Journal of Intelligent Manufacturing (2024) 35:885–903
https://doi.org/10.1007/s10845-023-02087-3

Safe contextual Bayesian optimization integrated in industrial control for self-learning machines

Stefano De Blasi[1,2] · Maryam Bahrami[1] · Elmar Engels[2] · Alexander Gepperth[2]

Received: 7 July 2022 / Accepted: 25 January 2023 / Published online: 13 February 2023

Abstract

Intelligent manufacturing applications and agent-based implementations are scientifically investigated due to the enormous potential of industrial process optimization. The most widespread data-driven approach is the use of experimental history under test conditions for training, followed by execution of the trained model. Since factors, such as tool wear, affect the process, the experimental history has to be compiled extensively. In addition, individual machine noise implies that the models are not easily transferable to other (theoretically identical) machines. In contrast, a continual learning system should have the capacity to adapt (slightly) to a changing environment, e.g., another machine under different working conditions. Since this adaptation can potentially have a negative impact on process quality, especially in industry, safe optimization methods are required. In this article, we present a significant step towards self-optimizing machines in industry, by introducing a novel method for efficient safe contextual optimization and continuously trading-off between exploration and exploitation. Furthermore, an appropriate data discard strategy and local approximation techniques enable continual optimization. The approach is implemented as generic software module for an industrial edge control device. We apply this module to a steel straightening machine as an example, enabling it to adapt safely to changing environments.

Keywords Safe optimization · Intelligent manufacturing · Automation · Self-learning · Continual learning

Introduction

Reaching the vision of Industry 4.0 can be interpreted as a two step revolution (Rauch, 2020): First, technology-driven innovations enabling data processing, and second, data- and algorithm-driven innovations. While the first point becomes more and more concrete, the second point remains a broad research field. The combination has the potential to enable the vision of self-optimizing machines (Möhring et al., 2020; Iwanek et al., 2018; Permin et al., 2016) adapting to uncontrollable variables (Kim et al., 2018), e.g. measured thickness of material or environmental temperature. Accordingly, intelligent manufacturing applications (Azizi, 2019) and agent-based implementations (Zhong et al., 2017) are scientifically investigated due to their enormous potential to improve efficiency. As manufacturing complexity grows, it becomes increasingly challenging to tune process parameters with traditional methods to achieve the best process quality, e.g. minimal actual/target deviation, optimal product quality or minimal operating time. For this reason, methods of machine learning (ML) are increasingly investigated even in the conservative automation sector (Weichert et al., 2019; Zeiser et al., 2021) to contribute to industrial visions of the future (Zuehlke, 2010).

The currently most widespread approach to training a ML model for process optimization uses experimental history under test conditions, followed by execution of the trained model. Here, evolutionary techniques (e.g. genetic algorithms) or artificial neural networks (ANNs) (Azizi, 2020) can be used. For example, an optimizing control system uses ANNs trained beforehand to self-adjust cutting parameters like feed rate (Park & Tran, 2014). Thus, actors operate following a system behavior function based on real-time sensor data to ensure the best possible operating result (Qu et al., 2019). Since these optimal parameters depend, among other

✉ Stefano De Blasi
Stefano.DeBlasi@BoschRexroth.de

[1] Bosch Rexroth AG, Bürgermeister-Dr.-Nebel-Straße 2, 97816 Lohr am Main, Germany

[2] University of Applied Sciences Fulda, Leipziger Str. 123, 36037 Fulda, Germany

factors, on tool wear, the experimental history has to be compiled extensively to include all wear conditions. Furthermore, individual machine noise can mean that the models are not easily transferable to other machines identical in construction and only work on the test machine. Besides this disadvantage, one important aspect is the deployment of agents in control architecture (Ghafoorpoor Yazdi et al., 2018), which might become necessary as soon as a previously unconsidered variable in the environment changes.

In general, control variables can be process parameters which are either adapted before the process, or during the process depending on the environmental behavior. While the latter describes a full reinforcement learning task (Geramifard et al., 2013), we focus on the former case, which is an associative search task aiming to solve so-called contextual bandit problems. This focus is justified by the fact that most tasks within production processes are on the one hand recurring and on the other hand divided into several subtasks. Due to the online model adaption under consideration of historical decisions, solving contextual bandit problems has even potential to avoid catastrophic forgetting (Bouneffouf & Rish, 2019) and enable so-called continual, life-long or permanent learning. These are the most common classes of learning algorithms for self-adaptive systems (Gheibi et al., 2021).

