendering the learned safe area almost unchanged. The true safety function used here is $q(x) = \sin(10x^3 - 5x - 10) + \frac{1}{3}x^2 - \frac{1}{2}$. The observations are with noise drawn from $\mathcal{N}(0, 0.1^2)$.

-0.2in

The key insight is that, a δ in Theorem 5.2, which bounds the safety likelihood, always exists for common selection of safety level $\beta^{1/2}$. There are two scenarios considered in Corollary 5.3. The first scenario, $T_j \geq 0, \beta^{1/2} > 0$ is common because $\beta^{1/2} > 0$ is always desired for safe exploration and stricter safety thresholds $T_1 \geq 0, \dots, T_J \geq 0$ may also occur. In the second scenario, the thresholds are softer, i.e. one or more of the thresholds T_1, \dots, T_J are smaller than zero. It turns out that $\beta^{1/2} > |T_j|/\sqrt{k_{scale}^j}, j = 1, \dots, J$ is desired as well for safe learning. Consider $|T_j|/\sqrt{k_{scale}^j} \geq \beta^{1/2} > 0$ for a $j \in \{1, \dots, J\}$, which is the scenario not fulfilling the condition. We focus on normalized variable z_j , where the underlying function is modeled by a GP $q^j \sim \mathcal{GP}(0, k_{qj})$. When this model extrapolates in regions where data are absent, the inference is highly based on the prior $q^j(x) \sim \mathcal{N}(0, k_{scale}^j)$. The safe set considers $p(q^j(x) > T_j) > \Phi(\beta^{1/2})$ as a safety condition on the j -th constraint, but the prior indicates $\Phi(-T_j/\sqrt{k_{scale}^j}) > \Phi(\beta^{1/2})$ which becomes a trivial condition when $|T_j|/\sqrt{k_{scale}^j} \geq \beta^{1/2} > 0$. In other words, any input x has a safe prior unless the data disagree. This is a scenario that is not of interest for safe learning.

Therefore, for all common selection of safety level $\beta^{1/2}$, Corollary 5.3 implies that we can find a δ and apply Theorem 5.2 to quantify the upper bound of explorable safe set, which shows the presence of local exploration.

Illustrating the theoretical result: In the following, we plug exact numbers into Theorem 5.2 for an illustration.

Example 5.4. We consider a one-dimensional toy dataset visualized in Figure 2. Assume $N = 10$, $\sigma_q^2 = 0.01$ and constraint $T = 0$. We omit safety constraint index j because, since $J = 1$ here, σ_q/\sqrt{N} is roughly 0.0316. In this example, the generated data have $\|z_{1:N}\| \leq \sqrt{10}$. σ_q/\sqrt{N} is roughly 0.0316. We train an unit-variance ($k_{scale} = \max k_q(\cdot, \cdot) = 1$, theorem requires $\delta < 0.0316$) Matérn-5/2 kernel on this example, and we obtain lengthscale ≈ 0.1256 . This kernel is strictly decreasing, so ?? is satisfied resulting in a learned lengthscale of around 0.1256. The Matérn-5/2 kernel is a kernel with correlation weakened by

distance (Definition 5.1). In particular, $r = 4.485 * 0.1256 = 0.563316 \Rightarrow \delta \leq 0.002$ (Appendix B), noticing that $\delta = 0.002 \Rightarrow \Phi\left(\frac{N\delta/\sigma_q^2 - T}{\sqrt{1 - (\sqrt{N}\delta/\sigma_q)^2}}\right) \approx \Phi(2)$. ~~When the safety tolerance is set to Consider $\beta^{1/2} = 2$, we then it is safe only when the safety likelihood is above $\Phi(2)$. We~~ can thus know from Theorem 5.2 that safe regions that are 0.563316 further from the observed ones are always identified as unsafe and is not explorable. In Figure 2, the two safe regions are more than 0.7 distant from each other, indicating that the right safe region is never explored by conventional safe learning methods. ~~Please see ?? for numerical details and additional illustrations.~~

GP might even explore less in practice: Our probability bound $\Phi\left(\frac{N\delta/\sigma_q^2 - T}{\sqrt{k_{scale}^j - (\sqrt{N}\delta/\sigma_q)^2}}\right)$ $\Phi\left(\frac{N\delta/\sigma_{q_j}^2 - T_j}{\sqrt{k_{scale}^j - (\sqrt{N}\delta/\sigma_{q_j})^2}}\right)$ (for each $j = 1, \dots, J$) is the worst case obtained with very mild assumptions. Empirically, the explorable regions found by GP models are smaller (see Figure 2 and appendix ??).

6 Modularized GP transfer learning

Transfer learning may overcome the local exploration: We extended the Example 5.4 to compare the standard GP model against a transfer task GP (Section 3.2). In Figure 3, a linear model of corregionalization is trained (a kind of multitask GP, Álvarez et al. (2012)). On the right region, which is beyond the explorable bound of a standard single task GP, the transfer task GP incorporates the source data allowing to build high safety confidence. As a result, the right region can be included into the explorable safe set (next Section 6). We also confirm in our experiments in Section 7 that guidance from source data enables our new safe transfer AL framework to explore beyond the immediate neighborhood of the target points $\mathbf{x}_{1:N}$.

~~In the previous section, we introduced GP safe learning technique, and we analyzed the local exploration problem. In this section, we present our transfer learning strategy, where the aim is to facilitate safe learning and to enable global exploration if properly guided by the source data. To summarize, we see in this section that the safe set of standard GPs (Equation (6)) is limited to a local region. In the next section, we transfer knowledge from the source data to expand the exploration beyond the seed dataset of the target task.~~

~~**Modeling the data with source knowledge:** The idea is to extend the GPs to multi-output models (Journel & Huijbregts, 1976; Álvarez et al., 2012; Tighineanu et al., 2022). We have source task index s , as described previously. We say here that t is the index of our target task (note that we have only one target task). For example, $t = 0$ can be target task, $s = 1, \dots$ can be indices of multiple source tasks. In our main paper, we consider one source task, so the task indices s, t are just binary. Scenarios of more source tasks are provided in ???. We concatenate the source and target tasks and then define notation $\mathbf{f} : \mathcal{X} \times index_space \rightarrow \mathbb{R}$ and $\mathbf{q}^j : \mathcal{X} \times index_space \rightarrow \mathbb{R}$, where $\mathbf{f}(\cdot, s) = f_s(\cdot)$, $\mathbf{f}(\cdot, t) = f(\cdot)$, $\mathbf{q}^j(\cdot, s) = q_s^j(\cdot)$ and $\mathbf{q}^j(\cdot, t) = q^j(\cdot)$. Please also see Table 1 for the summary of our notation. The multi-task functions can then be modeled with GP as well.~~

~~$\mathbf{f} \sim \mathcal{GP}(\mathbf{0}, k_{\mathbf{f}})$ and $\mathbf{q}^j \sim \mathcal{GP}(\mathbf{0}, k_{q^j})$ for some stationary kernels $k_{\mathbf{f}}, k_{q^j} : (\mathcal{X} \times index_space) \times (\mathcal{X} \times index_space) \rightarrow \mathbb{R}$.~~

6 Safe Transfer Active Learning and Source Pre-Computation

~~Then, we can concatenate the data with task indices and express the GP predictive distributions. Recall first that N_{source} is the number of~~ In this section, we formulate our safe transfer AL method. We start from leveraging the source data and multitask GPs (introduced in Section 3.2) to adapt Algorithm 1. We state the resulting new constraint optimization problem for safe transfer AL. We then explain the complexity and consider a modular computation to facilitate the algorithm. We conclude the Section by describing our kernel choices for the experiments.

0.2in

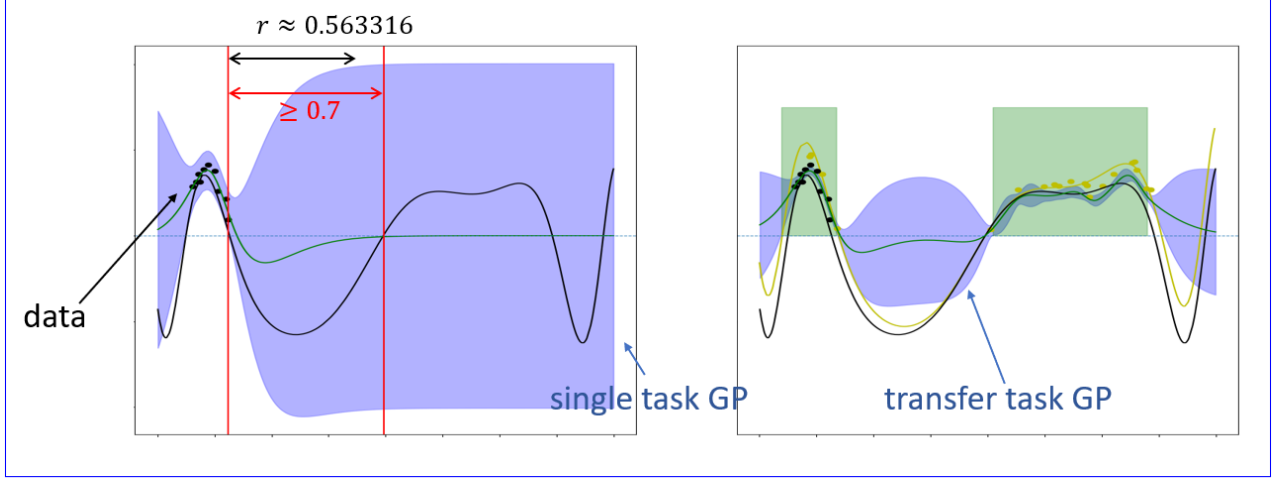


Figure 3: The same safety constraint as in Figure 2 with two safe regions. Left: the single task GP cannot reach the right safe region as the distance is greater than the radius r . Right: The multitask GP is able to exploit knowledge from the source data and build high safety confidence on the right region. The source data comes from the function $g_s(x) = \sin(10x^3 - 5x - 10) + \sin(x^2) - \frac{1}{2}$ and is shown in yellow.

-0.2in

6.1 Safe AL with Transfer Task GPs

We employ multitask GPs to model the target task jointly with the source data. Let $\hat{\mathbf{x}}_{s,1:N_{source}} := \{(\mathbf{x}_{s,i}, s) | \mathbf{x}_{s,i} \in \mathbf{x}_{s,1:N_{source}}\}$ and $\hat{\mathbf{x}}_{1:N} := \{(\mathbf{x}_i, t) | \mathbf{x}_i \in \mathbf{x}_{1:N}\}$ denote the input data concatenated with the task indices. $\hat{\mathbf{x}}_* := (\mathbf{x}_*, t)$ is a test point with As introduced in Section 3.2, the multitask functions $\mathbf{g} \in \{\mathbf{f}, \mathbf{q}^1, \dots, \mathbf{q}^J\}$ are assumed to be GPs. At an unobserved point $\mathbf{x} \in \mathcal{X}$, $p(\mathbf{g}(\mathbf{x}, t) | \mathcal{D}_N, \mathcal{D}_{N_{source}}^{\text{source}}) = \mathcal{N}(\mu_{\mathbf{g},N}(\mathbf{x}), \sigma_{\mathbf{g},N}^2(\mathbf{x}))$ (t is the target task index. Then the predictive distribution given in Equation (2) becomes (similarly, we write down the distribution for \mathbf{f} , while distributions for \mathbf{q}^j can be obtained by replacing \mathbf{f} with \mathbf{q}^j and y with z^j).

$$\begin{aligned} \underline{\mu_{\mathbf{f},N}(\hat{\mathbf{x}}_*)} &= \underline{\mathbf{v}_{\mathbf{f}}^T \Omega_{\mathbf{f}}^{-1} \begin{pmatrix} y_{s,1:N_{source}} \\ y_{1:N} \end{pmatrix}}, \\ \underline{\sigma_{\mathbf{f},N}^2(\hat{\mathbf{x}}_*)} &= \underline{k_{\mathbf{f}}(\hat{\mathbf{x}}_*, \hat{\mathbf{x}}_*) - \mathbf{v}_{\mathbf{f}}^T \Omega_{\mathbf{f}}^{-1} \mathbf{v}_{\mathbf{f}}}, \\ \underline{\mathbf{v}_{\mathbf{f}}} &= \underline{k_{\mathbf{f}} \left(\begin{pmatrix} \hat{\mathbf{x}}_{s,1:N_{source}} \\ \hat{\mathbf{x}}_{1:N} \end{pmatrix}, \hat{\mathbf{x}}_* \right)}, \\ \underline{\Omega_{\mathbf{f}}} &= \underline{\begin{pmatrix} K_{f_s} + \sigma_{f_s}^2 I_{N_{source}} & K_{f_s, f} \\ K_{f_s, f}^T & K_f + \sigma_f^2 I_N \end{pmatrix}} \end{aligned}$$

Equation (3)). The safe set and the acquisition function may then incorporate the source task information:

$$\begin{aligned}
\mathcal{S}_N &= \cap_{j=1}^J \{ \mathbf{x} \in \mathcal{X}_{\text{pool}} \mid \mu_{\mathbf{q}^j, N}(\mathbf{x}) - \beta^{1/2} \sigma_{\mathbf{q}^j, N}(\mathbf{x}) \geq T_j \}, \\
a(\mathbf{x}) &= \sum_{\mathbf{g} \in \{\mathbf{f}, \mathbf{q}^1, \dots, \mathbf{q}^J\}} H_{\mathbf{g}} [\mathbf{x} \mid \mathcal{D}_N, \mathcal{D}_{N_{\text{source}}}^{\text{source}}] \\
&= \sum_{\mathbf{g} \in \{\mathbf{f}, \mathbf{q}^1, \dots, \mathbf{q}^J\}} \frac{1}{2} \log (2\pi e \sigma_{\mathbf{g}, N}^2(\mathbf{x})), \\
\mathbf{x}_* &= \operatorname{argmax}_{\mathbf{x} \in \mathcal{S}_N} a(\mathbf{x}).
\end{aligned} \tag{8}$$

where $K_{f_s} = k_{\mathbf{f}}(\hat{\mathbf{x}}_{s,1:N_{\text{source}}}, \hat{\mathbf{x}}_{s,1:N_{\text{source}}})$, $K_{f_s, f} = k_{\mathbf{f}}(\hat{\mathbf{x}}_{s,1:N_{\text{source}}}, \hat{\mathbf{x}}_{1:N})$ and $K_f = k_{\mathbf{f}}(\hat{\mathbf{x}}_{1:N}, \hat{\mathbf{x}}_{1:N})$. The GP model In contrast to the standard safe AL, $a(\cdot)$ and \mathcal{S}_N here depend on the observed target data \mathcal{D}_N and the source data $\mathcal{D}_{N_{\text{source}}}^{\text{source}}$, as they rely on \mathbf{g} and \mathbf{f} (and \mathbf{q}^j) is governed by the multitask kernel $k_{\mathbf{f}}$ (and $k_{\mathbf{q}^j}$ for each safety function) and noise parameters $\sigma_{f_s}^2, \sigma_f^2$ (and $\sigma_{q_s^j}^2, \sigma_{q^j}^2$) which can be fitted with observations: multitask functions in bold symbols), which represent the multitask GPs based on source and target data.

In this formulation, the covariance bound δ in Theorem 5.2 takes the source input $\mathbf{x}_{s,1:N_{\text{source}}}$ into consideration. Thus, comparing to modeling solely with target task, incorporating a source task provides the potential to significantly enlarge the area with high safety confidence (i.e. region not bounded by Theorem 5.2). We show empirically in ?? that global exploration is indeed easier to achieve with appropriate $\mathbf{x}_{s,1:N_{\text{source}}}$. The whole learning process is summarized in Algorithm 2. Its computational complexity is dominated by fitting the GPs (line 2). Common fitting techniques include Type II ML, Type II MAP and Bayesian treatment (Snoek et al., 2012; Riis et al., 2022) over kernel and noise parameters (Rasmussen & Williams, 2006). All of these approaches have in common that they require computing the marginal likelihoods,

$$\mathcal{N} \left(\begin{pmatrix} y_{s,1:N_{\text{source}}} \\ y_{1:N} \end{pmatrix} \mid \mathbf{0}, \Omega_{\mathbf{f}} \right) \text{ and } \mathcal{N} \left(\begin{pmatrix} z_{s,1:N_{\text{source}}}^j \\ z_{1:N}^j \end{pmatrix} \mid \mathbf{0}, \Omega_{\mathbf{q}^j} \right),$$

for each safety constraint $j = 1, \dots, J$. In this work, we do not consider Bayesian treatments due to the high computational cost of Monte Carlo sampling. Obtaining $\Omega_{\mathbf{f}}^{-1}$ (and $\Omega_{\mathbf{q}^j}^{-1}, j = 1, \dots, J$) for the marginal likelihood takes $\mathcal{O}((N_{\text{source}} + N)^3)$ time, where N_{source} can be large in our set-up. Moreover, the process must be iterated for $N = N_{\text{init}}, \dots, N_{\text{init}} + N_{\text{query}}$ adding to the computational burden. In the next section, we demonstrate how the computational burden can be significantly reduced by pre-computing the source-specific terms necessary for the matrix inversion.

In our paper, we assume all safety constraints are independent. If this is not the case, one may still model different safety constraints with our notation. This has no impact on any of our arguments. For example, we have J unknown constraints where the first three of the target task q^1, q^2, q^3 are not independent, and the corresponding source q_s^1, q_s^2, q_s^3 may or may not be independent. Then Assumption 3.4 still holds if we consider the safety functions in the following way: we have safety functions q^{123}, q^4, \dots, q^J and q_s^{123}, q_s^4, \dots , where joint functions $q^{123}(\cdot, j = 1, 2, 3) := q^j(\cdot), q_s^{123}(\cdot, j = 1, 2, 3) := q_s^j(\cdot), q^{j>3}(\cdot, j) := q^j(\cdot), q_s^{j>3}(\cdot, j) := q_s^j(\cdot)$. This allows one to model the dependent safety constraints jointly with q^{123} (and q_s^{123}), and can be simply achieved by augmenting the data such that they have not only task indices but also safety constraint indices. If we consider the safety function in this way, we resort the problem back to $J - 2$ independent safety constraints. Thus, dependent safety constraints have no impact to the argument that single-task methods explore locally around the scarce target data while multi-task approaches expand the exploration by incorporating the source knowledge.

In-experiment speed-up via source pre-computation: Computation of $\Omega_{\mathbf{f}}^{-1}$ (and $\Omega_{\mathbf{q}^j}$) has cubic complexity $\mathcal{O}((N_{\text{source}} + N)^3)$ in time, for $N = N_{\text{init}}, \dots, N_{\text{init}} + \text{num_steps}$. This computation

is also required for fitting the models: common fitting techniques include Type II ML, Type II MAP and Bayesian treatment (Snoek et al., 2012; Riis et al., 2022) over kernel and noise parameters (Rasmussen & Williams, 2006), all of which involves computing the marginal likelihoods $\mathcal{N}\left(\begin{pmatrix} y_{s,1:N_{source}} \\ y_{1:N} \end{pmatrix} | \mathbf{0}, \Omega_{\mathbf{f}}\right)$ and $\mathcal{N}\left(\begin{pmatrix} z_{s,1:N_{source}}^j \\ z_{1:N}^j \end{pmatrix} | \mathbf{0}, \Omega_{q^j}\right)$. In our paper, Bayesian treatment is not considered because MC sampling is time-consuming.

The goal now is to avoid

6.2 Source Pre-Computation

In this section, we propose an efficient algorithm to mitigate the computational burden of repeatedly calculating $\Omega_{\mathbf{f}}^{-1}$ and $\Omega_{q^j}^{-1}$ repeatedly in the experiments. For brevity in full. For clarity, we describe how we do this with the approach for $\Omega_{\mathbf{f}}^{-1}$, while the same principle applies the same principles apply to $\Omega_{q^j}^{-1}$ for all $j = 1, \dots, J$.