Since self-optimization of a process is not supposed to have a negative impact on the quality of this process, standard learning methods are not applicable. In contrast to just focusing on minimizing the required number of real-world experiments (Maier et al., 2020), an optimization with constraints (Rattunde et al., 2021) is more appropriate for industrial standards. Such safe learning methods become particularly relevant for real-world interacting applications of machines (Chatzilygeroudis et al., 2018). Even if safe learning approaches receive increasing attention (Garcıa & Fernández, 2015), the safe optimization of manufacturing-like tasks is still a rather unexplored research topic with predominantly high cycle times. Due to their computational cost, current safe optimization approaches focus mainly on black-box problems, which are expensive to execute (Sergeyev et al., 2020). Another problem is the strict application-orientation of current solutions, e.g. hybrid solutions (Azizi, 2020). In the extreme case, even a rule-based adaption can be regarded as a primary intelligence level (Vargas et al., 2016), which might even be based on application-engineering exclusively. To take a significant step towards self-optimizing machines, it is essential that approaches are generic and do not require significant modifications to adapt them to other applications.

Therefore, we present a novel method for efficient safe optimization of contextual bandit-problems in this article. Our approach can be adjusted by the hyper-parameter setup to be applied to different applications, which makes it less application-oriented than former industrial process optimization approaches. We modify the contextual Bayesian optimization so that safety requirements are met and real-time capability is satisfied by using local approximations and intelligent forgetting strategies that increase efficiency. These modifications lead to the first safe optimization method for continual use with balancing exploration-exploitation. Furthermore, a generic software module is implemented for the use within industrial edge control and applied to real-world manufacturing machines in order to enable the safe self-optimization to changing environments and realize the vision of self-learning plants to a certain extent. We applied the module to a real-world industrial machine, see Fig. 1, and evaluated our approach.

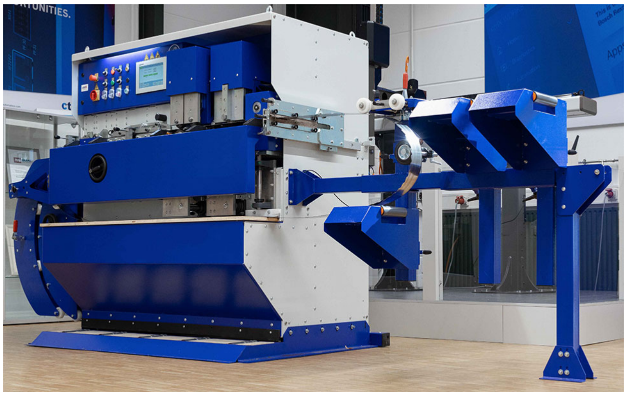

Fig. 1 A straightening machine for saw blades works the steel via adjustable process parameters in order to perform processing in the least possible time. The optimal parameters depend on the working conditions, e.g. steel characteristics, target shape and environmental temperature

Related work

As achieving the vision of self-optimizing machines is an interdisciplinary endeavor, the related work is wide-ranging and will be divided into three subsections, each of which will be enriched by the presented work.

Data driven process optimization in industrial control

Although the real-time execution of an ANN in industrial control was implemented early on Parrot and Venayagamoorthy (2008), this is intended exclusively for executing a previously trained model and not for enabling self-learning machines with continual optimization of the process quality. Such optimization problem formulations, on the other hand, can be addressed by reinforcement learning (RL) approaches. While semi-automated approaches require the operator to approve proposed improvements (Kirschner et al., 2019), the vision of self-optimization is supposed to be fully automatic.

This can be achieved in industrial systems for state-of-the-art RL algorithms (Gulde et al., 2019; De Blasi et al., 2021) or with safety-constrained optimization (Rattunde et al., 2021). Here, the decisions are automatically communicated on-demand between the two devices: The control unit providing the real-time capable system to fulfill the industrial requirements and a computer providing the ML framework to enable iterative learning. Instead of exchanging decisions, a message broker communication between the two software systems can be established to exchange the whole model to be executed on the controller (Schmidt et al., 2020). Furthermore, simple RL algorithms using discrete state-action-space and their framework can also be implemented directly within the control code (Demirkıran et al., 2020; Schwung et al., 2017; Hameed & Schwung, 2020). However, especially if this is intended for more complex algorithms (Schwung et al., 2018), the approach is time-consuming or even unfeasible, depending on the control system (Nian et al., 2020).

While the majority of efforts focuses on incorporating an ML framework into process control, it is not addressed how safety guarantees in an industrial environment are ensured. For instance, RL methods usually learn through errors, which cannot be tolerated in an industrial setting. In addition, we also contribute to the integration of an ML framework by using the capabilities of new control units to make communication highly efficient, secure and flexible to multiple solution architectures. This novel concept can be used in the future on a single edge device or distributed.

Safe Bayesian optimization

Three types of safe optimization can be defined. First, aiming for a safe optimum and allow unsafe evaluations while learning (Gelbart et al., 2014; Hernández-Lobato et al., 2016). Second, so-called *conservative* methods (Wu et al., 2016; Kazerouni et al., 2017; Jagerman et al., 2020) define safety as an improvement of the cumulative reward and also allow unsafe evaluations. Third, each action should not violate specific constraints of the environment. For most industrial self-optimization settings, only the third type of safety definition is suitable as the production should constantly provide satisfactory, near-optimal performance under a high level of safety conditions. Thus, the second field of related work addresses the algorithmic state-of-the-art regarding the prevention of fatal outcome during the optimization.

Stochastic approaches have proved their worth by the possibility of guaranteeing safety during learning (Sui et al., 2015; Sergeyev et al., 2020; Liu et al., 2019; Turchetta et al., 2019), e.g. based on particle swarm optimization (Duivenvoorden et al., 2017). This certainty is particularly important for industrial applications (Akametalu et al., 2014; Fisac et al., 2018). For example, the intelligent policy search for a physical system with minimized risk of catastrophic failure (Polymenakos et al., 2019) and the safe exploration of the system behavior (Liu et al., 2019) are current research topics using Gaussian process modeling. These methods concentrate mainly on safely solving a problem once, without focus on the computational cost, due to their experimental construction. While high-dimensional problems can be safely solved using subspace techniques (Kirschner et al., 2019; De Blasi et al., 2021), the growing number and dimensionality of data points raise a computational challenge for continual optimization. Furthermore, the involvement of context to the optimization is only possible to a very limited extent, which also severely restricts usability (Fiducioso et al., 2019; De Blasi, 2019; Berkenkamp et al., 2021).

In contrast to related work, we intend to continuously delete less informative data so that safe continual optimization can be enabled in a real-time capable way. Furthermore, we extend the state-of-the-art safe optimization concept to include high-dimensional context, which allows self-learning systems to safely adapt to environmental variables.

Efficient Gaussian process regression

The underlying Gaussian process (GP) surrogate model of Bayesian optimization leads to cubic complexity $\mathcal{O}(N^3)$ with N data points, which complicates its use for larger amount of data (Liu et al., 2020). As a reduced calculation time enables applications for real-time ML (Boedecker et al., 2014; Bui et al., 2017), it is the focus of several works using different approaches, e.g. by approximating the covariance matrix (Williams & Seeger, 2001). State-of-the-art approaches optimally choose the $M \ll N$ most important data points for regression, which is referred to as sparse models. These can be separated into two classes. The first one is subset-based and leads, in the simplest case, to a random subset of M points to reduce regression complexity to $\mathcal{O}(M^3)$ (Quiñonero-Candela & Rasmussen, 2005). Further approaches are a gradual deletion of old data (Oba et al., 2001; Csató & Opper, 2002), and a selection based on Kullback-Leibler divergence (Seeger et al., 2003) or entropy (Herbrich et al., 2003). Another strategy is a local approximation of the full GP close to the desired prediction, e.g., by moving neighborhoods (Cressie, 1990). Here, the selection of neighborhoods strongly affects prediction accuracy (RENARD & YANCEY, 1984). While, in the past, a neighborhood size of up to 30 points was common (Chilès & Desassis, 2018), moving neighborhood approaches have become less common as technical advances have made regressions with significantly more points possible. The renaissance of such approaches in recent years is rooted in the relevance of Bayesian online optimization for real-world applications, e.g., by incrementally updating the corresponding neighborhood sets (Gramacy & Apley, 2015; Gramacy, 2016). Here, whenever a distance threshold is reached, a new local model is created and the predictions of the models are combined via a weighted distance (Nguyen-Tuong et al., 2009). In contrast to these subset methods, the class of inducing point methods of sparse models aims to obtain M calculated support points with the maximum of information representing the regression. The selection of these points strongly affects accuracy (Titsias, 2009; Hensman et al., 2013). Here, the regression complexity is $\mathcal{O}(NM^2)$ and for both sparse model classes the prediction complexity is reduced to $\mathcal{O}(M^2)$ (Snelson & Ghahramani, 2006; Titsias, 2009). In this way, modeling with a fixed number of M points allows the use of sparse models for continual online learning with streaming data (Bui et al., 2017, 2018) as the computational complexity can be well estimated.