For GP models, the inversion is matrix inversion is routinely achieved by performing a Cholesky decomposition $L(\Omega_{\mathbf{f}})$, i.e. which has cubic complexity. This decomposes $\Omega_{\mathbf{f}}$ as $\Omega_{\mathbf{f}} = L(\Omega_{\mathbf{f}})L(\Omega_{\mathbf{f}})^T$, where $L(\Omega_{\mathbf{f}})$ is a lower triangular matrix (Rasmussen & Williams, 2006). Once the decomposition is obtained, operations such as $L(\Omega_{\mathbf{f}})^{-1}C$, and then for any matrix C , $L(\Omega_{\mathbf{f}})^{-1}C$ is can be efficiently computed by solving a linear system with minor complexity. The Cholesky decomposition is well known for its numerical stability and computational efficiency, making it a widely preferred approach for efficient GP computations.

We propose to perform the Cholesky decomposition in two steps, as described below. The aim here is to compute part of $L(\Omega_{\mathbf{f}})$ beforehand. The key idea is to cluster the parameters of \mathbf{f} into precompute the source-specific terms of the Cholesky decomposition, which account for a large amount of the computational costs. Importantly, our technique is general and can be applied to any multi-output kernel. Recall from Equation (5) that the covariance $\Omega_{\mathbf{f}}$ has a block structure, in which the source block $K_{f_s} + \sigma_{f_s}^2 I_{N_{source}}$ has size $N_{source} \times N_{source}$ that dominates the computation. The Cholesky decomposition can also be expressed as block structure,

$$L(\Omega_{\mathbf{f}}) = \begin{pmatrix} L_{f_s} & 0 \\ L_{f_s, f} & L_f \end{pmatrix},$$

where the source block L_{f_s} can be precomputed independently of the remaining covariances (Press et al., 1988). Once L_{f_s} is obtained, it is then used to compute the cross-term $L_{f_s, f}$ and target block L_f that are both a function of the source block L_{f_s} (details in Appendix D.1). If the source covariance, $K_{f_s} + \sigma_{f_s}^2 I_{N_{source}}$, remains unchanged between different covariances $\Omega_{\mathbf{f}}$, its precomputed Cholesky decomposition L_{f_s} can be reused, significantly reducing computational overhead.

Algorithm 2 Modularized SL Full safe transfer AL

Require: $\mathcal{D}_{N_{source}}^{source}, \mathcal{D}_{N_{init}}, \mathcal{X}_{pool}$, Assumption 3.1, Assumption 3.3, Assumption 3.4, $\mathcal{D}_{N_{init}}, \mathcal{D}_{N_{source}}^{source}, \mathcal{X}_{pool}, \beta$
or α, N_{query} , thresholds T_1, \dots, T_L

- 1: **for** $N = N_{init}, \dots, N_{init} + N_{query} - 1$ **do**
- 2: Fit GPs and then fix $\mathbf{f}, \mathbf{q}_1, \dots, \mathbf{q}_L$ with $\mathcal{D}_N, \mathcal{D}_{N_{source}}^{source}$
- 3: $\mathcal{S}_N \leftarrow \bigcap_{j=1}^J \{\mathbf{x} \in \mathcal{X}_{pool} | \mu_{q^j, N}(\mathbf{x}) - \beta^{1/2} \sigma_{q^j, N}(\mathbf{x}) > T_j\}$ (Equation (8))
- 4: $\mathbf{x}_* \leftarrow \operatorname{argmax}_{\mathbf{x} \in \mathcal{S}_N} a(\mathbf{x}), a(\mathbf{x}) = \sum_{\mathbf{g} \in \{\mathbf{f}, \mathbf{q}_1, \dots, \mathbf{q}_L\}} H_{\mathbf{f}}[\mathbf{x} | \mathcal{D}_N, \mathcal{D}_{N_{source}}^{source}]$
- 5: Query \mathbf{x}_* to get evaluations y_* and z_*
- 6: $\mathcal{D}_{N+1} \leftarrow \mathcal{D}_N \cup \{\mathbf{x}_*, y_*, z_*\}, \mathcal{X}_{pool} \leftarrow \mathcal{X}_{pool} \setminus \{\mathbf{x}_*\}$
- 7: **end for**
- 8: **Return** $\mathcal{D}_{N_{init}+N_{query}}$, trained models $\mathbf{f}, \mathbf{q}_1, \dots, \mathbf{q}_L$

Fixed source parameters for efficient training: To leverage the precomputed Cholesky decomposition L_{f_s} during our safe transfer AL scheme, the parameters governing the source covariance, $K_{f_s} + \sigma_{f_s}^2 I_{N_{\text{source}}}$, must remain fixed throughout the algorithm.

To achieve this, we split the kernel parameters θ_f into two groups, $\theta_f = (\theta_{f_s}, \theta_f)$, where the source $k_f((\cdot, s), (\cdot, s))$ is independent of θ_f . Then, as $\mathbf{x}_{s,1:N_{\text{source}}}$ is invariant, where θ_{f_s} include all parameters required for computing K_{f_s} adapts only to θ_{f_s} . Given that the source tasks are well explored, the source likelihood $p(y_{s,1:N_{\text{source}}} | \mathbf{x}_{s,1:N_{\text{source}}}) = \mathcal{N}(y_{s,1:N_{\text{source}}} | \mathbf{0}, K_{f_s} + \sigma_{f_s}^2 I_{N_{\text{source}}})$ can be barely increased while we explore for the target task. Thus we assume K_{f_s} (i.e. θ_{f_s}) and $\sigma_{f_s}^2$ remain fixed in the experiments, this allows us to isolate the source relevant computations, as the source relevant block (top left block) of $L(\Omega_f)$ is also fixed. We can then prepare a safe learning experiment with pre-computed $L_{f_s} = L(K_{f_s} + \sigma_{f_s}^2 I_{N_{\text{source}}})$. The same procedure applies to each q^i . The θ_f contains the remaining parameters. We first train on the source data $\mathcal{D}_{N_{\text{source}}}^{\text{source}}$ alone, then fix θ_{f_s} and $\sigma_{f_s}^2$. Once these parameters are fixed, the Cholesky decomposition L_{f_s} can be precomputed and reused across all subsequent iterations when acquiring the target dataset \mathcal{D}_N . During this phase, we can still update the parameters θ_f and the target noise variance σ_f^2 .

The learning procedure is summarized in Algorithm 3. In each iteration (line 4 of Algorithm 3), the time complexity becomes $\mathcal{O}(N_{\text{source}}^2 N) + \mathcal{O}(N_{\text{source}} N^2) + \mathcal{O}(N^3)$ instead of $\mathcal{O}((N_{\text{source}} + N)^3)$ reduces from $\mathcal{O}((N_{\text{source}} + N)^3)$ to $\mathcal{O}(N_{\text{source}}^2 N) + \mathcal{O}(N_{\text{source}} N^2) + \mathcal{O}(N^3)$. We provide mathematical details in the Appendix D.1. Our technique can be applied to any multi-output kernel because the clustering $\theta_f = (\theta_{f_s}, \theta_f)$ does not require independence of $k_f((\cdot, s), (\cdot, t))$ and $k_f((\cdot, t), (\cdot, t))$ from θ_{f_s} . The same principle applies to q^i . Appendix D.1. Note that our approach offers a trade-off: it reduces parameter flexibility in exchange for computational efficiency. We will discuss the pros and cons of this approach, depending on the kernel choice, in more detail in the following section.

Kernel selection: In the following, we briefly review existing multi-output GP models and motivate selection of the model we use later in our experiments.

6.3 Kernel Selection

Multitask kernels are often defined as a matrix of functions (Journel & Huijbregts, 1976; Álvarez et al., 2012), where each element maps $\mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ similar to a standard kernel. The task indices determine which element of the matrix is used. Specifically, for task indices $i, i' \in \{s, t\}$, the kernel can be expressed as

$$k_g((\cdot, i), (\cdot, i')) = \begin{pmatrix} k_g((\cdot, s), (\cdot, s)) & k_g((\cdot, s), (\cdot, t)) \\ k_g((\cdot, t), (\cdot, s)) & k_g((\cdot, t), (\cdot, t)) \end{pmatrix}_{i, i'},$$

Algorithm 3 Modularized safe transfer AL

Require: Assumption 3.1, Assumption 3.3, Assumption 3.4, $\mathcal{D}_{N_{\text{init}}}$, $\mathcal{D}_{N_{\text{source}}}^{\text{source}}$, $\mathcal{X}_{\text{pool}}$, β or α , N_{query} , thresholds T_1, \dots, T_J

- 1: Fit GPs f, q_1, \dots, q_J with $\mathcal{D}_{N_{\text{source}}}^{\text{source}}$
 - 2: Fix source specific parameters $\theta_{f_s}, \theta_{q_s^j}, \sigma_{f_s}, \sigma_{q_s^j}, \forall j = 1, \dots, J$
 - 3: Compute and fix $L_{f_s}, L_{q_s^j}, \forall j = 1, \dots, J$ (line 5, 6, 7 below faster)
 - 4: **for** $N = N_{\text{init}}, \dots, N_{\text{init}} + N_{\text{query}} - 1$ **do**
 - 5: Fit GPs (remaining with \mathcal{D}_N and $\mathcal{D}_{N_{\text{source}}}^{\text{source}}$ (free parameters $\theta_f, \theta_{q^j}, \sigma_f, \sigma_{q^j}, \forall j = 1, \dots, J$)
 - 6: $\mathbf{x}_* \leftarrow \arg\max_{\mathbf{x} \in \mathcal{S}_N} a(\mathbf{x} | \mathcal{D}_N)$, $\mathcal{S}_N \leftarrow \bigcap_{j=1}^J \{\mathbf{x} \in \mathcal{X}_{\text{pool}} | \mu_{q^j, N}(\mathbf{x}) - \beta^{1/2} \sigma_{q^j, N}(\mathbf{x}) > T_j\}$ (Equation (8))
 - 7: Evaluate at \mathbf{x}_* , $\arg\max_{\mathbf{x} \in \mathcal{S}_N} a(\mathbf{x})$, $a(\mathbf{x}) = \sum_{g \in \{f, q^1, \dots, q^J\}} H_f[\mathbf{x} | \mathcal{D}_N, \mathcal{D}_{N_{\text{source}}}^{\text{source}}]$
 - 8: Query \mathbf{x}_* to get evaluations y_* and z_*
 - 9: $\mathcal{D}_{N+1} \leftarrow \mathcal{D}_N \cup \{\mathbf{x}_*, y_*, z_*\}$, $\mathcal{X}_{\text{pool}} \leftarrow \mathcal{X}_{\text{pool}} \setminus \{\mathbf{x}_*\}$, $\mathcal{D}_{N+1} \leftarrow \mathcal{D}_N \cup \{\mathbf{x}_*, y_*, z_*\}$, $\mathcal{X}_{\text{pool}} \leftarrow \mathcal{X}_{\text{pool}} \setminus \{\mathbf{x}_*\}$
 - 10: **end for**
 - 11: **Return** $\mathcal{D}_{N_{\text{init}}+N_{\text{query}}}$, trained models f, q_1, \dots, q_J
-

where the dots \cdot are placeholders for input data from \mathcal{X} . Here, each $\mathbf{g} \in \{\mathbf{f}, \mathbf{q}^1, \dots, \mathbf{q}^J\}$ is a multi-output GP correlating source and target tasks. The task indices are binary: $s = 0$ is source and $t = 1$ is target, for the main and safety functions.

Linear model of coregionalization (LMC): A widely investigated multi-output framework is the linear model of ~~coregionalization~~ coregionalization (LMC) $\div k_{\mathbf{g}} = \sum_l \begin{pmatrix} W_{l,s}^2 + \kappa_s & W_{l,s}W_{l,t} \\ W_{l,s}W_{l,t} & W_{l,t}^2 + \kappa_t \end{pmatrix} \otimes k_l(\cdot, \cdot)$, i.e. a 2×2 matrix specified by task indices, where $k_l(\cdot, \cdot)$ is a standard kernel as in Assumption 3.2, and $(W_l W_l^T + \text{diag}(\kappa_s, \kappa_t))$ learns the task correlation which we also use in our experiments. In our setup, the kernel is defined as

$$k_{\mathbf{g}}((\cdot, \{s, t\}), (\cdot, \{s, t\})) = \sum_{l=1}^2 \left(W_l W_l^T + \begin{pmatrix} \kappa_s & 0 \\ 0 & \kappa_t \end{pmatrix} \right) \otimes k_l(\cdot, \cdot),$$

where \otimes denote the Kronecker product. We assume two latent effects and each latent effect is specified by the base kernel $k_l(\cdot, \cdot) : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$. The parameters $W_l \in \mathbb{R}^{2 \times 1}$, $\kappa_s, \kappa_t > 0$ model the task correlations induced by the l -th latent function pattern encoded by k_l (Álvarez et al., 2012). Here, each \mathbf{g} has its own kernel, but for brevity, we omit \mathbf{g} in the parameter subscripts for brevity. When pairing this kernel with our ??, we observe that the. Furthermore, if k_l includes a variance scaling term, e.g. Matérn kernels, it is fixed to 1 because the scale can be absorbed into W_l , κ_s and κ_t .

This kernel design is tied to our experimental setup and facilitates the transfer of information from the source to the target task. However, when paired with Algorithm 3, training can become unstable due to multiple local optima in the first phase (line 1 of ??). This may be because LMClearns joint patterns from all present tasks, because the algorithm assumes that the kernel parameters can be cleanly separated between source and task terms. In the case of the LMC, this separation is not straightforward: The latent components W_l encode shared task correlations, while κ_s and κ_t represent task-specific effects. Training on source data alone provides insufficient information to disentangle these shared and individual contributions, potentially leading to suboptimal solutions that destabilize the training process.

Hierarchical GP (HGP): In Poloczek et al. (2017); Marco et al. (2017); Tighineanu et al. (2022), the authors consider a hierarchical GP (HGP) $\div k_{\mathbf{g}} = \begin{pmatrix} k_s(\cdot, \cdot) & k_s(\cdot, \cdot) \\ k_s(\cdot, \cdot) & k_s(\cdot, \cdot) + k_t(\cdot, \cdot) \end{pmatrix}$. Similarly, each \mathbf{g} has its own kernel, but we omit \mathbf{g} in the parameter subscripts for brevity framework, where the kernel is defined as:

$$k_{\mathbf{g}}((\cdot, \{s, t\}), (\cdot, \{s, t\})) = \begin{pmatrix} k_s(\cdot, \cdot) & k_s(\cdot, \cdot) \\ k_s(\cdot, \cdot) & k_s(\cdot, \cdot) + k_t(\cdot, \cdot) \end{pmatrix},$$

with $k_s, k_t : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ as base kernels. HGP is a variant of LMC, where the target task is treated as a modeled as the sum of the source (modeled by kernel k_s) and the target-source residual (modeled by and the target-specific residual k_t) (Tighineanu et al., 2022). This formulation has the benefit that the fitting of source (terms k_s) and residual (k_t) are separated and thus makes HGP a good model to run ?? (set θ_{g_s} the parameters of k_s and θ_{g_t} the parameters of k_t) can be easily decoupled, making HGP particularly well suited for the use of Algorithm 3.

In Tighineanu et al. (2022), the authors derived an ensembling technique allowing also for a that also supports source pre-computation. Their technique While their approach is equivalent to our method when we use HGP, but our approach can be generalized applied to HGP, our framework generalizes to any multi-output kernels (with implicit restriction that a source kernel, provided that the fitting of the chosen model needs to be accurate) while source and target parameters can be decoupled. In contrast, the ensembling technique is limited to HGP, explicitly tailored to HGP and does not generalize to other kernel structures.

Kernel selection in our experiments: In our experiments, we perform ?? Algorithm 3 with HGP as our main pipeline method, and Algorithm 1 with LMC (more flexible in learning yet slow) and with HGP

as full transfer scenarios. The base kernels k_s, k_t, k_l are all Matérn-5/2 kernel kernels with D lengthscale parameters ($\mathcal{X} \subseteq \mathbb{R}^D$). The scaling variance of k_l is fixed to 1 because it can be absorbed into the output covariance terms (see above). One can of course change the base kernel as long as it is suitable for the application. This choice is not restrictive and can be replaced with other base kernels suited to specific applications.

Although we did not pair Algorithm 3 with LMC as discussed above, note that our modularized computation scheme can still benefit the general LMC provide benefits in closely related settings, e.g. (i) datasets in which more than one source task is available with multiple sources or (ii) sequential learning schemes that only refit the GPs after receiving frameworks where GPs are refitted only after a batch of query points has been acquired.

7 Experiments

0.2in

Safe AL experiments on Hartmann3 and two types of engine modeling problems. Hartmann3: $N_{source} = 100$, N is from 20 to 120, results are mean and one standard error of 25 experiments. PEngine: $N_{source} = 500$, N is from 20 to 120, results are mean and one standard error of 5 repetitions. GEngine: $N_{source} = 500$, N is from 20 to 220, plotted every 50 queries, results are mean and one standard error of 5 repetitions. Please see Figure 5 for the zoom-in RMSE plot of GEngine.

Number of discovered regions **methods** GP1D+z GP2D+z Branin num_steps 50 100 100 EffTransHGP 1.79 ± 0.07 2.77 ± 0.13 2 ± 0 FullTransHGP 1.78 ± 0.07 3 ± 0.14 213 2 ± 0 FullTransLMC 1.78 ± 0.08 2.68 ± 0.14 2 ± 0 Rothfuss2022 1.22 ± 0.05 1.07 ± 0.03 1 ± 0 SAL 1 ± 0 1.29 ± 0.09 1 ± 0 Transfer learning discovers multiple disjoint safe regions while baselines stick to neighborhood of the initial region. In appendix Figure 11, we track the number of explored regions per iteration.

Training time of f and g datasets EffTransHGP FullTransHGP FullTransLMC Rothfuss2022 SAL(N_{source}, N) GP1D+z 8.947 ± 0.198 9.171 ± 0.133 26.56 ± 0.628 0.0 ± 0.0 6.881 ± 0.083 (100, 10 + 50) GP2D+z 10.73 ± 0.190 39.31 ± 0.639 202.8 ± 12.43 0.0 ± 0.0 8.044 ± 0.142 (250, 20 + 100) Branin 3.754 ± 0.121 8.129 ± 0.267 21.16 ± 1.207 0.0 ± 0.0 4.691 ± 0.078 (100, 20 + 100) Hartmann3 3.662 ± 0.089 9.092 ± 0.467 34.43 ± 1.664 0.0 ± 0.0 4.073 ± 0.083 (100, 20 + 100) PEngine 9.596 ± 0.418 124.99 ± 5.608 615.7 ± 27.99 0.0 ± 0.0 4.686 ± 0.243 (500, 20 + 100) GEngine 18.525 ± 2.508 503.11 ± 63.94 4357.8 ± 661.4 0.0 ± 0.0 10.485 ± 0.578 (500, 20 + 150)

In this section, we perform safe AL experiments to answer the following questions: **1)** do multi-output GPs facilitate In this section, we empirically evaluate our approach against state-of-the-art competitors on a range of synthetic and real-world datasets. We first provide details on the experimental setup in Section 7.1. Then, we analyze whether our transfer learning scheme is more data-efficient than conventional methods in Section 7.2, whether it facilitates the learning of disconnected safe regions, **2)** is it more data efficient to learn with transfer safe learning than applying a conventional method, and **3)** how is in Section 7.3, and how the runtime of our modularized approach compared with the baseline? compares in Section 7.4.

Our code will be published on GitHub after acceptance.

7.1 Experimental Details

First, we describe comparison partners and the datasets we use in our experiments.