While the efficient prediction at a particular point is well studied, safe Bayesian optimization methods do not only require a single, but multiple predictions at different points, the so-called *candidates*. To the best of the authors' knowledge, an efficient safe Bayesian optimization by using Gaussian process approximations has not yet been presented. This gap is addressed within this work by combining several concepts mentioned in this section and justifying their applicability.

Scientific gap and own contributions

The continuity and the real-time capability of safe Bayesian optimization is insufficiently investigated for the application in industrial environments. From an algorithmic point of view, we contribute to the state-of-art by combining multiple methodologies for the first efficient, safe contextual Bayesian optimization, which can be applied for continual learning. This is enabled by an appropriate data discard strategy, the use of local approximation techniques, and an ongoing trade-off between exploration and exploitation. Furthermore, to our best knowledge, this work provides the first Python-based ML framework integrated in an industrial control for safe process optimization. Based on our novel concept, an application is implemented and installed on a state-of-the-art control unit. This device is used for intelligent process optimization for the first time. In addition, we aim for a generic design, which makes it possible to apply the concept for several industrial machines with different optimization goals. We validate this by presenting a real-world application enabling self-optimizing saw blade straightening machines all over the world. All these contributions are consistent with the target vision of self-optimizing machines in smart factories.

Background and methods

First, the mathematical problem to be solved with safe contextual optimization is defined. Next, this section covers the fundamentals of Gaussian process regression (GPR), which describes the distribution of random variables over space in our case and can be used for Bayesian optimization. In contrast to other methods, Bayesian methods, due to their probabilistic nature, have the advantage of providing not only the prediction but also an estimate of uncertainty. More precisely, this estimate can be exploited for state-of-the-art safe optimization methods, which are introduced in this section as well.

Problem statement

The goal of industrial self-optimization is to find the optimal parameterization for the current situation leading to the best possible process quality. This optimization is constrained to fulfill the requirements: on the one hand, the parameter range is restricted, and on the other hand, the re-parameterization must not lead to prohibited conditions. In the following, we call acting in compliance with these constraints *safe* acting. In the following, an unknown process quality function $f(x)$ should be optimized, which can be controlled via the parameterization $x \in \mathcal{X}^{D_\text{x}}$. A minimum tolerable process quality f_min should always be ensured, leading to the following formulation for M optimization iterations:

$$\max_{x \in \mathcal{X}^{D_\text{x}}} f(x) \text{ s.t. } f(x_i) \geq f_\text{min} \; \forall i = 1, 2, .., M. \tag{1}$$

Since $f(x)$ is unknown, we have to assume a non-linear function. A further assumption is that the objective function has to be a member of a reproducing kernel Hilbert space within the limitations of the optimization space. By having bounded norm in this space, the smoothness to the kernel can be measured so that Gaussian process regression can be applied (Turchetta et al., 2019). Moreover, Lipschitz continuity is assumed for the objective function, so that further information can be obtained by appropriately careful exploration based on previous experiments. Without this assumption, even small steps caused by sensor noise could lead to large deviations, making safe exploration impossible.

Equation (1) presupposes that a problem always occurs under the same conditions. More realistically, one experiment is similar to another only to a limited extent because, e.g., manufacturing dimensions or environmental variables such as humidity or temperature vary. These variables cannot be directly controlled by the system and can be interpreted as environmental prerequisites for the next experiment. The environment is numerically described by a context $z \in \mathcal{Z}^{D_\text{z}}$, leading to a more complex optimization problem formulation:

$$\max_{x \in \mathcal{X}^{D_\text{x}}} f(x, z_i) \text{ s.t. } f(x_i, z_i) \geq f_\text{min} \; \forall i = 1, 2, .., M. \tag{2}$$

In the special case where the context remains the same over M iterations, (2) is equivalent to (1). However, in real applications, it is unlikely if not impossible to perform exactly the same experiment under the same conditions, such as in foundry manufacturing processes. Mathematically, the goal of this work is to solve the problem (2) in high-dimensional spaces to enable self-learning machines in industry.

Gaussian process regression

First, a regression model is required to approximate the system behavior after each optimization iteration. By experimentally sampling points from an unknown function $f(x)$, one obtains a set $f(\mathbf{x}) = [f(x_1), .., f(x_N)]$ with a finite number N. In Bayesian theory, this set is assumed to be drawn from a jointly Gaussian distribution, which can be determined with a sufficiently large N (Rasmussen & Williams, 2006):

$$f(\mathbf{x}) \sim \mathcal{N}(m(\mathbf{x}), \Sigma(\mathbf{x}, \mathbf{x}))\,. \tag{3}$$

This joint distribution is a distribution of Gaussian process (GP) and can be used to regress $f(x)$ to predict a function output for arbitrary x. Such a GP is obtained by multiple generated sample functions, each fitting the sampling points $\mathbf{x}$. However, real-world systems lead to rather noisy measurement points, so-called observations $\mathbf{y} = f(\mathbf{x}) + \mathcal{N}(0, \sigma_{\text{noise}})$. In contrast to other regression methods, a GP returns no single prediction values $f^*(x)$, but a mean and variance of the Gaussian normal distribution for each x:

$$f^*(x) \sim \mathcal{GP}(m(x), k(x, x')), \tag{4}$$

with the average value over the generated sample functions $m(x)$ and the kernel or covariance function $k(x, x')$. In the following, radial basis function (RBF) kernels will be used to regress the system behavior. They can be characterized as universal kernels (Micchelli et al., 2006), being able to model all continuous functions under conditions with optimal hyper-parameters.

Bayesian optimization

Based on the regression of $f(x)$, the most promising parameterization for the next iteration should be selected aiming for its optimum following (1). Bayesian optimization (Mockus, 2012; Shahriari et al., 2015) methods determinate such query points to optimize an unknown objective function by iterative sampling from it. After each iteration, the GP regression is updated and provides a prediction for variance $\sigma^2(x)$ and mean value $\mu(x)$. By this prediction, a so-called acquisition function, e.g. upper confidence bound (UCB), indicates how informative a sampling of the objective function at an arbitrary x would be. The next query points can be determined by maximizing the acquisition function:

$$u_i(x) = \mu_i(x) + \beta\sigma_i(x), \tag{5}$$

$$x_{i+1} = \underset{x \in \mathcal{X}^{D_\mathrm{x}}}{\arg\max}\, u_i(x). \tag{6}$$

for iteration i. The parameter β scales the confidence interval, while larger values increase the importance of exploration. To handle the restriction of (1), the parameter gets an additional meaning, which is explained in the following.

Safe Bayesian optimization

For industrial usage, an online optimization is required to fulfill given standards to ensure pre-defined safety restrictions. Here, safe Bayesian optimization (Sui et al., 2015) aims to find the optimum of an unknown objective function while guaranteeing this fulfillment based on the regression model. Commonly, the restriction is represented by a threshold $f_{\min}$, which limits the optimization space $\mathcal{X}^{D_\mathrm{x}}$ to a so-called safe set $\mathcal{S} = \{x \in \mathcal{X}^{D_\mathrm{x}} | f(x) \geq f_{\min}\}$. In this way, safe optimization is only partly global, as the global optimum might not be reachable depending on the initial observation. Since the objective function $f(x)$ is unknown, the true safe set $\mathcal{S}$ can only be estimated, for example by using the lower confidence interval (Berkenkamp et al., 2016),

$$l_i(x) = \mu_i(x) - \beta\sigma_i(x), \tag{7}$$

$$\hat{\mathcal{S}}_i = \{x \in \mathcal{X}^{D_\mathrm{x}} | l_i(x) \geq f_{\min}\}. \tag{8}$$

SafeUCB is known to be greedy and less exploration-driven. To improve the exploration-exploitation balancing, SafeOpt (Berkenkamp et al., 2016) distinguishes between promising maximizers $\mathcal{M}_i$ (points with increased probability of being the global maximum) and expanders $\mathcal{E}_i$ (points with increased probability to expand the safe set):

$$\mathcal{M}_i = \{x \in \hat{\mathcal{S}}_i | u_i(x) \geq \max_{x' \in \mathcal{X}^{D_\mathrm{x}}} l_i(x')\}, \tag{9}$$

$$\mathcal{E}_i = \{x \in \hat{\mathcal{S}}_i | l_i(x) < f_{\min} + \gamma\}, \tag{10}$$

where parameter γ is a small positive value. To ensure a trade-off between finding the maximal objective value and minimizing the uncertainty, SafeOpt selects within the union of both calculated sets:

$$x_{i+1} = \underset{x \in \mathcal{E}_i \cup \mathcal{M}_i}{\arg\max}\, (u_i(x) - l_i(x)). \tag{11}$$