7.1.1 Comparison Partners

We compare five ~~experimental setups~~ different methods: 1) EffTransHGP: ~~??~~ Algorithm 3 with multi-output HGP, 2) FullTransHGP: ~~Algorithm 1~~ Algorithm 2 with multi-output HGP, 3) FullTransLMC: ~~Algorithm 1~~ Algorithm 2 with multi-output LMC, 4) ~~Rothfuss et al. 2022~~ Rothfuss2022: GP model meta learned with the source data by applying Rothfuss et al. (2022), and 5) SAL: the conventional Algorithm 1 with single-output GPs ~~and Matérn-5/2 kernel~~.

~~The first three methods are our proposed approaches, listed in order of increasing complexity. The HGP kernel is a variant of the LMC kernel. We test two variations of the HGP: one using our modularized implementation (Algorithm 3), with a runtime complexity comparable to the single-task approach, and another one using a naive implementation (Algorithm 2) that has a similar runtime complexity as LMC. For the safety tolerance, we always fix $\beta = 4$, i.e. which corresponds to $\alpha = 1 - \Phi(\beta^{1/2}) = 0.02275$ (Equations (6) and (8)), implying that each fitted GP safety model allows 2.275% unsafe tolerance. Notice that with For the baseline following Rothfuss et al. (2022), the GP model parameters are meta learned up-front with using source data, and remain fixed during throughout the experiments. Rothfuss et al. 2022 considered While the authors of the original paper applied this approach to safe BO problems. We change, we modify the acquisition function to entropy so it becomes, transforming it into a safe AL method. Our code will be published on GitHub.~~

~~**Track safe regions:** We start from 3 simple simulated problems with input dimension $D=1$ or $D=2$ (GP1D, GP2D, Branin problems). In such cases, it is analytically and computationally possible to cluster the disconnected safe regions via connected component labeling (CCL) algorithms (He et al., 2017). This means, in each iteration of the experiments, we track to which safe region each observation belongs. In these initial experiments, we generate one source dataset and one target dataset such that the target task has at least two disjoint safe regions, each of which has a portion also safe in the source problem. The design is due to the selection of our kernels. Our base kernel, the Mat All methods use Matérn-5/2 kernel, correlates closeness of data points, and LMC and HGP rescale the Matérn-5/2 kernel measures for different tasks, which means patterns of the same area in the space are transferred. Modeling a more complicated transferring pattern, e.g. correlation on an abstract feature space, may require a careful selection of an appropriate base kernel (see e.g. Bitzer et al. (2022)). kernels as the base kernels.~~

~~**General test and real-world problems, no safe regions clustering:** We further consider 3 problems **Hartmann3** ($D=3$), **PEngine** ($D=2$) and **GEngine** ($D=13$). **PEngine** datasets have noisy measurements where true grid values cannot be accessed, which makes the CCL algorithm inaccurate. The safe region clustering is thus not performed in this problem. **Hartmann3** has higher dimension, so it is computationally not possible to cluster the safe regions. **GEngine** datasets have noisy measurements and are high dimensional. The CCL algorithm cannot be leveraged.~~

~~**Metrics:** The learning result of \mathbf{f} is shown as RMSEs between the GP predictive mean and test y sampled from true safe regions. To measure the performance of \mathbf{q} , we use the area of \mathcal{S}_N , as this indicates the explorable coverage of the space. In particular, we look at the area of $\mathcal{S}_N \cap \mathcal{S}_{true}$ (true positive or TP area, the larger the better) and $\mathcal{S}_N \cap (\mathcal{X} \setminus \mathcal{S}_{true})$ (false positive or FP area, the smaller the better). Here, $\mathcal{S}_{true} \subseteq \mathcal{X}_{pool}$ is the set of true safe candidate inputs, and this is available since our datasets in the experiments are prepared as executed queries. With~~

7.1.2 Datasets

We benchmark our methods on six datasets. An overview of the datasets is given in Table 2.

7.2 AL results on problems with tractable safe regions

~~**Datasets:** We adapt~~

Table 2: Dataset Summary: For each dataset, we list the input dimension D , the size of the source dataset N_{source} , the size of the initial target dataset N_{init} , the number of queries N_{query} , description, safety threshold and whether the disjoint safe regions can be tracked. Datasets are listed in order of increasing complexity. Each task has one safety variable.

<u>Dataset</u>	<u>D</u>	<u>N_{source}</u>
GP1D	1	1
GP2D and Branin data, CCL (He et al., 2017) is performed to cluster which safe region each query belongs to	2	2
Branin	2	1
Hartmann3	3	1
PEngine	2	5
GEngine	13	5

Synthetic Datasets: We first create two low-dimensional synthetic datasets, GP1D ($D = 1$) and GP2D ($D = 2$), generating multi-output GP samples following algorithm 1 of Kanagawa et al. (2018) to generate multi-output GP samples. The first output is treated as our source task and the second output as the target task. Each dataset has a main function f and an additional safety function q . Numerical details and example datasets are plotted in ???. We generate 10 datasets and repeat the AL experiments each experiment five times for each method on every dataset. For Branin data, we take the numerical setting the Branin dataset, we follow the settings from Rothfuss et al. (2022); Tighineanu et al. (2022) to generate five different datasets. With each dataset, we repeat the experiments for five times.

Result: In ??, we show the results of produce five datasets and run five repetitions for each method on each dataset. Unlike GP1D and GP2D, Branin uses the same function for both main and safety tasks. In these initial experiments, we simulate multiple datasets but retain only those in which the target task has at least two disjoint safe regions, GP2D and of Branin data. We see that EffTransHGP, FullTransHGP and FullTransLMC experiments achieve accurate and much larger safe set coverage (larger TP area and small FP area). In addition, the learning of f is more efficient with EffTransHGP, FullTransHGP and FullTransLMC as the RMSE drops faster compared to the baseline methods. Note that the test points are sampled from all of the true safe area, including the part baseline SAL fails to explore. It is thus not guaranteed that RMSE of SAL monotonically decreases (Branin). We observe from the experiments that the meta learning approach, Rothfuss et al. 2022, fails to generalize to larger area, which might be due to a lack of data in target task representativeness (one source, very few for meta learning) or/and in quantity.

In Table 3, we count the number of safe regions explored by the queries. This confirms the ability to explore disjoint safe regions. One remark is that Branin function is smooth and has two clear safe regions; while huge stochasticity exists in GP data and we may have various number of small or large safe regions scattered in the space. Table 4 shows the model fitting time, confirming that EffTransHGP has comparable time complexity as baseline SAL, as opposed to FullTransHGP and FullTransLMC. We provide additional ratios of safe queries in appendix Table 5, which is a sanity check that the methods are indeed safe.

Please note the learning flexibility is FullTransLMC $>$ FullTransHGP $>$ EffTransHGP, and our experimental results are consistent to this intuition (RMSE of FullTransLMC in 1D data is worse because we starts with 10 data points which is less than the number of LMC parameters, ??).

In the main experiments, N_{source} (the number of source data points) is fixed for each problem. In our ??, we provide ablation studies on with each disjoint region also having a safe counterpart in the Branin dataset, where we vary the number of source data points and number of source tasks.

7.2 AL results on general test and real-world problems

Hartmann problem: We take the numerical setting source dataset. This design aligns with our use of the Matérn-5/2 kernel, which measures similarity between data points based on proximity. Our fourth synthetic

dataset is the Hartmann3 dataset ($D = 3$), created using the settings from Rothfuss et al. (2022); Tighineanu et al. (2022) to generate five different Hartmann3 datasets. Here the source task. We generate five datasets and repeat experiments on each datasets five times. Here, the source data and the initial target data are all and initial target datasets are sampled randomly, in contrast to unlike the structured, disjoint safe regions in GP1D, GP2D and Branin, where we enforce disjoint safe regions and overlapped source and initial target region.

With each dataset, we repeat the experiments for five times. Please see ?? for details. In this experiment, EffTransHGP, FullTransLMC and FullTransHGP provide much smaller RMSEs and larger safe area, and Branin. All datasets are normalized, and the constraint thresholds are set to zero. Further details on our synthetic datasets are provided in Appendix E.2.

Semi-Real-World Dataset (PEngine datasets): This is a real-world problem, with interpolation performed for our experiments. We have two datasets, measured from the same prototype of engine under different conditions. Both datasets measure the The PEngine dataset consists of two datasets measured on the same engine prototype under varying conditions. The outputs temperature, roughness, emission HC, and emission NOx NOx emissions are recorded. We perform independent separate AL experiments to learn about roughness roughness (Figure 4) and temperature (put in appendix Appendix Figure 12), both constrained by the normalized temperature values $q \leq 1.0$. The safe set is around 0.5293 of the entire space. The raw datasets have a normalized temperature q , threshold on noisy observation $z \leq 1.0$, resulting in a safe set covering approximately 52.93% of the input space.² The upper bound constraint is equivalent to $-z \geq -1.0$ as described in our Section 2, $-z$ being the negative noisy temperature. The raw datasets contain four input variables: two free variables and two contextual inputs which are supposed to be fixed, i.e. a total of $D = 4$ inputs. The contextual inputs are variables, with the contextual inputs recorded with noise, so we interpolate the values with. To fix the contextual inputs at constant values, we interpolate these noisy values using a multi-output GP simulator, trained on the full datasets. Thus this experiment is performed on a This allows us to perform active learning experiments solely on the two-dimensional space of the free variables, creating a semi-simulated condition (free input variables $D = 2$). Details are given in ?? environment. Further details are provided in Appendix E.2 and we will provide a GitHub link to the dataset after acceptance.

The safe set of this target task is not clearly separated into multiple disjoint regions. Thus the conventional method can eventually identify most part of the safe area. Nevertheless, we still see a much better RMSEs and much less data consumption for large safe set coverage. We also observe that Rothfuss et al. 2022 failed to generalize the meta-learned source knowledge to the entire target space exploration.

GEngine datasets: Next, we apply our method to

Real-World Dataset (GEngine): Our final benchmark is a high-dimensional, real-world problem consisting of involving two datasets, each recorded by a related but distinct engine, with one serving (one as the source task and the other as the target task Both datasets were published by) from Li et al. (2022). Each dataset is The original dataset are split into training set and test set. We use the source training set and target training set to run our AL experiments, and we use test sets, and we conduct AL experiments on the training sets, while RMSE and safe set performance are evaluated on the target test set to evaluate the RMSE and the safe set. The. These datasets are dynamic, and our model applies a history structure is applied, concatenating the by concatenating relevant past points into the inputs, which results resulting in an input dimension of $D = 13$. We provide further details in ??.

This problem measures the The recording include emissions and temperatures as well. We, and we learn the normalized emission (f), subject to normalized temperature $-1.5 \leq q \leq 0.5$. This constraint is equivalent to two safety conditions $-1.5 \leq q$ and $-0.5 \leq -q$. q , threshold on noisy value $-1.5 \leq z \leq 0.5$.² The upper bound on temperature is crucial for safety, while the lower bound increases robustness against outliers. Overall, the safe region covers approximately 65% of the target dataset. For the source tasks, the constraint is $-2 \leq q_s \leq 0.5$. The temperature lower bound matters mainly to the outlier task, we sample the data

² In general, we use the notation $z = \{z^1, \dots, z^J\}$ to represent J safety constraints. However, since all datasets in our experiments involve only a single safety constraint, we simplify the notation to z .

under a different constraint of $-2 < z_s \leq 0.5$ to make the model more resistant to outliers. More details can be found in Appendix E.2.

7.2 Modeling Performance & Safety Coverage

In the following, we study the empirical performance of our algorithms to find out whether our methods can accelerate space exploration and model convergence while maintaining safety.

Metrics: We evaluate model convergence of the main function f using root mean square error (RMSE) between the GP predictive mean and test y sampled from true safe regions. To measure the performance of our safety function q , we use the area of \mathcal{S}_N (Equation (6)), as this indicates the explorable coverage of the space. Specifically, we consider the area of $\mathcal{S}_N \cap \mathcal{S}_{\text{true}}$ (true positive or TP area, the larger the better) and $\mathcal{S}_N \cap (\mathcal{X}_{\text{pool}} \setminus \mathcal{S}_{\text{true}})$ (false positive or FP area, the smaller the better). Here, $\mathcal{S}_{\text{true}} \subseteq \mathcal{X}_{\text{pool}}$ denotes the set of true safe candidate inputs, which we can precompute as we use a fixed data pool. Area of $\mathcal{X}_{\text{pool}}$, \mathcal{S}_N , $\mathcal{S}_{\text{true}}$ are all measured by counting the number of points.

Results: We report results in Figure 4.

Results on GP1D, GP2D, Branin: We begin by focusing on the GP1D, it is the upper bound 0.5 that plays the major role. Overall, such constraint means around 65% of the target dataset is safe. We provide an appendix Figure 8 to illustrate the raw data distributions. In this problem, the effect of one single query on the GP hyperparameters is not obvious. Therefore, to speed up the experiments, we train the hyperparameters only every 50 queries (and at the beginning). We query for 200 iterations. GP2D, and Branin datasets, which have been simulated to contain multiple disconnected safe regions. On these datasets, only methods capable of jumping between regions can achieve optimal performance. In Figure 4, we observe that our transfer learning approaches achieve lower RMSE and significantly greater safe set coverage than competing methods, while maintaining small false detection rate of safe area. These results suggest that our methods can successfully identify and explore disconnected safe regions, while our competitor methods cannot. We will conduct an in-depth analysis of this aspect in the next section. The higher RMSE of our competing methods can be partially attributed to the evaluation approach: test points are sampled from the entire safe area, including regions that competing methods fail to explore. Additional safe query ratios, provided in Appendix Table 5, confirm that our methods maintain high levels of safety.

Overall, we observe a clear outperformance of

Results on Hartmann, PEngine: In the Hartmann and PEngine experiments, our transfer learning approaches demonstrate superior performance, achieving lower RMSEs and broader safe area coverage with fewer data points than competing methods (see Figure 4). Since SAL eventually covers the entire safe area by the end of the iterations, we hypothesize that the target task do not contain clearly separated disjoint regions. Nonetheless, conventional SAL requires more queries to achieve the same performance, as they lack the efficiency of our approach.

Results on GEngine: Our final dataset, GEngine, has a larger input space, resulting in more hyperparameters and making GP fitting more computationally expensive (see also Table 4). Given that each individual query minimally affects the GP hyperparameters, we update them every 50 queries to enhance runtime efficiency and report results at the same interval. Overall, the HGP-based multitask approaches transfer learning approaches clearly outperform competitors, as they explore the safe set with significantly fewer target task queries while being better or at least as good as their competitors in terms of achieving better or equal test error and proportion of false positives. As shown in ??, the HGP methods, particularly EffTransHGP, identify around two-thirds of the safe set with 50 queries, and they have almost no false safe classifications. In contrast, with the same number of queries, the baseline SAL discovers less than half of the safe set. From the RMSEs, HGP false safe positive rates. Zooming into the RMSE results in Figure 5, we find that the HGP approaches learns the main function as well as the baseline SAL. In this problem, training. Training the LMC model seems, however, appears to be more challenging. Only, only after the second training iteration (iteration 100) does the RMSE of FullTransLMC stabilize, the RMSE stabilizes and

Table 3: Identified Disjoint Safe Regions: We count the number of safe regions explored by the queries. The total numbers of queries are listed in Table 2. Transfer learning discovers multiple disjoint safe regions while baselines stick to neighborhood of the initial region.

<u>methods</u>	<u>GP1D</u>	<u>GP2D</u>	<u>Branin</u>
<u>EffTransHGP</u>	<u>1.79 ± 0.07</u>	<u>2.77 ± 0.13</u>	<u>2 ± 0</u>
<u>FullTransHGP</u>	<u>1.78 ± 0.07</u>	<u>3 ± 0.14213</u>	<u>2 ± 0</u>
<u>FullTransLMC</u>	<u>1.78 ± 0.08</u>	<u>2.68 ± 0.14</u>	<u>2 ± 0</u>
<u>Rothfuss2022</u>	<u>1.22 ± 0.05</u>	<u>1.07 ± 0.03</u>	<u>1 ± 0</u>
<u>SAL</u>	<u>1 ± 0</u>	<u>1.29 ± 0.09</u>	<u>1 ± 0</u>

the number of false ~~safe classifications~~ ~~positives~~ reduces. Initially, LMC ~~appears seems~~ to be overconfident regarding safety conditions, which we think ~~might be~~ ~~can~~ be attributed to overfitting caused by the larger number of hyperparameters due to the higher input ~~dimensions leading to a dimension~~.

In the main experiments, N_{source} (the number of source data points) is fixed for each dataset. In our Appendix F, we provide ablation studies on the Branin dataset, in which ~~we vary the number of source data points and number of source tasks.~~

Summary: Our approaches generally demonstrate improved convergence in terms of model performance and the extent of explored safe regions, while maintaining safety levels comparable to the baseline SAL. The benefits of our methods are most pronounced when multiple unconnected safe regions exist, as our methods are the only one capable of finding them. Among the three variants of our approach, we observe that LMC struggles when the input space is high-dimensional and data is scarce, potentially due to the larger number of hyperparameters. ~~Nevertheless, the queries of FullTransLMC are still more than 90% safe. In contrast, the HGP-based methods show consistently strong performance across all experiments.~~

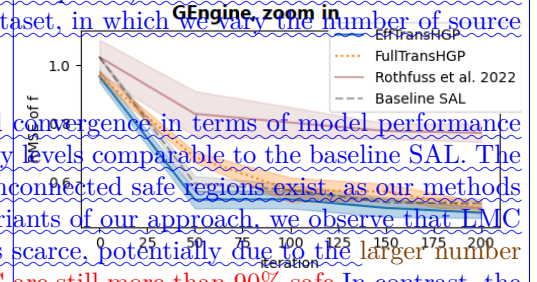


Figure 5: The RMSE zoom-in version of GEngine in Figure 4.

7.3 Disconnected Regions

Next, we examine in more detail whether the increased safe coverage observed in the previous section can be attributed to our transfer learning approaches effectively jumping between disconnected regions.

We analyse the number of disjoint regions for our synthetic problems with input dimension $D = 1$ or $D = 2$ (GP1D, ~~as shown in appendix Table 5.~~ GP2D, Branin). For these datasets, it is analytically and computationally possible to cluster the disconnected safe regions via connected component labeling (CCL) algorithms (He et al., 2017). Please see Appendix E.1 for further discussion of the CCL algorithm and its applicability. This allows us to track, in each experiment iteration, the specific safe region to which each observation belongs and count the number of disconnected regions (see Appendix Figure 11). At the end of the AL algorithm, we report the number of explored safe regions in Table 3. We say a region is explored if at least one query is in the region. This is valid because the safe set can expand from the at least one point. The results confirm the ability of our transfer learning approaches to explore disjoint safe regions, while the baseline methods cannot jump to disconnected regions. Notably, the Branin function is smooth and has two well-defined safe regions, while the GP data exhibit high stochasticity, leading to a range of small or large safe regions scattered throughout the space. While limited exploration is expected for the single-task approach SAL, it is surprising that the meta-learning approach Rothfuss2022 also fails to reach disconnected regions. This could be due to having only a single source task, which is uncommon for meta-learning as it typically involves multiple source tasks to differentiate between common and task-specific effects.

For the remaining datasets (Hartmann3, PEngine and GEngine), we cannot count the number of disconnected regions since the CCL algorithm cannot be applied. This is due to its limitations in dealing with noisy measurements (PEngine, GEngine) and dimensions greater than $D = 2$ (Hartmann, GEngine).

Our findings demonstrate that our transfer learning approaches effectively identify and explore multiple disjoint safe regions when they are present, a capability lacking in competing methods.

Table 4: Training Time of \mathbf{f} and \mathbf{q} (in seconds) at the last AL training: We observe that runtime increases sequentially from SAL to EffTransHGP, then to FullTransHGP, and finally to FullTransLMC. Rothfuss2022 performs only an initial training upfront which is not included in our runtime estimate, resulting in zero traing time.

datasets	EffTransHGP	FullTransHGP	FullTransLMC	Rothfuss2022	SAL
GP1D	8.947 ± 0.198	9.171 ± 0.133	26.56 ± 0.628	0.0 ± 0.0	6.881 ± 0.083
GP2D	10.73 ± 0.190	39.31 ± 0.639	202.8 ± 12.43	0.0 ± 0.0	8.044 ± 0.142
Branin	3.754 ± 0.121	8.129 ± 0.267	21.16 ± 1.207	0.0 ± 0.0	4.691 ± 0.078
Hartmann3	3.662 ± 0.089	9.092 ± 0.467	34.43 ± 1.664	0.0 ± 0.0	4.073 ± 0.083
PEngine	9.596 ± 0.418	124.99 ± 5.608	615.7 ± 27.99	0.0 ± 0.0	4.686 ± 0.243
GEngine	18.525 ± 2.508	503.11 ± 63.94	4357.8 ± 661.4	0.0 ± 0.0	10.485 ± 0.578

7.4 Runtime Analysis

Finally, we report training times in Table 4, measured as the time (in seconds) required to optimize the GP hyperparamters at the final iteration.

We observe that runtime increases sequentially from $SAL < \text{EffTransHGP} < \text{FullTransHGP} < \text{FullTransLMC}$, which aligns with our theoretical findings in Section 6. While both, SAL and EffTransHGP, scale cubically with the number of target points N , EffTransHGP takes longer due to the increased number of hyperparameters to optimize. FullTransHGP and FullTransLMC, in contrast, scale cubically with the combined number of source and target data $N_{\text{source}} + N$, with FullTransLMC requiring additional runtime due to an even larger number of hyperparameters.

The flexibility of our transfer approaches is inversely proportional to the training time. However, in our experiments, we do not observe a significant advantage of the FullTransLMC approach over HGP, likely due to the increased hyperparameter count in FullTransLMC, which can lead to overfitting issues. In summary, HGP proves to be the strongest approach, offering high efficiency without compromising on performance.

8 Conclusion

We propose a ~~transfer-safe-safe transfer~~ sequential learning to facilitate ~~real-real-world~~ experiments. We demonstrate its pronounced acceleration of learning~~which can be seen by a faster drop of RMSE and a larger,~~ evidenced by faster RMSE reduction and a greater safe set coverage. ~~At the same time~~Additionally, our modularized multi-output modeling 1) retains the potential ~~of performing for~~ global GP safe learning and 2) alleviates the cubic complexity ~~in the experiments, leading to a considerable reduce of time complexity from the source data, significantly reducing the runtime.~~

Limitations: Our modularized method is in theory compatible with any multi-output kernel, in contrast to the ensemble technique in Tighineanu et al. (2022) which is ~~only valid for limited to~~ a specific kernel ~~structure~~. However, one limitation of source precomputation is that it requires to fix correct source relevant hyperparameters solely with source data~~(e.g., HGP is a good candidate, For example, HGP is well-suited~~ due to its separable source-target structure while LMC, which learns joint patterns of tasks, ~~will not be fixed correctly with only source data).~~ Another limitation is that the benefit of transfer

~~learning relies on multi-task correlation. This means transfer learning will not be helpful when the correlation is absent, or when the source data are not present in our target safe area. Modeling with more complicated base kernel (we use Matérn-5/2 kernel) may enable more sophisticated multi-task correlations, but this may not correctly optimize with source data only.~~

~~While we only explored linear task correlations in this work, more sophisticated multi-output kernels, such as those in Álvarez et al. (2019), or the use of more complex base kernels, could support richer multitask correlations. However, investigating these approaches is beyond the scope of this paper (see e.g., e.g., Bitzer et al. (2022) for kernel selections). selection strategies).~~

~~When no correlation exists between the source and the target data, two outcomes are possible depending on the kernel design: (i) if the multi-output kernel includes the standard single-task kernel as a special case, performance may revert to that of baseline methods; (ii) if the standard kernel is not included as a special case, the signal may not be effectively modeled, resulting in suboptimal performance.~~

Future work: In this paper, we focus on problems of hundreds or up to thousands of data points (source and target data). ~~If we wish to scale further up~~ Scaling further to tens of thousands or millions of data points, ~~approximated models may require approximations,~~ such as sparse GP models (Titsias, 2009; Hensman et al., 2015) ~~may be required. These sparse GP models infer with a few inducing points, representing the original observation set. However, a suitable method of the inducing points selection remains opened, which use a limited set of inducing points to represent the original data. However, the optimal selection strategy for inducing points for sequential learning approaches is still an open research question (Moss et al., 2023; Pescador-Barrios et al., 2024).~~ ~~For example~~ For instance, the safety model ~~needs inducing points approximating a good~~ requires inducing points that effectively represent the safe set, while the ~~acquisition function needs to consider how the inducing points change~~ inducing points of the acquisition model need to be updated after each query (or ~~each~~ batch of queries) ~~to appropriately reflect changes in uncertainty.~~

Acknowledgements

References

- Mauricio A. Álvarez, Lorenzo Rosasco, and Neil D. Lawrence. Kernels for vector-valued functions: a review. *arXiv*, 2012.
- Mauricio A Álvarez, Wil Ward, and Cristian Guarnizo. Non-linear process convolutions for multi-output gaussian processes. *International Conference on Artificial Intelligence and Statistics*, 2019.
- Dominik Baumann, Alonso Marco, Matteo Turchetta, and Sebastian Trimpe. GoSafe: Globally Optimal Safe Robot Learning. *IEEE International Conference on Robotics and Automation*, 2021.
- Felix Berkenkamp, Angela P. Schoellig, and Andreas Krause. Safe controller optimization for quadrotors with gaussian processes. *International Conference on Robotics and Automation*, 2016.

- Felix Berkenkamp, Andreas Krause, and Angela P. Schoellig. Bayesian optimization with safety constraints: Safe and automatic parameter tuning in robotics. *Machine Learning*, 2020.
- Matthias Bitzer, Mona Meister, and Christoph Zimmer. Structural kernel search via bayesian optimization and symbolical optimal transport. *Advances in Neural Information Processing Systems*, 2022.
- Eric Brochu, Vlad M. Cora, and Nando de Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv*, 2010.
- Kathryn Chaloner and Isabella Verdinelli. Bayesian experimental design: A review. *Statistical Science*, 1995.
- Javier García, Fern, and o Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 2015.
- Michael A. Gelbart, Jasper Snoek, and Ryan P. Adams. Bayesian optimization with unknown constraints. *Conference on Uncertainty in Artificial Intelligence*, 2014.
- Susan Harkema, Yury Gerasimenko, Jonathan Hodes, Joel Burdick, Claudia Angeli, Yangsheng Chen, Christie Ferreira, Andrea Willhite, Enrico Rejc, Robert G Grossman, and V Reggie Edgerton. Effect of epidural stimulation of the lumbosacral spinal cord on voluntary movement, standing, and assisted stepping after motor complete paraplegia: a case study. *The Lancet*, 2011.
- Lifeng He, Xiwei Ren, Qihang Gao, Xiao Zhao, Bin Yao, and Yuyan Chao. The connected-component labeling problem: A review of state-of-the-art algorithms. *Pattern Recognition*, 2017.
- James Hensman, Alexander Matthews, and Zoubin Ghahramani. Scalable Variational Gaussian Process Classification. *International Conference on Artificial Intelligence and Statistics*, 2015.
- Jose Miguel Hernandez-Lobato, Michael Gelbart, Matthew Hoffman, Ryan Adams, and Zoubin Ghahramani. Predictive entropy search for bayesian optimization with unknown constraints. *International Conference on Machine Learning*, 2015.
- José Miguel Hernández-Lobato, Michael A. Gelbart, Ryan P. Adams, Matthew W. Hoffman, and Zoubin Ghahramani. A general framework for constrained bayesian optimization using information-based search. *Journal of Machine Learning Research*, 2016.
- A. G. Journel and C. J. Huijbregts. Mining geostatistics. *Academic Press London*, 1976.
- M. Kanagawa, P. Hennig, D. Sejdinovic, and B. K. Sriperumbudur. Gaussian processes and kernel methods: A review on connections and equivalences. *arXiv*, 2018.

- Andreas Krause, Ajit Singh, and Carlos Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 2008.
- Punit Kumar and Atul Gupta. Active learning query strategies for classification, regression, and clustering: A survey. *Journal of Computer Science and Technology*, 2020.
- Armin Lederer, Jonas Umlauf, and Sandra Hirche. Posterior variance analysis of gaussian processes with application to average learning curves. *arXiv*, 2019.
- Cen-You Li, Barbara Rakitsch, and Christoph Zimmer. Safe active learning for multi-output gaussian processes. *International Conference on Artificial Intelligence and Statistics*, 2022.
- Shibo Li, Wei Xing, Robert Kirby, and Shandian Zhe. Multi-fidelity bayesian optimization via deep neural networks. *Advances in Neural Information Processing Systems*, 2020.
- D. V. Lindley. On a Measure of the Information Provided by an Experiment. *The Annals of Mathematical Statistics*, 1956.
- Alonso Marco, Felix Berkenkamp, Philipp Hennig, Angela P. Schoellig, Andreas Krause, Stefan Schaal, and Sebastian Trimpe. Virtual vs. real: Trading off simulations and physical experiments in reinforcement learning with bayesian optimization. *IEEE International Conference on Robotics and Automation*, 2017.
- Henry B. Moss, Sebastian W. Ober, and Victor Picheny. Inducing point allocation for sparse gaussian processes in high-throughput bayesian optimisation. *International Conference on Artificial Intelligence and Statistics*, 2023.
- Guimar Pescador-Barrios, Sarah Filippi, and Mark van der Wilk. "how big is big enough?" adjusting model size in continual gaussian processes. *arXiv*, 2024.
- Matthias Poloczek, Jiale Wang, and Peter Frazier. Multi-information source optimization. *Advances in Neural Information Processing Systems*, 2017.
- William H Press, William T Vetterling, Saul A Teukolsky, and Brian P Flannery. Numerical recipes. *Cambridge University Press*, 1988.
- CE. Rasmussen and CKI. Williams. Gaussian processes for machine learning. *MIT Press*, 2006.
- Christoffer Riis, Francisco Antunes, Frederik Boe Hüttl, Carlos Lima Azevedo, and Francisco Câmara Pereira. Bayesian active learning with fully bayesian gaussian processes. *Advances in Neural Information Processing Systems*, 2022.
- Jonas Rothfuss, Christopher Koenig, Alisa Rupenyan, and Andreas Krause. Meta-Learning Priors for Safe Bayesian Optimization. *Conference on Robot Learning*, 2022.

- Bernhard Schoelkopf and Alexander J. Smola. Learning with kernels: Support vector machines, regularization, optimization, and beyond. *MIT Press*, 2002.
- Jens Schreiter, Duy Nguyen-Tuong, Mona Eberts, Bastian Bischoff, Heiner Markert, and Marc Toussaint. Safe exploration for active learning with gaussian processes. *Machine Learning and Knowledge Discovery in Databases*, 2015.
- Yaroslav D. Sergeyev, Antonio Candelieri, Dmitri E. Kvasov, and Riccardo Perego. Safe global optimization of expensive noisy black-box functions in the δ -Lipschitz framework. *Soft Computing*, 2020.
- B. W. Silverman. Spline smoothing: The equivalent variable kernel method. *Annals of Statistics*, 1984.
- Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. *Advances in Neural Information Processing Systems*, 2012.
- Niranjan Srinivas, Andreas Krause, Sham M. Kakade, and Matthias W. Seeger. Information-theoretic regret bounds for gaussian process optimization in the bandit setting. *IEEE Transactions on Information Theory*, 2012.
- Yanan Sui, Alkis Gotovos, Joel Burdick, and Andreas Krause. Safe exploration for optimization with gaussian processes. *International Conference on Machine Learning*, 2015.
- Yanan Sui, Vincent Zhuang, Joel W. Burdick, and Yisong Yue. Stagewise Safe Bayesian Optimization with Gaussian Processes. *International Conference on Machine Learning*, 80, 2018.
- Kevin Swersky, Jasper Snoek, and Ryan P Adams. Multi-task bayesian optimization. *Advances in Neural Information Processing Systems*, 2013.
- Petru Tighineanu, Kathrin Skubch, Paul Baireuther, Attila Reiss, Felix Berkenkamp, and Julia Vinogradskaa. Transfer learning with gaussian processes for bayesian optimization. *International Conference on Artificial Intelligence and Statistics*, 2022.
- Michalis Titsias. Variational learning of inducing variables in sparse gaussian processes. *International Conference on Artificial Intelligence and Statistics*, 2009.
- Matteo Turchetta, Felix Berkenkamp, and Andreas Krause. Safe Exploration for Interactive Machine Learning. *Advances in Neural Information Processing Systems*, 2019.
- Yehong Zhang, Trong Nghia Hoang, Kian Hsiang Low, and Mohan Kankanhalli. Near-optimal active learning of multi-output gaussian processes. *AAAI Conference on Artificial Intelligence*, 2016.
- Christoph Zimmer, Mona Meister, and Duy Nguyen-Tuong. Safe active learning for time-series modeling with gaussian processes. *Advances in Neural Information Processing Systems*, 2018.

A Appendix Overview

~~?? provides detailed analysis and illustrations.~~ Appendix B lists commonly used kernels and the r - δ relation needed for our theoretical analysis. Appendix C provides the proof of our main theorem. In ~~??~~ Appendix D, we demonstrate the math of our source pre-computation technique ~~.-?? as well as general transfer task GPs with more than one source tasks.~~ Appendix E contains the experiment details and ~~??~~ Appendix F the ablation studies, additional plots and tables.

B GPs with classical stationary kernels cannot jump through an unsafe valley

Common Kernels and r - δ Relation

B.1 Bound of explorable region of safe learning methods

~~In our main script, we provide a bound of the safety probability. The theorem is restated here.~~

~~We are given $\forall \mathbf{x}_* \in \mathcal{X}$ Our main theorem use Definition 5.1, which is restated here, $\mathbf{x}_{1:N} \subseteq \mathcal{X}$, a kernel k_{q^j} satisfying ?? and $k_{q^j}(\cdot, \cdot) \leq 1$. Denote $k_{scale}^j := \max k_{q^j}(\cdot, \cdot)$. $q^j \sim \mathcal{GP}(0, k_{q^j})$ is a GP, $z_{1:N}^j := (z_1^j, \dots, z_N^j)$ is a set of observed noisy values and $\|(z_1^j, \dots, z_N^j)\| \leq \sqrt{N}$. Then $\forall \delta \in (0, \sqrt{k_{scale}^j \sigma_{q^j} / \sqrt{N}})$, $\exists r > 0$ s.t. when $\min_{\mathbf{x}_i \in \mathbf{x}_{1:N}} \|\mathbf{x}_* - \mathbf{x}_i\| \geq r$, the probability thresholded on a constant T_j is bounded by~~

$$p\left((q^j(\mathbf{x}_*) \geq T_j) | \mathbf{x}_{1:N}, z_{1:N}^j\right) \leq \Phi\left(\frac{N\delta/\sigma_{q^j}^2 - T_j}{\sqrt{k_{scale}^j - (\sqrt{N}\delta/\sigma_{q^j})^2}}\right).$$

~~In this section, we illustrate a concrete example of our theorem, where conventional methods cannot explore the entire safe set in the space. Then we provide the proof of this theorem.~~

B.1 Single-output GP does not reach disconnected safe region

~~We plug some exact numbers into the probability bound. Consider an one dimensional situation as Figure 2 and ??.~~ We omit j because $J = 1$ here. Assume

1. $N = 10$,
2. $\sigma_q^2 = 0.01$,
3. $T = 0$ (notice $z_{1:N}^j$ is normalized to 0-mean and unit-variance).

~~In this example, the generated data have $\|z_{1:N}\| \leq \sqrt{N}$ (see Figure 2 for the rough functional values). Noticed also that σ_q/\sqrt{N} is around 0.0316. We fix $k_{scale} := \max k_q(\cdot, \cdot) = 1$ (the surrogate model in Figure 2). Then our theoretical bound of the safety probability is $\Phi\left(\frac{N\delta/\sigma^2 - T}{\sqrt{1 - (\sqrt{N}\delta/\sigma)^2}}\right) = \Phi\left(\frac{1000\delta}{\sqrt{1 - 1000\delta^2}}\right)$.~~

~~In our main script, \mathbf{x}_* is unsafe if $p\left((q^j(\mathbf{x}_*) \geq T_j) | \mathbf{x}_{1:N}, z_{1:N}^j\right) < 1 - \Phi(-\beta^{1/2}) = \Phi(\beta^{1/2})$. We set the safety tolerance to $\beta^{1/2} = 2$. The decision boundary of our theorem $\frac{1000\delta}{\sqrt{1 - 1000\delta^2}} = 2$ means $\delta \approx 0.002$.~~

~~From ?? we see that $\|\mathbf{x} - \mathbf{x}'\| \geq 4.485 \Rightarrow \delta \leq 0.002$ for unit lengthscale Matérn-5/2 kernel. With a lengthscale parameter l , this becomes $\frac{\|\mathbf{x} - \mathbf{x}'\|}{l} \geq 4.485 \Leftrightarrow \|\mathbf{x} - \mathbf{x}'\| \geq 4.485 * l$. Therefore $\delta \leq 0.002$ if $\|\mathbf{x} - \mathbf{x}'\| \geq 4.485 * l$.~~

~~The GP-model trained on this example has lengthscale ≈ 0.1256 (the surrogate model in Figure 2 and in left of ??), so points that are at least $4.485 * 0.1256 = 0.563316$ away from the observations are always identified unsafe. Thus the safe region on the right is never inferred as safe and is not explored with conventional single-output GP model (??, left), because the distance between the two disjoint safe regions is around 0.7.~~

We also show empirically that a multi-output GP model transfer safety confidence from a source task and identify safe region \mathcal{S}_{sub2} (??, right).

0.2in The safety function $q(x) = \sin(10x^3 - 5x - 10) + \frac{1}{3}x^2 - \frac{1}{2}$. Safety threshold is set to $T = 0$. The observations are with noise drawn from $\mathcal{N}(0, 0.01)$. Left: a GP with Matérn-5/2 kernel (lengthscale ≈ 0.1256) is shown. The red lines indicate the largest observed \mathbf{x} and the closest safe point of another region. The gap between the red lines is close to 0.7, which is beyond explorable region of conventional safe learning methods. Right: the multi-output model uses an LMC kernel with 2 latent Matérn-5/2 kernels (Álvarez et al., 2012). Additional noisy data from function $q_s(x) = \sin(10x^3 - 5x - 10) + \sin(x^2) - \frac{1}{2}$ are provided (yellow). \mathcal{S}_{sub1} and \mathcal{S}_{sub2} are the safe set inferred by the LMC. -0.2in

B.1 r - δ relation for commonly used kernels

Our main theorem consider kernels satisfying Assumption 3.2 which is restated here: Given a kernel function $k: \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$, assume $\forall \delta > 0, \exists r > 0$ s.t. $\|\mathbf{x} - \mathbf{x}'\| \geq r \Rightarrow k(\mathbf{x}, \mathbf{x}') \leq \delta$ under $L2$ norm. to measure the covariance with respect to the distance of data:

Definition 5.1. We call a kernel k a kernel with correlation weakened by distance if $k: \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$, assume fulfills the following property: $\forall \delta > 0, \exists r > 0$ s.t. $\|\mathbf{x} - \mathbf{x}'\| \geq r \Rightarrow k(\mathbf{x}, \mathbf{x}') \leq \delta$ under $L2$ norm. We provide expression

Notice that this assumption-property is weaker than k being strictly decreasing (see e.g. Lederer et al. (2019)), and. In addition, it does not explicitly force stationarity, while not all stationary kernels have this property, e.g. cosine kernel $k(\mathbf{x}, \mathbf{x}') = \cos(\|\mathbf{x} - \mathbf{x}'\|_2)$ does not follow this definition.