Multi-pronged safe Bayesian optimization

To solve (6) for higher D_{x}, a sampling in grid-discretization cannot be fulfilled in reasonable computing time for firm real-time applications on edge devices. A subset of $\mathcal{X}^{D_{\mathrm{x}}}$ has to be selected, which in the best case does not lead to any disadvantage compared to full sampling. Therefore, SafeMixedBO (De Blasi et al., 2021) combines a random line sub-space with origin at the current best observation for exploitation and an ellipsoidal sub-space at random origin for exploration. Based on this sampling sub-space combination and the adapted idea of StageOpt (Sui et al., 2018), which separates exploration and exploitation phase, a continual trade-off between exploration and exploitation in high-dimensional space can be enabled with the correct setup. This is required, because SafeOpt is overly exploration-driven, whereas SafeUCB is excessively exploitation-driven (Sui et al., 2018). StageOpt proposes a condition for the transition from exploration to exploitation. The transition condition of StageOpt can be a fixed and user-defined time step, or whenever the expander uncertainty is below an user-defined threshold ϵ. In the first stage, only the set of potential expanders $\mathcal{E}$ is calculated and used for acquisition. After the transition, a greedy method like SafeUCB is used for the entire safe region of the exploration stage. Although it is unlikely, further exploration of the safe region may occur during the exploitation phase. We present our detailed setup for contextual application in the self-adaptive balancing part of the next chapter.

Safe contextual Bayesian optimization

In the conventional form of Bayesian optimization, it is assumed that all variables are adjustable for the upcoming experiment. However, this is not the case in many real applications due to external influences. These variables are referred to as context, which include, for example, environmental variables. Accordingly, standard Bayesian optimization can be redefined as solving a problem with a constant context. Since the context has an influence on the objective, one can define an infinite number of sub-problems and optimize each of them separately. However, this is generally infeasible, especially with continuous context dimensions. Assuming that two similar contexts have more in common than a more distant context, the context can be included in the GP regression as additional dimensions. In this way, an external variable is considered as a fixed coordinate for the next experiment, which is given for each iteration and cannot be optimized, but affects the objective. This extension of Bayesian optimization to contextual Bayesian optimization (Krause & Ong, 2011) allows optimization of complex systems with environmental context. By multiplying the kernel functions of the context and the action space, we assume that an objective is dependent on these variables and there are no other variable influences that change the system behavior:

$$k(x, z, x', z') = k(x, x') \cdot k(z, z'). \tag{12}$$

In this way, whenever one context is very close to former context, the predicted $\mu_i(x, z_i)$ and $\sigma_i(x, z_i)$ are close to this observation and lead to a similar acquisition function.

The contextual Bayesian optimization can also perform a safe optimization through the appropriate adaptation of the GPR and the acquisition sampling (Fiducioso et al., 2019; De Blasi, 2019; Berkenkamp et al., 2021). For example, the SafeUCB contextual optimization determines its most promising candidate in the following way:

$$x_{i+1} = \underset{x \in \hat{\mathcal{S}}_i}{\arg\max}\, u_i(x, z_i). \tag{13}$$

This extension of contextual Bayesian optimization to include safe optimization concepts leads to corresponding changes with respect to the search space, since it may differ at each iteration due to the given context. This makes (2) theoretically solvable. However, it proves to be difficult to apply in practice, since the contextual changes from the next experiment to the previous ones would have to be minor so that the estimated safe set $\hat{\mathcal{S}}_i$ can always determine at least one safe candidate. As soon as a context is present that is significantly different from those observed so far, the estimation will provide $\hat{\mathcal{S}}_i = \emptyset$ and safe optimization will be interrupted. To circumvent this phenomenon, the optimization is trained in advance or regulated on the basis of predefined rules.

Default policy

Commonly, a safe initial set is provided to enable non-contextual safe optimization. In contrast to contextual optimization, this requirement can be fulfilled in the simplest case by a single prior experiment. However, if an arbitrary context is set for each experiment, at least one safe parameterization must be possible for this context. In this sequence, at least a certain number of experiments would have to be performed in advance, depending on the GP hyper-parameters and the limits of the system. This is on the one hand costly for the operator and on the other hand contrary to the vision of self-learning systems, but it is also applied in some studies (Berkenkamp et al., 2021). A more elegant way, which is in harmony with the automation, is the setup of a default policy $h(z_i)$ providing a parameterization depending on the context leading to acceptable process performance (De Blasi, 2019; Fiducioso et al., 2019). Even if this initially sounds like a strong assumption, it is in fact frequently encountered in everyday industrial practice anyway (Jagerman et al., 2020). Mostly, process parameters are set in a rule-based fashion or even rigidly, so that minimum standards are met, but without exploiting the full potential in terms of process quality. Accordingly, in terms of safe optimization, approaches have been proposed that use a safe standard policy instead of just prior data:

$$x_{i+1} = \begin{cases} h(z_i) & \text{if } \hat{\mathcal{S}}_i = \emptyset \\ \text{according to selected acquisition function} & \text{else.} \end{cases} \tag{14}$$