Here we want to find the exact r for commonly used kernels, given a δ . The following kernels (denoted by $k(\cdot, \cdot)$) are described in their standard forms. In the experiments, we often add a lengthscale l and variance $k_{scale}l > 0$ and variance $k_{scale} > 0$, i.e. $k_{parameterized}(\mathbf{x}, \mathbf{x}') = k_{scale}k(\mathbf{x}/l, \mathbf{x}'/l)$ where k_{scale} and l are trainable parameters. The lengthscale l can also be a vector, where each component is a scaling factor of the corresponding dimension of the data.

RBF kernel

$$k(\mathbf{x}, \mathbf{x}') = \exp(-\|\mathbf{x} - \mathbf{x}'\|^2/2):$$

$$k(\mathbf{x}, \mathbf{x}') \leq \delta \Leftrightarrow \|\mathbf{x} - \mathbf{x}'\| \geq \sqrt{\log \frac{1}{\delta^2}}.$$

$$\text{E.g. } \delta \leq 0.3 \Leftrightarrow \|\mathbf{x} - \mathbf{x}'\| \geq 1.552$$

$$\delta \leq 0.1 \Leftrightarrow \|\mathbf{x} - \mathbf{x}'\| \geq 2.146$$

$$\delta \leq 0.002 \Leftrightarrow \|\mathbf{x} - \mathbf{x}'\| \geq 3.526$$

Matérn-1/2 kernel

$$k(\mathbf{x}, \mathbf{x}') = \exp(-\|\mathbf{x} - \mathbf{x}'\|): k(\mathbf{x}, \mathbf{x}') \leq \delta \Leftrightarrow \|\mathbf{x} - \mathbf{x}'\| \geq \log \frac{1}{\delta}.$$

$$\text{E.g. } \delta \leq 0.3 \Leftrightarrow \|\mathbf{x} - \mathbf{x}'\| \geq 1.204$$

$$\delta \leq 0.1 \Leftrightarrow \|\mathbf{x} - \mathbf{x}'\| \geq 2.303$$

$$\delta \leq 0.002 \Leftrightarrow \|\mathbf{x} - \mathbf{x}'\| \geq 6.217$$

Matérn-3/2 kernel

$$k(\mathbf{x}, \mathbf{x}') = (1 + \sqrt{3}\|\mathbf{x} - \mathbf{x}'\|) \exp(-\sqrt{3}\|\mathbf{x} - \mathbf{x}'\|):$$

$$\text{E.g. } \delta \leq 0.3 \Leftrightarrow \|\mathbf{x} - \mathbf{x}'\| \geq 1.409$$

$$\delta \leq 0.1 \Leftrightarrow \|\mathbf{x} - \mathbf{x}'\| \geq 2.246$$

$$\delta \leq 0.002 \Leftrightarrow \|\mathbf{x} - \mathbf{x}'\| \geq 4.886$$

Matérn-5/2 kernel

$$k(\mathbf{x}, \mathbf{x}') = (1 + \sqrt{5}\|\mathbf{x} - \mathbf{x}'\| + \frac{5}{3}\|\mathbf{x} - \mathbf{x}'\|^2) \exp(-\sqrt{5}\|\mathbf{x} - \mathbf{x}'\|):$$

$$\text{E.g. } \delta \leq 0.3 \Leftrightarrow \|\mathbf{x} - \mathbf{x}'\| \geq 1.457$$

$$\delta \leq 0.1 \Leftrightarrow \|\mathbf{x} - \mathbf{x}'\| \geq 2.214$$

$$\delta \leq 0.002 \Leftrightarrow \|\mathbf{x} - \mathbf{x}'\| \geq 4.485$$

B.1 Proof of our main theorem**C GP Local Exploration - Proof**

In our main script, we provide a bound of the safety probability. In this section, we provide the proof of this theorem.

We first introduce some necessary theoretical properties in [?? Appendix C.1](#), and then use the properties to prove [Theorem 3.3 in ?? Theorem 5.2 and Corollary 5.3 in Appendix C.2](#).

C.0.1 Additional lemmas**C.1 Additional Lemmas**

Definition C.1. Let $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ be a kernel, $\mathbf{A} \subseteq \mathcal{X}$ be any dataset of finite number of elements, and let σ be any positive real number, denote $\Omega_{k, \mathbf{A}, \sigma^2} := k(\mathbf{A}, \mathbf{A}) + \sigma^2 I$.

Definition C.2. Given a kernel $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$, dataset $\mathbf{A} \subseteq \mathcal{X}$, and some positive real number σ , then for $\mathbf{x} \in \mathcal{X}$, the k -, \mathbf{A} -, and σ^2 -dependent function $\mathbf{h}(\mathbf{x}) = k(\mathbf{A}, \mathbf{x})^T \Omega_{k, \mathbf{A}, \sigma^2}^{-1}$ is called a weight function ([Silverman, 1984](#)).

Proposition C.3. $C \in \mathbb{R}^{M \times M}$ is a positive definite matrix and $\mathbf{b} \in \mathbb{R}^M$ is a vector. λ_{max} is the maximum eigenvalue of C . We have $\|C\mathbf{b}\|_2 \leq \lambda_{max} \|\mathbf{b}\|_2$.

Proof of Proposition C.3.

Because C is positive definite (symmetric), we can find orthonormal eigenvectors $\{\mathbf{e}_1, \dots, \mathbf{e}_M\}$ of C that form a basis of \mathbb{R}^M . Let λ_i be the eigenvalue corresponding to \mathbf{e}_i , we have $\lambda_i > 0$.

As $\{\mathbf{e}_1, \dots, \mathbf{e}_M\}$ is a basis, there exist $b_1, \dots, b_M \in \mathbb{R}$ s.t. $\mathbf{b} = \sum_{i=1}^M b_i \mathbf{e}_i$. Since $\{\mathbf{e}_i\}$ is orthonormal, $\|\mathbf{b}\|_2^2 = \sum_i b_i^2$. Then

$$\begin{aligned} \|C\mathbf{b}\|_2 &= \left\| \sum_{i=1}^M b_i \lambda_i \mathbf{e}_i \right\|_2 = \sqrt{\sum_{i=1}^M b_i^2 \lambda_i^2} \\ &\leq \sqrt{\sum_{i=1}^M b_i^2 \lambda_{\max}^2} = \lambda_{\max} \sqrt{\sum_{i=1}^M b_i^2} = \lambda_{\max} \|\mathbf{b}\|_2 \end{aligned}$$

□

Proposition C.4. $\forall \mathbf{A} \subseteq \mathcal{X}$, any kernel k , and any positive real number σ , an eigenvalue λ of $\Omega_{k, \mathbf{A}, \sigma^2}$ (Definition C.1) must satisfy $\lambda \geq \sigma^2$.

Proof of Proposition C.4.

Let $\mathbf{K} := k(\mathbf{A}, \mathbf{A})$. We know that

1. \mathbf{K} is positive semidefinite, so it has only non-negative eigenvalues, denote the minimal one by λ_K , and
2. σ^2 is the only eigenvalue of $\sigma^2 I$.

Then Weyl's inequality immediately gives us the result: $\lambda \geq \lambda_K + \sigma^2 \geq \sigma^2$. □

Corollary C.5. We are given $\forall \mathbf{x}_* \in \mathcal{X}$, $\mathbf{A} \subseteq \mathcal{X}$, any kernel k ~~satisfying Assumption 3.2~~ with correlation weakened by distance (Definition 5.1), and any positive real number σ . Let $M :=$ number of elements of \mathbf{A} , and let $\mathbf{B} \in \mathbb{R}^M$ be a vector. Then $\forall \delta > 0, \exists r > 0$ s.t. when $\min_{\mathbf{x}' \in \mathbf{A}} \|\mathbf{x}_* - \mathbf{x}'\| \geq r$, we have

1. $|\mathbf{h}(\mathbf{x}_*)\mathbf{B}| \leq \sqrt{M}\delta\|\mathbf{B}\|/\sigma^2$ (see also Definition C.2),
2. $k(\mathbf{x}_*, \mathbf{x}_*) - k(\mathbf{A}, \mathbf{x}_*)^T \Omega_{k, \mathbf{A}, \sigma^2}^{-1} k(\mathbf{A}, \mathbf{x}_*) \geq k(\mathbf{x}_*, \mathbf{x}_*) - M\delta^2/\sigma^2$ (see also Definition C.1).

Proof of Corollary C.5.

Let $\mathbf{K} := k(\mathbf{A}, \mathbf{A})$.

Proposition C.4 implies that the eigenvalues of $(\mathbf{K} + \sigma^2 I)^{-1}$ are bounded by $\frac{1}{\sigma^2}$.

In addition, Definition 5.1 gives us $\min_{\mathbf{x}' \in \mathbf{A}} \|\mathbf{x}_* - \mathbf{x}'\| \geq r \Rightarrow$ all components of row vector $k(\mathbf{x}_*, \mathbf{A})$ are in region $[0, \delta]$.

1. Apply Cauchy-Schwarz inequality (line 1) and Proposition C.3 (line 2), we obtain

$$\begin{aligned} |k(\mathbf{A}, \mathbf{x}_*)^T (\mathbf{K} + \sigma^2 I)^{-1} \mathbf{B}| &\leq \|k(\mathbf{A}, \mathbf{x}_*)^T\| \|(\mathbf{K} + \sigma^2 I)^{-1} \mathbf{B}\| \\ &\leq \|k(\mathbf{A}, \mathbf{x}_*)\| \frac{1}{\sigma^2} \|\mathbf{B}\| \\ &\leq \|(\delta, \dots, \delta)\| \frac{1}{\sigma^2} \|\mathbf{B}\| \\ &\leq \frac{\sqrt{M}\delta\|\mathbf{B}\|}{\sigma^2}. \end{aligned}$$

2. $(\mathbf{K} + \sigma^2 I)^{-1}$ is positive definite Hermitian matrix, so

$$\begin{aligned} k(\mathbf{A}, \mathbf{x}_*)^T (\mathbf{K} + \sigma^2 I)^{-1} k(\mathbf{A}, \mathbf{x}_*) &\leq \frac{1}{\sigma^2} \|k(\mathbf{A}, \mathbf{x}_*)\|^2 \\ &\leq \frac{1}{\sigma^2} M \delta^2. \end{aligned}$$

Then, we immediately see that

$$\begin{aligned} k(\mathbf{x}_*, \mathbf{x}_*) - k(\mathbf{A}, \mathbf{x}_*)^T (\mathbf{K} + \sigma^2 I)^{-1} k(\mathbf{A}, \mathbf{x}_*) &\geq k(\mathbf{x}_*, \mathbf{x}_*) - \frac{1}{\sigma^2} \|k(\mathbf{A}, \mathbf{x}_*)\|^2 \\ &\geq k(\mathbf{x}_*, \mathbf{x}_*) - \frac{1}{\sigma^2} M \delta^2. \end{aligned}$$

□

Remark C.6. ~~A CDF Φ is the cumulative density function (CDF) of a standard Gaussian distribution is often denoted by $p(x \leq T) = \Phi(T)$, $x \sim \mathcal{N}(0, 1)$. Notice that $\mathcal{N}(0, 1)$, $p(x \leq T) = \Phi(T)$, $p(x \leq -T) = \Phi(-T) = 1 - \Phi(T) = p(x \geq T)$.~~

C.1.1 Main proof

~~We are given $\forall \mathbf{x}_* \in \mathcal{X}$, $\mathbf{x}_{1:N} \subseteq \mathcal{X}$, a kernel k_{q^j} satisfying ?? and $k_{q^j}(\cdot, \cdot) \leq 1$. Denote $k_{scale}^j := \max k_{q^j}(\cdot, \cdot)$. $q^j \sim \mathcal{GP}(0, k_{q^j})$ is a GP, $z_{1:N}^j := (z_1^j, \dots, z_N^j)$ is a set of observed noisy values and $\|(z_1^j, \dots, z_N^j)\| \leq \sqrt{N}$. Then $\forall \delta \in (0, \sqrt{k_{scale}^j \sigma_{q^j}} / \sqrt{N})$, $\exists r > 0$ s.t. when $\min_{\mathbf{x}_i \in \mathbf{x}_{1:N}} \|\mathbf{x}_* - \mathbf{x}_i\| \geq r$, the probability thresholded on a constant T_j is bounded by $p\left((q^j(\mathbf{x}_*) \geq T_j) | \mathbf{x}_{1:N}, z_{1:N}^j\right) \leq \Phi\left(\frac{N\delta/\sigma_{q^j}^2 - T_j}{\sqrt{k_{scale}^j - (\sqrt{N}\delta/\sigma_{q^j})^2}}\right)$.~~

C.2 Main Proof

The theorem is restated again.

Theorem 5.2 (Local exploration of single-output GPs). ~~We are given $\forall \mathbf{x}_* \in \mathcal{X}$, $\mathbf{x}_{1:N} \subseteq \mathcal{X}$, a kernel k_{q^j} satisfying ?? (distant points result in weak correlation) and $k_{q^j}(\cdot, \cdot) \leq 1$. Denote $k_{scale}^j := \max k_{q^j}(\cdot, \cdot)$. $q^j \sim \mathcal{GP}(0, k_{q^j})$ is a GP, For any safety constraint indexed by $j = 1, \dots, J$, let $z_{1:N}^j := (z_1^j, \dots, z_N^j)$ is a set of observed noisy values and be the observed noisy safety values and let $\|(z_1^j, \dots, z_N^j)\| \leq \sqrt{N}$. The safety value $z^j = q^j(\mathbf{x}) + \epsilon_{q^j}$ satisfies the GP assumptions (Assumption 3.1, Assumption 3.2); $q^j \sim \mathcal{GP}(0, k_{q^j})$, $k_{q^j}(\cdot, \cdot) \leq 1$, $\epsilon_{q^j} \sim \mathcal{N}(0, \sigma_{q^j}^2)$. The kernel k_{q^j} is a kernel with correlation weakened by distance (Definition 5.1). Denote $k_{scale}^j := \max k_{q^j}(\cdot, \cdot)$. Then $\forall \delta \in (0, \sqrt{k_{scale}^j \sigma_{q^j}} / \sqrt{N})$, $\exists r > 0$ s.t. when $\forall \mathbf{x}_* \in \mathcal{X}$ that fulfill $\min_{\mathbf{x}_i \in \mathbf{x}_{1:N}} \|\mathbf{x}_* - \mathbf{x}_i\| \geq r$, the probability thresholded on a constant T_j is bounded by $p\left((q^j(\mathbf{x}_*) \geq T_j) | \mathbf{x}_{1:N}, z_{1:N}^j\right) \leq \Phi\left(\frac{N\delta/\sigma_{q^j}^2 - T_j}{\sqrt{k_{scale}^j - (\sqrt{N}\delta/\sigma_{q^j})^2}}\right)$. Φ is the CDF of standard Gaussian.~~

Proof.

From Equation (2) in the main script, we know that

$$\begin{aligned} p\left(q^j(\mathbf{x}_*)|\mathbf{x}_{1:N}, z_{1:N}^j\right) &= \mathcal{N}\left(\mathbf{x}_*|\mu_{q^j,N}(\mathbf{x}_*), \sigma_{q^j,N}^2(\mathbf{x}_*)\right) \\ \mu_{q^j,N}(\mathbf{x}_*) &= k_{q^j}(\mathbf{x}_{1:N}, \mathbf{x}_*)^T \left(k_{q^j}(\mathbf{x}_{1:N}, \mathbf{x}_{1:N}) + \sigma_{q^j}^2 I_N\right)^{-1} z_{1:N}^j \\ \sigma_{q^j,N}^2(\mathbf{x}_*) &= k_{q^j}(\mathbf{x}_*, \mathbf{x}_*) - k_{q^j}(\mathbf{x}_{1:N}, \mathbf{x}_*)^T \left(k_{q^j}(\mathbf{x}_{1:N}, \mathbf{x}_{1:N}) + \sigma_{q^j}^2 I_N\right)^{-1} k_{q^j}(\mathbf{x}_{1:N}, \mathbf{x}_*). \end{aligned}$$

We also know that (Remark C.6)

$$\begin{aligned} p\left((q^j(\mathbf{x}_*) \geq T_j)|\mathbf{x}_{1:N}, z_{1:N}^j\right) &= 1 - \Phi\left(\frac{T_j - \mu_{q^j,N}(\mathbf{x}_*)}{\sigma_{q^j,N}(\mathbf{x}_*)}\right) \\ &= \Phi\left(\frac{\mu_{q^j,N}(\mathbf{x}_*) - T_j}{\sigma_{q^j,N}(\mathbf{x}_*)}\right). \end{aligned}$$

From Corollary C.5, we get $\frac{\mu_{q^j,N}(\mathbf{x}_*) - T_j}{\sigma_{q^j,N}(\mathbf{x}_*)} \leq \frac{\sqrt{N}\delta\|z_{1:N}^j\|/\sigma_{q^j}^2 - T_j}{\sqrt{k_{q^j}(\mathbf{x}_*, \mathbf{x}_*) - N\delta^2/\sigma_{q^j}^2}}$. This is valid because we assume $\delta < \sqrt{k_{scale}^j \sigma_{q^j}^2}/\sqrt{N}$. Then with $\|z_{1:N}^j\| \leq \sqrt{N}$ and the fact that Φ is an increasing function, we immediately see the result

$$p\left((q^j(\mathbf{x}_*) \geq T_j)|\mathbf{x}_{1:N}, z_{1:N}^j\right) \leq \Phi\left(\frac{N\delta/\sigma_{q^j}^2 - T_j}{\sqrt{k_{scale}^j - (N\delta/\sigma_{q^j})^2}}\right).$$

□

Then, we would like to prove the Corollary 5.3 which is restated here.

Corollary 5.3 (Existence of δ). We are given the assumptions in Theorem 5.2. For each $j = 1, \dots, J$, if either (1) $T_j \geq 0, \beta^{1/2} > 0$ or (2) $T_j < 0, \beta^{1/2} > \frac{|T_j|}{\sqrt{k_{scale}^j \sigma_{q^j}^2}/\sqrt{N}}$, then $\exists \delta \in (0, \sqrt{k_{scale}^j \sigma_{q^j}^2}/\sqrt{N})$ s.t.

$$\Phi\left(\frac{N\delta/\sigma_{q^j}^2 - T_j}{\sqrt{k_{scale}^j - (N\delta/\sigma_{q^j})^2}}\right) \leq \Phi(\beta^{1/2}).$$

Proof. This can be proved by substituting the constants.

Condition (1) $T_j \geq 0, \beta^{1/2} > 0$:

$$\frac{N\delta/\sigma_{q^j}^2 - T_j}{\sqrt{k_{scale}^j - (N\delta/\sigma_{q^j})^2}} \leq \frac{N\delta/\sigma_{q^j}^2}{\sqrt{k_{scale}^j - (N\delta/\sigma_{q^j})^2}},$$

$\lim_{\delta \rightarrow 0^+} \frac{N\delta/\sigma_{q^j}^2}{\sqrt{k_{scale}^j - (N\delta/\sigma_{q^j})^2}} = 0$ guarantees $\exists \delta \in (0, \sqrt{k_{scale}^j \sigma_{q^j}^2}/\sqrt{N})$ s.t. $\frac{N\delta/\sigma_{q^j}^2}{\sqrt{k_{scale}^j - (N\delta/\sigma_{q^j})^2}} \leq \beta^{1/2}$ for $\beta^{1/2} > 0$. Then because Φ is strictly increasing, the same δ gives $\Phi\left(\frac{N\delta/\sigma_{q^j}^2 - T_j}{\sqrt{k_{scale}^j - (N\delta/\sigma_{q^j})^2}}\right) \leq \Phi(\beta^{1/2})$.

Condition (2) $T_j < 0, \beta^{1/2} > \frac{|T_j|}{\sqrt{k_{scale}^j \sigma_{q^j}^2}/\sqrt{N}}$: We see here that $\lim_{\delta \rightarrow 0^+} \frac{N\delta/\sigma_{q^j}^2 - T_j}{\sqrt{k_{scale}^j - (N\delta/\sigma_{q^j})^2}} = \frac{-T_j}{\sqrt{k_{scale}^j}} < \beta^{1/2}$. Therefore, there must exist $\delta \in (0, \sqrt{k_{scale}^j \sigma_{q^j}^2}/\sqrt{N})$ s.t. $\Phi\left(\frac{N\delta/\sigma_{q^j}^2 - T_j}{\sqrt{k_{scale}^j - (N\delta/\sigma_{q^j})^2}}\right) \leq \Phi(\beta^{1/2})$. □

D Multi-output GPs with ~~source pre-computation~~ Source Pre-Computation

D.1 Two-steps Cholesky Decomposition

Given a multi-output GP $\mathbf{g} \sim \mathcal{GP}(0, k_{\mathbf{g}})$, $\mathbf{g} \in \{\mathbf{f}, \mathbf{q}^1, \dots, \mathbf{q}^J\}$, where $k_{\mathbf{g}}$ is an arbitrary kernel, the main computational challenge is to compute the inverse or Cholesky decomposition of

$$\Omega_{\mathbf{g}} = \begin{pmatrix} K_{g_s} + \sigma_{g_s}^2 I_{N_{\text{source}}} & K_{g_s, g} \\ K_{g_s, g}^T & K_g + \sigma_g^2 I_N \end{pmatrix}.$$

Such computation has time complexity ~~$\mathcal{O}((N_{\text{source}} + N)^3)$~~ $\mathcal{O}((N_{\text{source}} + N)^3)$. We wish to avoid this computation repeatedly. As in our main script, $k_{\mathbf{g}}$ is parameterized and we write the parameters as $\boldsymbol{\theta}_{\mathbf{g}} = (\theta_{g_s}, \theta_g)$, where $k_{\mathbf{g}}((\cdot, s), (\cdot, s))$ is independent of θ_g . ~~$k_{\mathbf{g}}((\cdot, s), (\cdot, t))$ and $k_{\mathbf{g}}((\cdot, t), (\cdot, t))$ does not need to be independent of θ_{g_s}~~

Here we propose to fix K_{g_s} (~~i.e. θ_{g_s} must be fixed~~) and $\sigma_{g_s}^2$ and precompute the Cholesky decomposition of the source components, ~~$L_{g_s} = L(K_{g_s} + \sigma_{g_s}^2 I_{N_{\text{source}}})$, then $L_{g_s} = L(K_{g_s} + \sigma_{g_s}^2 I_{N_{\text{source}}})$, then~~

$$L(\Omega_{\mathbf{g}}) = \begin{pmatrix} L_{g_s} & \mathbf{0} \\ (L_{g_s}^{-1} K_{g_s, g})^T & L(\tilde{K}_t) \end{pmatrix}, \quad (9)$$

$$\tilde{K}_t = K_g + \sigma_g^2 I_N - (L_{g_s}^{-1} K_{g_s, g})^T L_{g_s}^{-1} K_{g_s, g}.$$

This is obtained from the definition of Cholesky decomposition, i.e. $\Omega_{\mathbf{g}} = L(\Omega_{\mathbf{g}}) L(\Omega_{\mathbf{g}})^T$, and from the fact that a Cholesky decomposition exists and is unique for any positive definite matrix.

The complexity of computing $L(\Omega_{\mathbf{g}})$ thus becomes ~~$\mathcal{O}(N_{\text{source}}^2 N) + \mathcal{O}(N_{\text{source}} N^2) + \mathcal{O}(N^3)$~~ $\mathcal{O}(N_{\text{source}}^2 N) + \mathcal{O}(N_{\text{source}} N^2) + \mathcal{O}(N^3)$ ~~instead of $\mathcal{O}((N_{\text{source}} + N)^3)$~~ instead of $\mathcal{O}((N_{\text{source}} + N)^3)$. In particular, computing $L_{g_s}^{-1} K_{g_s, st}$ is ~~$\mathcal{O}(N_{\text{source}}^2 N)$~~ $\mathcal{O}(N_{\text{source}}^2 N)$, acquiring matrix product \hat{K}_t is ~~$\mathcal{O}(N_{\text{source}} N^2)$~~ $\mathcal{O}(N_{\text{source}} N^2)$ and Cholesky decomposition $L(\hat{K}_t)$ is $\mathcal{O}(N^3)$.

The learning procedure is summarized in ~~??~~ Algorithm 3 in the main script. We prepare a safe learning experiment with ~~$\mathcal{D}_{N_{\text{source}}}^{\text{source}}$~~ $\mathcal{D}_{N_{\text{source}}}^{\text{source}}$ and initial \mathcal{D}_N ; we fix ~~$\theta_{f_s}, \theta_{q_s^j}, \sigma_{f_s}, \sigma_{q_s^j}$~~ $\theta_{f_s}, \theta_{q_s^j}, \sigma_{f_s}, \sigma_{q_s^j}$ $j=1, \dots, J$ to appropriate values, and we precompute $L_{f_s}, L_{q_s^j}$. During the experiment, the fitting and inference of GPs (for data acquisition) are achieved by incorporating Equation (9) in ~~??~~ Equation (3) of the main script (Section 6).

D.2 Transfer Task GPs beyond One Source Tasks

We extend Section 3.2 beyond one single source task. Let us say we have a total of P source tasks, and the source task index is $s=1, \dots, P$. In our main paper (Section 3), $\mathcal{D}_{N_{\text{source}}}^{\text{source}}$ is the source data with only

one task. Here, $\mathcal{D}_{N_{\text{source}}}^{\text{source}} := \cup_{s=1}^P \mathcal{D}_{M_s}^s \subset \mathcal{X} \times \mathbb{R} \times \mathbb{R}$, $\mathcal{D}_{M_s}^s = \{\mathbf{x}_{s,1:M_s}, y_{s,1:M_s}, \mathbf{z}_{s,1:M_s}\}$ is the dataset of source task indexed by s , M_s is the number of data of task s , and $N_{\text{source}} = \sum_s M_s$ is still the number of data of all P source tasks jointly.

We now want to write down the predictive distributions for each $\mathbf{g} \in \{\mathbf{f}, \mathbf{q}^1, \dots, \mathbf{q}^J\}$. Similar to Section 3.2, $\hat{\mathbf{x}}_{s,1:M_s} = \{(\mathbf{x}_{s,n}, s)\}_{n=1}^{M_s} \subset \mathcal{X} \times \{\text{task indices}\}$ denotes the input data with task index. The data can be plugged in as how it was in Section 3.2, and the predictive distributions have only minor changes. We write \mathbf{f} as an example below in Equation (10), while $\mathbf{q}^1, \dots, \mathbf{q}^J$ are analogous. $\hat{\mathbf{x}}_* = (\mathbf{x}_*, t)$, $\mathbf{x}_* \in \mathcal{X}$ is again a test point and t is the index of target task. We color the modification compared to single source task (Equation (3)).

$$\begin{aligned}
p(\mathbf{f}(\mathbf{x}_*, t) | \mathcal{D}_N, \mathcal{D}_{N_{\text{source}}}^{\text{source}}) &= \mathcal{N}(\mu_{\mathbf{f}, N}(\mathbf{x}_*), \sigma_{\mathbf{f}, N}^2(\mathbf{x}_*)), \\
\mu_{\mathbf{f}, N}(\mathbf{x}_*) &= \mathbf{v}_{\mathbf{f}}^T \Omega_{\mathbf{f}}^{-1} \begin{pmatrix} y_{1,1:M_1} \\ \vdots \\ y_{P,1:M_P} \\ y_{1:N} \end{pmatrix}, \\
\sigma_{\mathbf{f}, N}^2(\mathbf{x}_*) &= k_{\mathbf{f}}(\hat{\mathbf{x}}_*, \hat{\mathbf{x}}_*) - \mathbf{v}_{\mathbf{f}}^T \Omega_{\mathbf{f}}^{-1} \mathbf{v}_{\mathbf{f}}, \\
\mathbf{v}_{\mathbf{f}} &= \begin{pmatrix} k_{\mathbf{f}}(\hat{\mathbf{x}}_{1,1:M_1}, \hat{\mathbf{x}}_*) \\ \vdots \\ k_{\mathbf{f}}(\hat{\mathbf{x}}_{P,1:M_P}, \hat{\mathbf{x}}_*) \\ k_{\mathbf{f}}(\hat{\mathbf{x}}_{1:N}, \hat{\mathbf{x}}_*) \end{pmatrix}, \\
\Omega_{\mathbf{f}} &= (K_{N_{\text{source}}+N}) + \begin{pmatrix} \sigma_{f_1}^2 I_{M_1} & 0 & & \\ 0 & \ddots & 0 & \\ & 0 & \sigma_{f_P}^2 I_{M_P} & 0 \\ & & 0 & \sigma_f^2 I_N \end{pmatrix},
\end{aligned} \tag{10}$$

where $[K_{N_{\text{source}}+N}]_{i,j} = k_{\mathbf{f}}([\hat{\mathbf{x}}_{\cup}]_i, [\hat{\mathbf{x}}_{\cup}]_j)$, and $\hat{\mathbf{x}}_{\cup}$ is a joint expression of source and target data $(\hat{\mathbf{x}}_{s=1,1:M_1}, \dots, \hat{\mathbf{x}}_{s=P,1:M_P}, \hat{\mathbf{x}}_{1:N})$ placed exactly in this order. The GP model \mathbf{f} is governed by the multitask kernel $k_{\mathbf{f}}$ and noise parameters $\sigma_{f_s}^2, \sigma_f^2$, where $\sigma_{f_s}^2$ is a noise variance of source task $s = 1, \dots, P$. The pre-computation will fix the part of all source tasks (still the top left N_{source} by N_{source} block of $\Omega_{\mathbf{f}}$).

Multitask Kernels: Few examples of actual GP models, i.e. actual kernels, are described as the following. The LMC, linear model of coregionalization, can be taken simply by adding more dimension:

$$k_{\mathbf{g}}((\cdot, \cdot), (\cdot, \cdot)) = \sum_{l=1}^{P+1} \left(\mathbf{w}_l \mathbf{w}_l^T + \begin{pmatrix} \kappa_1 & 0 & & \\ 0 & \ddots & 0 & \\ & 0 & \kappa_P & 0 \\ & & 0 & \kappa \end{pmatrix} \right) \otimes k_l(\cdot, \cdot),$$

where \mathbf{g} is a multitask function but does not matter to the expression here, each $k_l : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ is a standard kernel such as a Matérn-5/2 kernel encoding the l -th latent pattern, \otimes is a Kronecker product, and $\mathbf{W}_l \in \mathbb{R}^{(P+1) \times 1}$ and $\kappa_1, \dots, \kappa_P, \kappa > 0$ are task scale parameters (Álvarez et al., 2012). l is a numbering index used only here.

The HGP can be extended in two ways, models in Poloczek et al. (2017) or in Tighineanu et al. (2022). Here we take the model from Tighineanu et al. (2022):

$$k_g((\cdot, \cdot), (\cdot, \cdot)) = \sum_{i=0}^P \begin{pmatrix} \mathbf{0}^{i \times i} & \mathbf{0}^{i \times (P+1-i)} \\ \mathbf{0}^{(P+1-i) \times i} & \mathbf{1}^{(P+1-i) \times (P+1-i)} \end{pmatrix} \otimes k_i(\cdot, \cdot),$$

where $\mathbf{0}^{m \times n}$ and $\mathbf{1}^{m \times n}$ are matrices of shape m by n with all elements being zero and one, respectively, $m, n = 0, \dots, P$. $k_i(\cdot, \cdot)$ is a standard kernel such as a Matérn-5/2 kernel, i is a numbering index.

E Experiment details & Numerical Details

E.1 Labeling safe regions

The goal is to label disjoint safe regions, so that we may track the exploration of each land. In our experiments, the test safety values are always available because we are dealing with executed pool of data. It is thus possible to access safety conditions of each test point as a binary label. We access safety values as binary labels of equidistant grids (as if these are pixels). This is always possible for synthetic problems. We then perform connected component labeling (CCL, see He et al. (2017)) to the safety classes over grids (grids are available, see the following sections). This algorithm will cluster safe pixels into connected lands. When $D = 1$, this labeling is trivial. When $D = 2$, we consider 4-neighbors of each pixel (He et al., 2017). With simulated datasets, the ground truth is available, and thus For noise-free ground truth safety values, the CCL is deterministic. The CCL can be computationally intractable on high dimension (number of grids grows exponentially), and this method can be inaccurate over real data where because the observations are noisy and the grid values need interpolation from the measurements.

After clustering the safe regions over grids, we identify which safe region each test point \mathbf{x}_* belongs to by searching the grid nearest to \mathbf{x}_* . The accuracy can be guaranteed by considering grids denser than the pool. This is computationally possible only for $D = 1, 2$. See main Table 3 and the queried regions count of Figure 11 for the results.

E.2 Numerical details Setup & Datasets

When we run algorithm 1 and 2 (in the main paper) For our main experiments (Algorithm 1, Algorithm 2, Algorithm 3), we set N_{init} (N_{init} (number of initial observed target data), N_{source} (N_{source} (number of observed source data) and N_{pool} , N_{query} (number of AL queries/ learning iterations) and N_{pool} (size of discretized input space $\mathcal{X}_{pool}, \mathcal{X}_{pool}$) as follows:

1. GP1D: $N_{source} = 100$, $N_{init} = 10$, run Algorithm 1 or ?? for 50 iterations, and $N_{pool} = 5000$, $N_{source} = 100$, $N_{init} = 10$, $N_{query} = 50$, $N_{pool} = 5000$, constraints $q > 0$ up to noise;
2. GP2D: $N_{source} = 250$, $N_{init} = 20$, run Algorithm 1 or ?? for 100 iterations, and $N_{pool} = 5000$, $N_{source} = 250$, $N_{init} = 20$, $N_{query} = 100$, $N_{pool} = 5000$, constraints $q > 0$ up to noise;
3. Branin & Hartmann3: $N_{source} = 100$, $N_{init} = 20$, run Algorithm 1 or ?? for 100 iterations, and

~~$N_{pool} = 5000$~~ , $N_{source} = 100$, $N_{init} = 20$, $N_{query} = 100$,
 $N_{pool} = 5000$, $q = f > 0$ up to noise;

4. PEngine: ~~$N_{source} = 500$, $N_{init} = 20$, run Algorithm 1 or ?? for 100 iterations, and $N_{pool} = 3000$~~ , $N_{source} = 500$, $N_{init} = 20$, $N_{query} = 100$, and $N_{pool} = 3000$, constraints $q < 1$ up to noise;
5. GEngine: ~~$N_{source} = 500$, $N_{init} = 20$, run Algorithm 1 or ?? for 200 iterations, and $N_{pool} = 10000$~~ , $N_{source} = 500$, $N_{init} = 20$, $N_{query} = 200$, $N_{pool} = 10000$, $-1.5 < q < 0.5$ up to noise.

In the following, we describe in details how to prepare each dataset.

E.2.1 Synthetic Datasets of Tractable Safe Regions

We first sample source and target test functions and then sample initial observations from the functions. With GP1D, GP2D and Branin problems, we reject the sampled functions unless all of the following conditions are satisfied: (i) the target task has at least two disjoint safe regions, (ii) each of these regions has a common safe area shared with the source, and (iii) for at least two disjoint target safe regions, each aforementioned shared area is larger than 5% of the overall space (in total, at least 10% of the space is safe for both the source and the target tasks).

~~In our general test problems, i.e. Hartmann3, we generate functions as they are. In other words, we do not restrict the datasets to any safe region characteristics.~~

GP dataData: We generate datasets of two outputs. The first output is treated as our source task and the second output as the target task.

To generate the multi-output GP datasets, we use GPs with zero mean prior and multi-output kernel $\sum_{l=1}^2 W_l W_l^T \otimes k_l(\cdot, \cdot)$, where \otimes is the Kronecker product, each W_l is a 2 by 2 matrix and k_l is a unit variance Matérn-5/2 kernel (Álvarez et al., 2012). All components of W_l are generated in the following way: we randomly sample from a uniform distribution over interval $[-1, 1]$, and then the matrix is normalized such that each row of W_l has norm 1. Each k_l has a unit variance and a vector of lengthscale parameters, consisting of D components. For GP1D and GP2D problems, each component of the lengthscale is sampled from a uniform distribution over interval $[0.1, 1]$. We adapt algorithm 1 of Kanagawa et al. (2018) for GP sampling, detailed as follows:

1. sample input dataset $\mathbf{X} \in \mathbb{R}^{n \times D}$ within interval $[-2, 2]$, and $n = 100^D$.
2. for $l = 1, 2$, compute Gram matrix $K_l = k_l(\mathbf{X}, \mathbf{X})$.
3. compute Cholesky decomposition $L_l = L(W_l W_l^T \otimes K_l) = L(W_l W_l^T) \otimes L(K_l)$ (i.e. $W_l W_l^T \otimes K_l = L_l L_l^T$, $L_l \in \mathbb{R}^{2 \times n \times 2 \times n}$).

4. for $l = 1, 2$, draw $u_l \sim \mathcal{N}(\mathbf{0}, I_{2*n})$ ($u_l \in \mathbb{R}^{(2*n) \times 1}$).
5. obtain noise-free output dataset $\mathbf{F} = \sum_{l=1}^2 L_l u_l$
6. reshape $\mathbf{F} = \begin{pmatrix} \mathbf{f}(\mathbf{X}, s) \\ \mathbf{f}(\mathbf{X}, t) \end{pmatrix} \in \mathbb{R}^{2*n \times 1}$ into $\mathbf{F} = (\mathbf{f}(\mathbf{X}, s) \parallel \mathbf{f}(\mathbf{X}, t)) \in \mathbb{R}^{n \times 2}$.
7. normalize \mathbf{F} again s.t. each column has mean 0 and unit variance.
8. generate initial observations (more than needed in the experiments, always sampled from the largest safe region shared between the source and the target).

During the AL experiments, the generated data \mathbf{X} and \mathbf{F} are treated as grids. We construct an oracle on continuous space $[-2, 2]^D$ by interpolation. During the experiments, the training data and test data are blurred with a Gaussian noise of standard deviation 0.01 $\mathcal{N}(0, 0.01^2)$.

Once we sample the GP hyperparameters, we sample one main function \mathbf{f} and an additional safety function from the GP. During the experiments, the constraint is set to $\mathbf{q} \geq 0, z_s, z \geq 0$ (z_s, z are noisy q_s, q). For each dimension, we generate 10 datasets and repeat the AL experiments 5 times for each dataset. We illustrate examples of \mathbf{X} and \mathbf{F} in Figure 6 and Figure 7.

Branin data: The Branin function is a function defined over $(x_1, x_2) \in \mathcal{X} = [-5, 10] \times [0, 15]$ as

$$f_{a,b,c,r,s,t}((x_1, x_2)) = a(x_2 - bx_1^2 + cx_1 - r) + s(1 - t)\cos(x_1) + s,$$

where a, b, c, r, s, t are constants. It is common to set $(a, b, c, r, s, t) = (1, \frac{5.1}{4\pi^2}, \frac{5}{\pi}, 6, 10, \frac{1}{8\pi})$, which is our setting for target task.