For commissioning, for example, the machines are delivered with standard parameter sets that “roughly fit” according to experience and the machine design.

Self-adapting safe Bayesian optimization

The presented safe Bayesian optimization approaches require an appropriate hyper-parameter setup of GPR. If this parameterization is insufficient, safe optimization is inefficient too or even unsafe. In order to reduce the effort involved and the demands on the prior understanding of the problem, self-adapting safe Bayesian optimization (SASBO) (De Blasi & Gepperth, 2020) iteratively scales the observations and uses a constrained optimization of the hyper-parameters. In this way, it requires easy-to-choose initial hyper-parameters. SASBO is less data-efficient than a comparable optimization with the ideal hyper-parameters because of the required hyper-parameter optimization. Due to the industrial application area, we will use SASBO with the multi-pronged approach based on the existing default policy. In this way, our approach is easy to apply and no effort is needed until the actual optimization should affect the process.

Efficient safe contextual Bayesian optimization

Based on the methodology of the last chapter, (2) can theoretically be solved for applications in industry. However, the computational effort is in practice a problem leading to a strong limitation of possible applications. Therefore, in the following, the safe contextual Bayesian optimization is enabled for continual optimization, which is achieved by improving the computational performance of the prediction. Therefore, we first present iterative local approximation and combine it with sparse GP regression based on a presented forgetting strategy. In addition, the balance between exploration and exploitation is addressed by adopting an adaptive approach to iteratively decide whether further exploration might be appropriate in the given context.

As SASBO is applied, the interactive normalization of the observations lead to a fixation of $\beta = 3$ and $\gamma = 0.1$. It should be emphasized that some required assumptions for the application of the approximation methods are only possible under this normalization.

Iterative local Gaussian process regression

The computational effort of full GPR can restrict the application field of complex problems with high-dimensional action space based on the amount of included data points. This is especially true for contextual optimization, since it contains a large number of related problems and not just a single one. Thus, if the number of points required for a GPR iteration can be reduced and the optimization is nevertheless safe, the field of application becomes less limited. This is the motivation behind the following series of thoughts. Theses statements are only true for GPR using a non-periodic and non-linear kernel function. While, in a grid-search optimization, all points can carry relevant information for the upcoming iteration step, this is not the case for sub-space methods. Since the relevance of a point for a certain prediction depends on its distance, one can say that points above a certain distance are irrelevant for this prediction if significantly closer points are present. Whenever the total number of data points exceeds n_i, we iteratively define a local neighborhood with distance 3ℓ around the prediction sub-space. For GP regression with RBF kernel, this approximation can be justified for a safe optimization by the fact that any safety-critical deviation can be converted into a deviation that is affecting performance only (see appendix for proof). However, this holds only when iterative normalization (De Blasi & Gepperth, 2020) is applied. The resulting prediction similarity of full and local Gaussian process (LGP) regression is illustrated in Fig. 2. In this example, only around 2% of the global data are used for the local GPR, leading to practically identical predictions of the current context.

The origin of the sub-space definition for sampling in high-dimensional space is chosen iteratively close to the current context. For this purpose, the distance from the context is increased in steps of 0.1ℓ until a sufficient number of points is contained in a hypersphere around this context. Then, the best observation is chosen as origin for the exploitation sub-space and a random observation as origin for the exploration sub-space. In Fig. 3, the local neighborhoods of the two used sub-space definitions are illustrated separately. A larger sub-space definition leads to a larger neighborhood and, accordingly, a more complex GPR, in extreme cases to the full GPR.

We use the iterative local GPR to reduce the amount of data points per iteration. However, for continual application the amount of points is required to be limited, which is addressed in the following.