We take the numerical setting of Tighineanu et al. (2022); Rothfuss et al. (2022) to generate five different source datasets (and later repeat 5 experiments for each dataset):

$$\begin{aligned} a &\sim \text{Uniform}(0.5, 1.5), \\ b &\sim \text{Uniform}(0.1, 0.15), \\ c &\sim \text{Uniform}(1.0, 2.0), \\ r &\sim \text{Uniform}(5.0, 7.0), \\ s &\sim \text{Uniform}(8.0, 12.0), \\ t &\sim \text{Uniform}(0.03, 0.05). \end{aligned}$$

After obtaining the constants for our experiments, we sample noise free data points and use the samples to normalize our output

$$f_{a,b,c,r,s,t}((x_1, x_2))_{\text{normalize}} = \frac{f_{a,b,c,r,s,t}((x_1, x_2)) - \text{mean}(f_{a,b,c,r,s,t})}{\text{std}(f_{a,b,c,r,s,t})}.$$

Then we set safety constraint $\mathbf{f} \geq 0, \mathbf{y} \geq 0$ (\mathbf{y} is noisy \mathbf{f}) and sample initial safe data. The sampling noise is Gaussian $\mathcal{N}(0, 0.01^2)$ during the experiments.

E.2.2 Hartmann3, PEngine, Gengine

Hartmann3 data: ~~The~~ Unlike GP and Branin data, we do not enforce disjoint safe regions, and do not track safe regions during the learning. The task generation is not restricted to any safe region characteristics.

The Hartmann3 function is a function defined over $\mathbf{x} \in \mathcal{X} = [0, 1]^3$ as

$$f_{a_1, a_2, a_3, a_4}((x_1, x_2, x_3)) = - \sum_i^4 a_i \exp \left(- \sum_{j=1}^3 A_{i,j} (x_j - P_{i,j})^2 \right),$$

$$\mathbf{A} = \begin{pmatrix} 3 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3 & 10 & 30 \\ 0.1 & 10 & 35 \end{pmatrix},$$

$$\mathbf{P} = 10^{-4} \begin{pmatrix} 3689 & 1170 & 2673 \\ 4699 & 4387 & 7470 \\ 1091 & 8732 & 5547 \\ 381 & 5743 & 8828 \end{pmatrix},$$

where a_1, a_2, a_3, a_4 are constants. It is common to set $(a_1, a_2, a_3, a_4) = (1, 1.2, 3, 3.2)$, which is our setting for target task.

We take the numerical setting of Tighineanu et al. (2022) to generate five different source datasets (and later repeat 5 experiments for each dataset):

$$\begin{aligned} a_1 &\sim \text{Uniform}(1.0, 1.02), \\ a_2 &\sim \text{Uniform}(1.18, 1.2), \\ a_3 &\sim \text{Uniform}(2.8, 3.0), \\ a_4 &\sim \text{Uniform}(3.2, 3.4). \end{aligned}$$

After obtaining the constants for our experiments, we sample noise free data points and use the samples to normalize our output

$$f_{a_1, a_2, a_3, a_4}((x_1, x_2, x_3))_{\text{normalize}} = \frac{f_{a_1, a_2, a_3, a_4}((x_1, x_2, x_3)) - \text{mean}(f_{a_1, a_2, a_3, a_4})}{\text{std}(f_{a_1, a_2, a_3, a_4})}.$$

Then we set safety constraint ~~$f \geq 0$~~ $y \geq 0$ (y is noisy f) and sample initial safe data. The sampling noise is Gaussian during the experiments $\mathcal{N}(0, 0.01^2)$.

PEngine data: We have 2 datasets, measured from the same prototype of engine under different conditions. Both datasets measure the temperature, roughness, emission HC, and emission NOx. The inputs are engine speed, relative cylinder air charge, position of camshaft phaser and air-fuel-ratio. The contextual input variables "position of camshaft phaser" and "air-fuel-ratio" are desired to be fixed. These two contextual inputs are recorded with noise, so we interpolate the values with a multi-output GP simulator. We construct a LMC trained with the 2 datasets, each task as one output. During the training, we split each of the datasets (both safe and unsafe) into 60% training data and 40% test data. After the model parameters

are selected, the trained models along with full dataset are utilized as our GP simulators (one simulator for each output channel, e.g. temperature simulator, roughness simulator, etc). The first output of each GP simulator is the source task and the second output the target task. The simulators provide GP predictive mean as the observations. During the AL experiments, the input space is a rectangle spanned from the datasets, and $\mathcal{X}_{\text{pool}} - \mathcal{X}_{\text{pool}}$ is a discretization of this space from the simulators with $N_{\text{pool}} = 3000$, $N_{\text{pool}} = 3000$. We set $N_{\text{source}} = 500$, $N_{\text{source}} = 500$, $N = 20$ (initially) and we query for 100 iterations ($N = 20 + 100$). When we fit the models for simulators, the test RMSEs (60% training and 40% test data) of roughness is around 0.45 and of temperature around 0.25.

In a sequential learning experiment, the surrogate models are trainable GP models. These surrogate models interact with the simulators, i.e. take $\mathcal{X}_{\text{pool}} - \mathcal{X}_{\text{pool}}$ from the simulators, infer the safety and query from $\mathcal{X}_{\text{pool}} - \mathcal{X}_{\text{pool}}$, and then obtain observations from the simulators. In our main [Algorithms 1 to 3](#), the surrogate models are the GP models while the GP simulators are systems that respond to queries \mathbf{x}_* .

GEngine [data](#)Data: This problem has two datasets, one taken as the source task and one as the target task. Both datasets were published by [Li et al. \(2022\)](#). Each dataset is split into training set and test set. The original datasets have the following inputs: (1) the first dataset has speed, load, lambda, ignition angle, and fuel cutoff (dimension $D = 5$) which we take as the source task (2) speed, load, lambda, and ignition angle ($D = 4$, no fuel cutoff) which we take as the target task. The 5th input of the source data, fuel cutoff, is irrelevant and we exclude it (not used in the original paper). Please see [Figure 8](#) for the data histogram. The datasets are dynamic and are available with a nonlinear exogenous (NX) history structure, concatenating the relevant past points into the inputs (handled by [Li et al. \(2022\)](#) in their published code). The final input dimension of this problem is $D = 13$. As outputs, the source dataset measures the temperature, emission particle numbers, CO, CO2, HC, NOx, O2 and temperature. The target dataset measures particle numbers, HC, NOx and temperature. We take HC as our main learning output and temperature as the constraints.

Both the source and target datasets have hundreds of thousands of data, but [Li et al. \(2022\)](#) discover that the performance saturates with few thousand randomly selected points or with few hundred actively selected points. We thus decide to run our experiments with $N_{\text{pool}} = 10000$, $N_{\text{pool}} = 10000$, a random subset of the training set. This pool subset is sampled before we compute the acquisition scores in each iteration. Furthermore, we start our AL experiments with $N_{\text{init}} = 20$, $N_{\text{init}} = 20$ and we query for 200 iterations. The initial target data are sampled from the following input domain (written in the original space, no NX history structure here) $[-1, -0.7] \times (-\infty, -0.5] \times [0, 0.5] \times [0, 0.2]$. This domain is chosen by taking the density peak of the inputs, see row 3 of [Figure 8](#) for the data histogram. Note that values of datasets were normalized.

In this problem, the effect of one single query on the GP hyperparameters is not obvious. Therefore, to speed up the experiments, we train the hyperparameters only every 50 queries (and at the beginning). The constraint is temperature $-1.5 \leq q \leq 0.5$, $-1.5 \leq z \leq 0.5$, and source temperature

~~$-2 \leq q_s \leq 0.5$~~ ~~$-2 \leq z_s \leq 0.5$~~ . The temperature lower bound matters only to the outliers, it is the upper bound 0.5 that plays the major role. The overall safe set is around 65% of the input space (target test set).

F Ablation Studies and Further Experiments

In this section, we provide ablation studies on the size of source dataset.

~~One source-task~~Source Task, ~~varied N_{source}~~ Varied N_{source} : We perform experiments on the Branin function. The results are presented in Figure 9. The first conclusion is that all of the ~~multioutput-multitask~~ methods outperform baseline safe AL (safe AL result shown in ~~??~~Figure 4). Note again that the RMSEs are evaluated on the entire space while the baseline safe AL explore only one safe region. In addition, we observe that more source data result in better performances, i.e. lower RMSE and larger safe set coverage (TF area), while there exist a saturation level at around ~~$N_{source} = 100$~~ $N_{source} = 100$.

~~Multiple source-tasks~~Source Tasks: Next, we wish to manipulate the number of source tasks. ~~Before presenting the results, we first introduce the model on multiple source tasks. In this paragraph, we say D_{source} is the number of source tasks. As described in ??, each $\mathbf{g} \in \{\mathbf{f}, \mathbf{q}^1, \dots, \mathbf{q}^J\}$ is a multi-output GP correlating source and target tasks. The LMC, linear model of coregionalization, can be taken without any change: $k_{\mathbf{g}} = \sum_l (\mathbf{W}_l \mathbf{W}_l^T + \text{diag}\{\boldsymbol{\kappa}\}) \otimes k_l(\cdot, \cdot)$, where $k_l(\cdot, \cdot)$ is a standard single task kernel as in Assumption 3.2, and \mathbf{W}_l and $\boldsymbol{\kappa}$ are vector of $(D_{source} + 1)$ elements (Álvarez et al., 2012).~~ The HGP can be extended into two ways, models in Poloczek et al. (2017) or in Tighineanu et al. (2022). Here we take the model from Tighineanu et al. (2022): ~~$k_{\mathbf{g}} = \sum_{i=0}^{D_{source}} \text{Mask}_i \otimes k_i(\cdot, \cdot)$, $\text{Mask}_i \in \mathbb{R}^{D_{source}+1 \times D_{source}+1}$ is a matrix where the first i rows and columns are zero and the other entries are all one (all elements of Mask_0 are ones). One can see that if $D_{source} = 1$, then we get the HGP described in ?? by reindexing k_0 and k_1 here.~~

The transfer task GP formulation and the exact models are described in Appendix D.2. We take LMC and HGP with Matérn-5/2 kernels as the base kernels. In this study, we generate source data with constraints, but disjoint safe regions requirement when we sample the source tasks and data (in ~~??~~Figure 4, the data are generated s.t. source and target task has large enough shared safe area). We consider 1, 3 or 4 source tasks, and we generate 20 or 30 data points per task (Figure 10). In general, we see that 3 source tasks significantly outperform 1 source task while the performance saturates as adding 10 more points per source task seems to benefit more than adding one more source task. Note here that all source data are generated independently, i.e. the observations of each task are not restricted to the same input locations.

~~Further plots~~Plots and experimentsExperiments: ~~In ??, we track the safe region of each query in AL experiments. We measure the model fitting time per iteration as well.~~ The main Table 3 and Table 4 present only the summary results. In Figure 11, we additionally provide the region clustering and fitting time w.r.t. AL iterations. Furthermore, Table 5 counts ~~among~~ the AL selected queries which, after a safety measurements are accessed, actually satisfy the safety constraints. This table is a sanity check that the methods are selecting points safely.

Table 5: Ratio of ~~safe queries~~ Safe Queries

methods num_steps N_{query}	GP1D +z	GP2D +z	Branin	Hartmann3	GEngine
	50	100	100	100	200
EffTransHGP	0.986 ± 0.001	0.974 ± 0.002	0.999 ± 0.0006	0.972 ± 0.003	0.936 ± 0.003
FullTransHGP	0.979 ± 0.004	0.952 ± 0.005	0.9996 ± 0.0004	0.972 ± 0.003	0.947 ± 0.01
FullTransLMC	0.984 ± 0.002	0.969 ± 0.002	0.993 ± 0.0009	0.968 ± 0.003	0.91 ± 0.008
Rothfuss2022	0.975 ± 0.003	0.905 ± 0.006	1.0 ± 0.0	0.84 ± 0.011	0.765 ± 0.035
SAL	0.995 ± 0.001	0.958 ± 0.005	1.0 ± 0.0	0.966 ± 0.002	0.954 ± 0.005

Ratio of all queries selected by the methods which are safe in the ground truth (initial data not included, see [??-Section 7](#) for the experiments). This is a sanity check in addition to FP safe set area, demonstrates that all the methods are safe during the experiments. Note that our benchmark problems all have around 35% to 65% of the space unsafe. Note that $\beta = 4$ implies that, with a well-fitted safety GP, we tolerate a 2.275% probability of unsafe evaluations. PEngine results are not shown because the queries are all safe (the modeling FP safe set area is almost zero in this problem, see [??-Figure 4](#) and Figure 12).

With the PEngine datasets, we perform additional experiments of learning $\mathbf{f} = \mathbf{q}$ =temperature, and the results are shown in Figure 12.

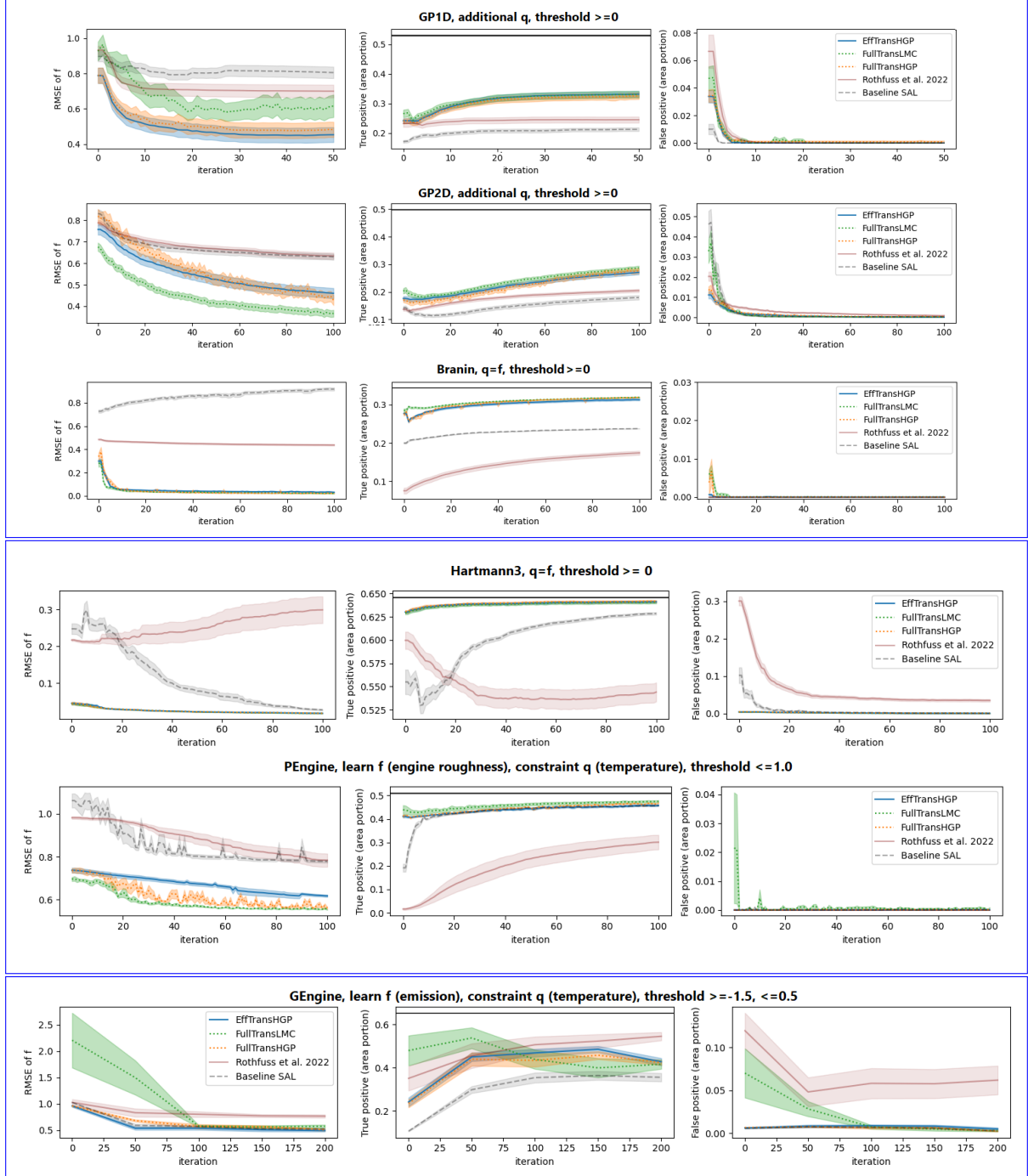


Figure 4: Safe AL experiments on three Empirical performance across all six benchmark datasets. GP data: f and safety function $q \geq 0$ over $\mathcal{X} = [-2, 2]^D$ RMSE to assess model convergence, $D=1$ ($N_{source}=50$, $N_{init}=10$, 50 data points are queried) or $D=2$ ($N_{source}=250$, $N_{init}=20$, 100 data points are queried). Branin data: constraint $q = f \geq 0$, $N_{source}=100$, $N_{init}=20$, 100 data points are queried. The results are mean and one standard error TP rate to measure the coverage of 100 (GP data) or 25 (Branin data) experiments. The test points for RMSEs are sampled from all of the true safe areaspace explored, including the regions individual methods (e.g., SAL) may fail and FP rate to exploreevaluate the safety of each approach. Note that FullTransLMC has more than ten model parameters, while in GP1D dataset we start with $N=10$. Both TP /and FP safe areas are portion of compute the input space rates to the area of \mathcal{X}_{pool} . Ground true-The ground truth safe area portion of for each dataset is marked indicated by a black line in the second column. Please also see appendix Figure 11 for fitting time Our approach generally shows improved convergence in terms of model performance and region cluster the extent of each query explored safe regions, while maintaining safety levels comparable to the baseline SAL. On GEngine, we additionally provide a zoomed-in RMSE figure (Figure 5).

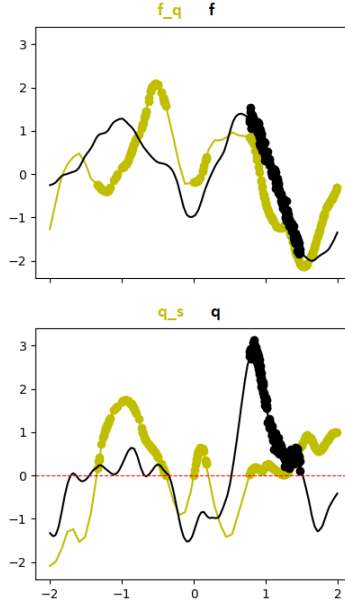


Figure 6: Example simulated GP data of $D = 1$, f is the function we want to learn (top), under an additional safety constraint $q \geq 0$ (bottom). The curves are true source (yellow) and target (black) functions. The dots are safe source data and a pool of initial target ticket (this pool of target data are more than those actually used in the experiments).

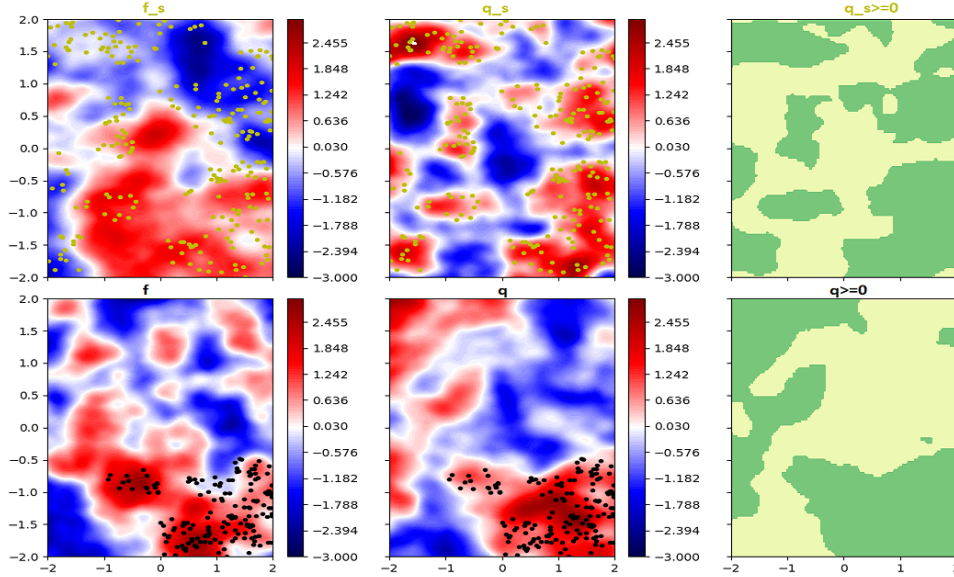


Figure 7: Example simulated GP data of $D = 2$, f is the function we want to learn (left), with an additional safety function q (middle), and the green is true safe regions $q \geq 0$ (right). The top is source task and the bottom is target task. The dots are safe source data and a pool of initial target ticket (this pool of target data are more than those actually used in the experiments).

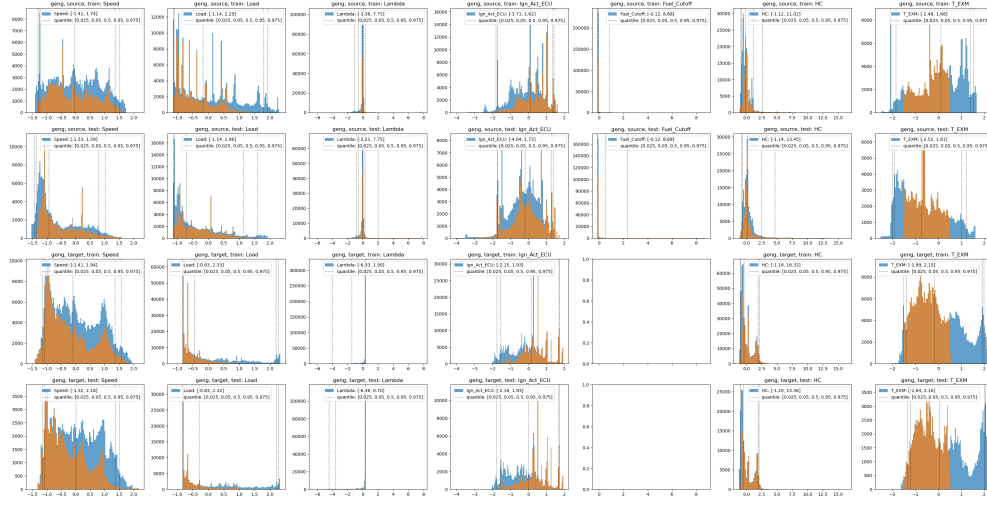


Figure 8: The histogram of GEngine data. The first 5 columns are inputs without NX history structure, the second last column is the output we model with f, f_s , and the last column is the temperature constraint. The rows are the following in order: (1) source task training set, (2) source task test set (not used in the experiments), (3) target task training set, and (4) target task test set. Blues are the histograms of raw data, and oranges are subsets if we add constraints on the temperature channel.

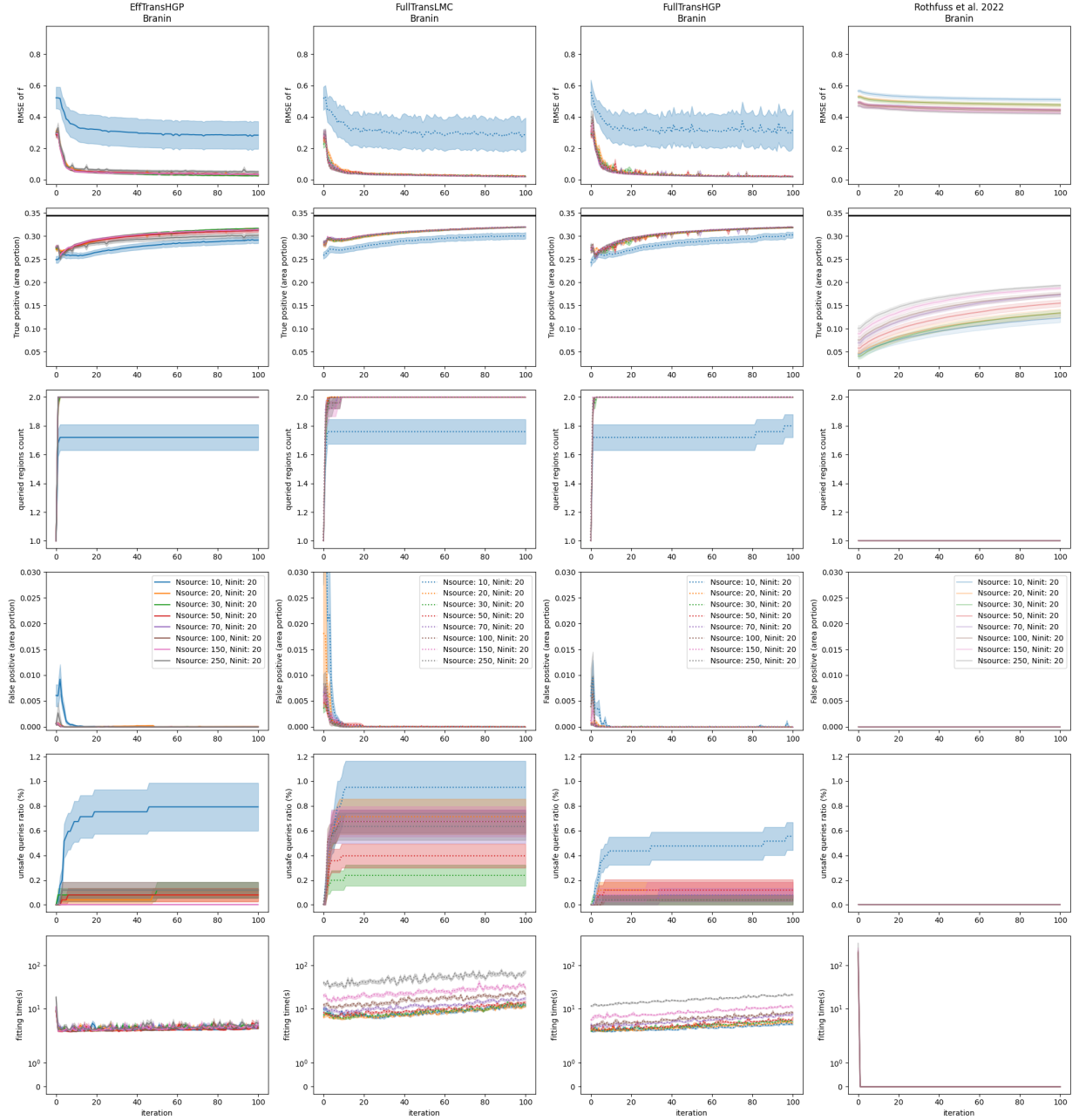


Figure 9: Safe AL experiments: Branin data with different number of source data. Each **multi-task-multitask** method is plotted in one column. The results are mean and one standard error of 25 experiments per setting. $\mathcal{X}_{\text{pool}}, \mathcal{X}_{\text{pool}}$ is discretized from \mathcal{X} with $N_{\text{pool}} = 5000, N_{\text{pool}} = 5000$. The TP/FP areas are computed as number of TP/FP points divided by $N_{\text{pool}}, N_{\text{pool}}$ (i.e. TP/FP as portion of $\mathcal{X}_{\text{pool}}, \mathcal{X}_{\text{pool}}$). The third row shows the number of disjoint safe regions explored by the queries. The fifth row, the unsafe queries ratio, are presented as percentage of number of iterations (e.g. at the 2nd-iteration out of a total of 100 iterations, one of the two queries is unsafe, then the ratio is 1 divided by 100). The last row demonstrates the model fitting time. At the first iteration (iter 0-th), this includes the time for fitting both the source components and the target components (EffTransHGP). With Rothfuss et al. 2022, source fitting is the meta learning phase.

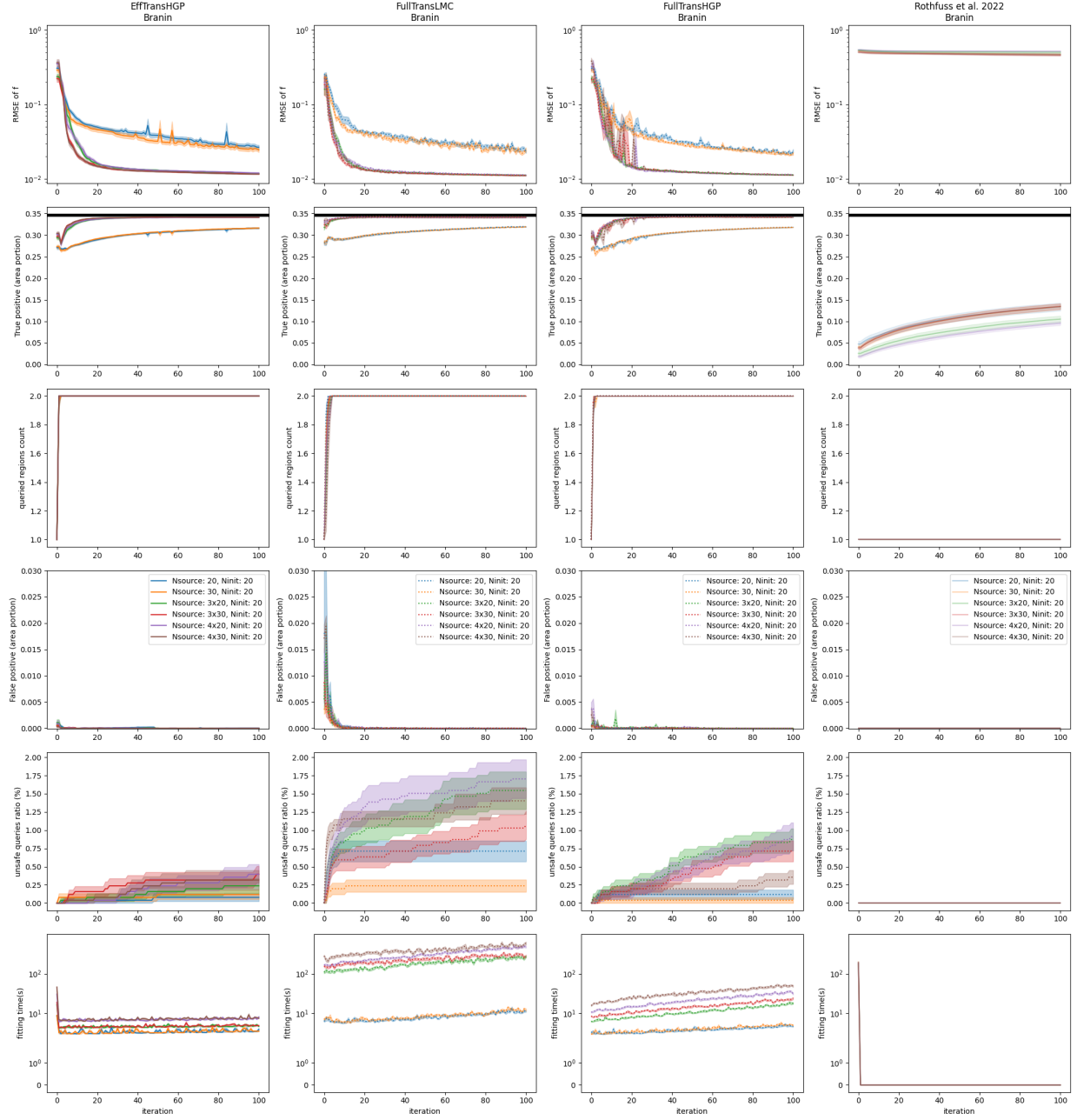


Figure 10: Safe AL experiments [with more than one source tasks](#): Branin data with multiple source tasks. Each [multi-task-multitask](#) method is plotted in one column. We consider 1, 3 or 4 source tasks and sample 20 or 30 data points per task. The remaining setting is the same as described in Figure 9. RMSE plots are plotted in log scale.

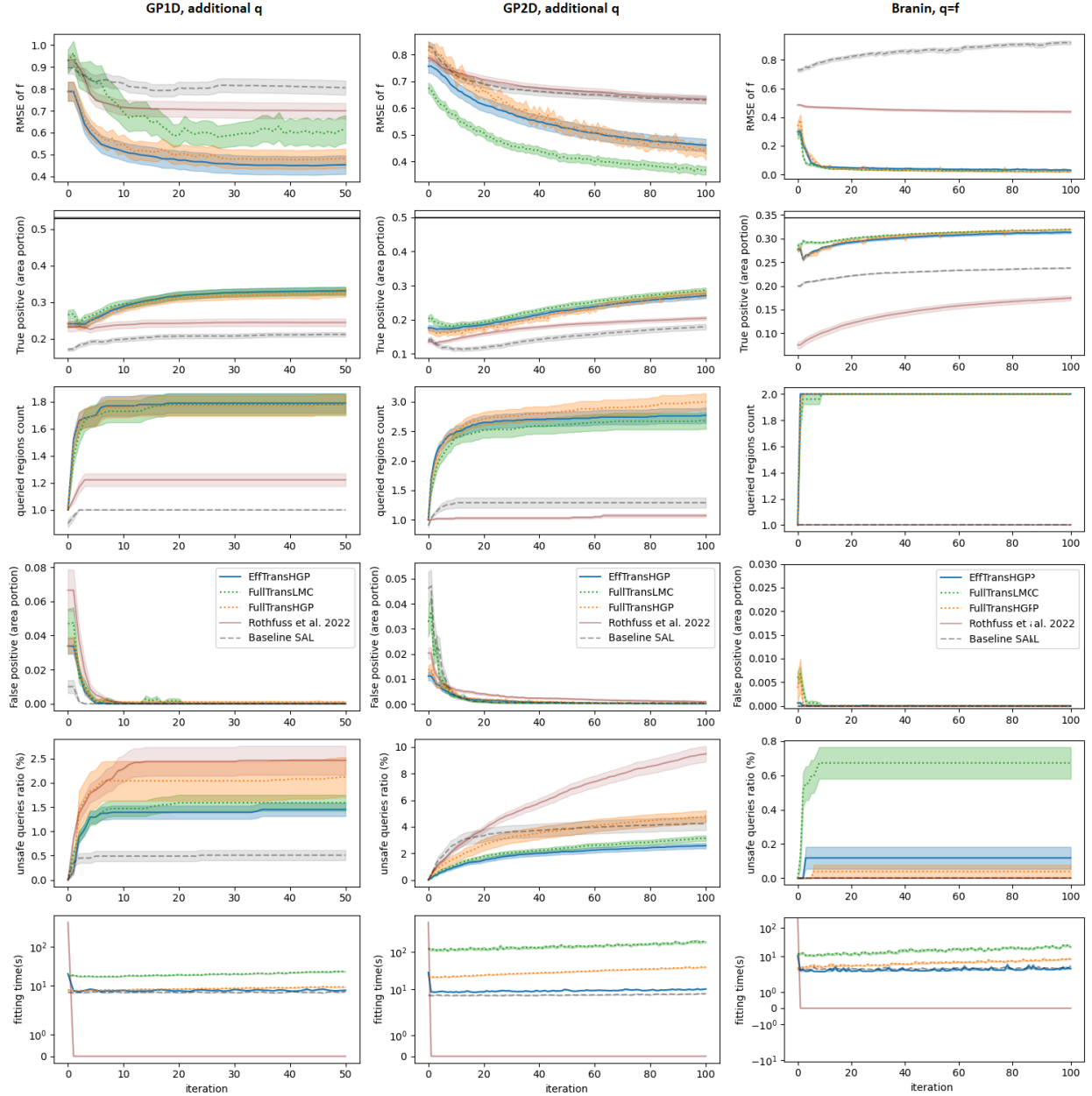


Figure 11: Safe AL experiments on three benchmark datasets: GP data with $\mathcal{X} = [-2, 2]^D$, $D = 1$ or 2 , constrained to $q \geq 0$, and the benchmark Branin function with constraint $f \geq 0$. The results are mean and one standard error of 100 (GP data) or 25 (Branin data) experiments. $\mathcal{X}_{\text{pool}} - \mathcal{X}_{\text{pool}}$ is discretized from \mathcal{X} with $N_{\text{pool}} = 5000$, $N_{\text{pool}} = 5000$. We set $N_{\text{source}} = 100$, $N_{\text{source}} = 100$ and N is from 10 (0th iteration) to 60 (50th iteration) for GP1D, $N_{\text{source}} = 250$, $N_{\text{source}} = 250$, N is 20 to 120 for GP2D, and $N_{\text{source}} = 100$, $N_{\text{source}} = 100$, N is 20 to 120 for Branin. The first, second and fourth rows are presented in ?? Figure 4 of the main paper. The TP/FP areas are computed as number of TP/FP points divided by $N_{\text{pool}} - N_{\text{pool}}$ (i.e. TP/FP as portion of $\mathcal{X}_{\text{pool}} - \mathcal{X}_{\text{pool}}$). The third row shows the number of disjoint safe regions explored by the queries (main Table 3 is taken from the last iteration here). The fifth row, the unsafe queries ratio, are presented as percentage of number of iterations (e.g. at the 2nd-iteration out of a total of 50 iterations, one of the two queries is unsafe, then the ratio is 1 divided by 50). The last row demonstrates the model fitting time. At the first iteration (iter 0-th), this includes the time for fitting both the source components and the target components (EffTransHGP). With Rothfuss et al. 2022, source fitting is the meta learning phase.

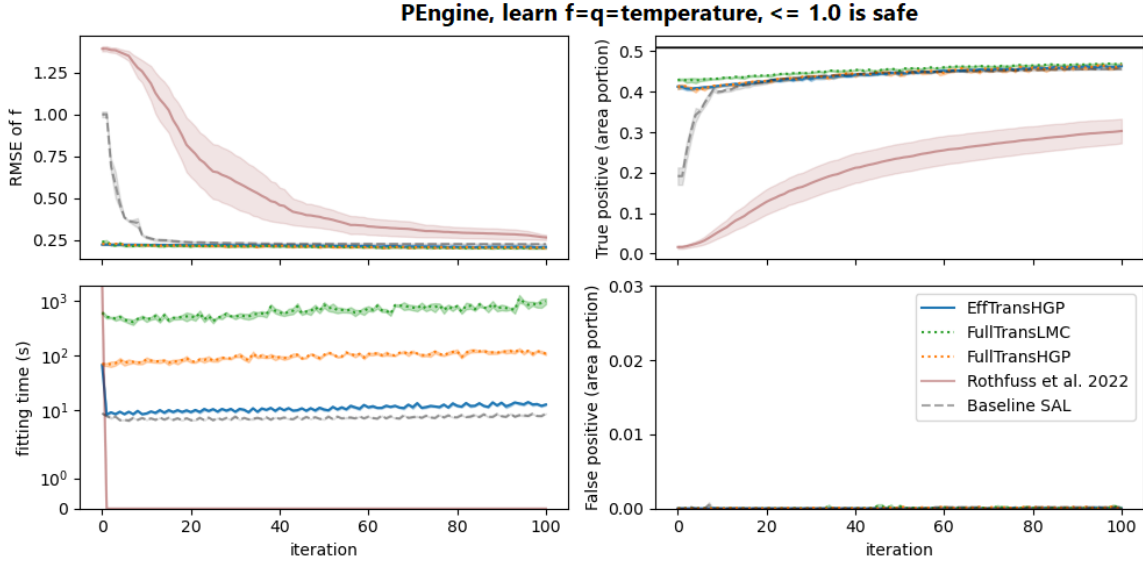


Figure 12: Safe AL experiments on PEngine temperature, AL on f (temperature) constrained by $q = f \leq 1.0$. Baseline is safe AL without source data. Transfer is LMC without modularization. Efficient_transfer is HGP with fixed and pre-computed source knowledge. $N_{source} = 500$, $N_{source} = 500$, N is from 20 to 120. The results are mean and one standard error of 5 repetitions. The fitting time is in seconds.