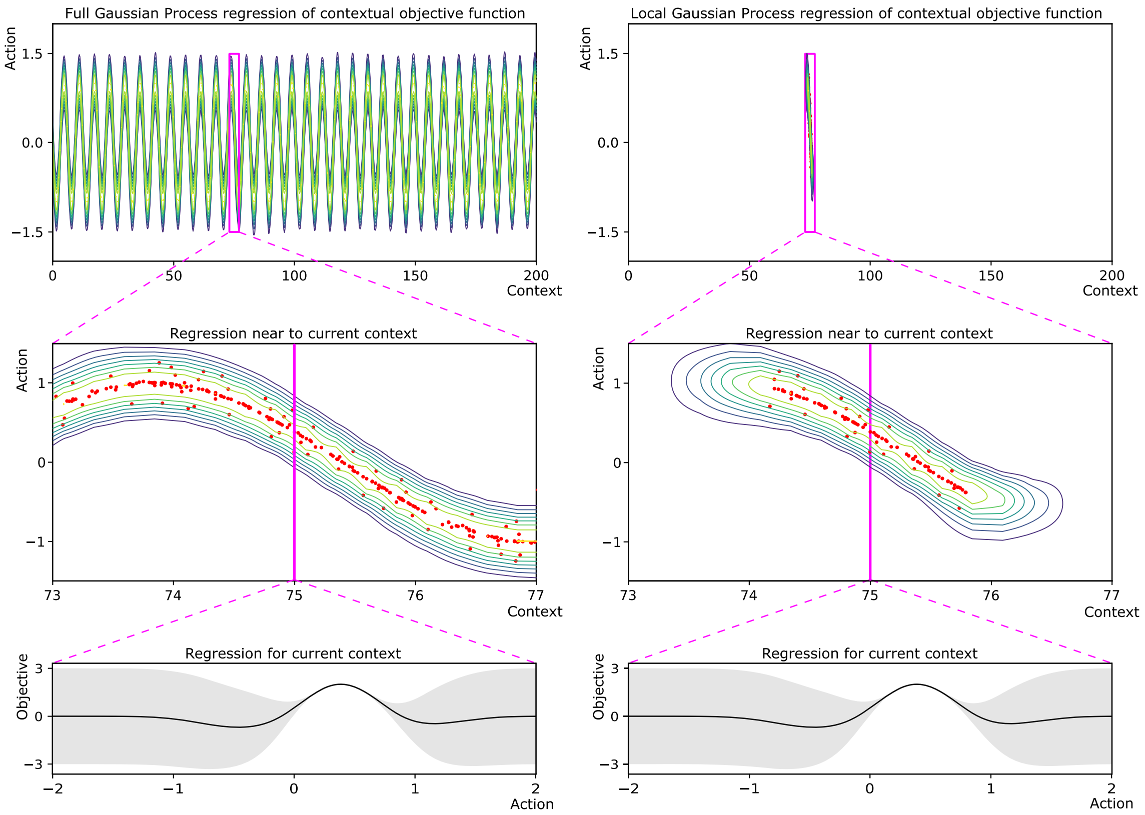

Fig. 2 Exemplary GP regression for a specific context: On the left is the full GP regression and on the right is the iterative local GP regression relevant to the current context of interest. The relevant areas are zoomed in vertically. For the actual prediction, see lower plots, practically no difference is visible

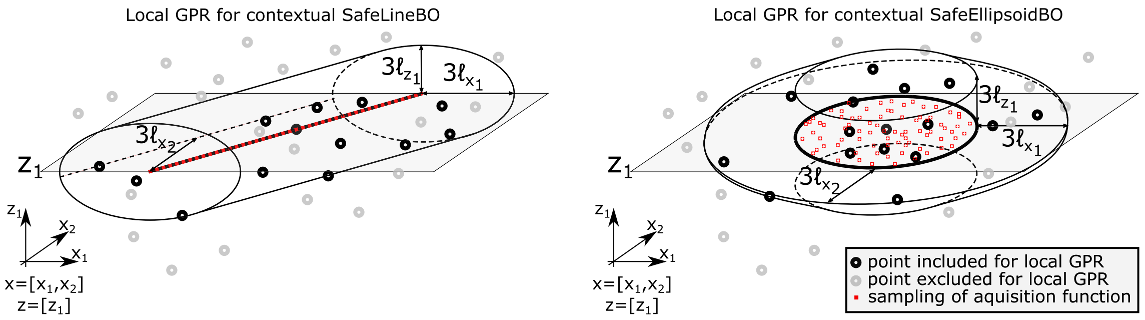
Local GPR for contextual SafeLineBO
Local GPR for contextual SafeEllipsoidBO
$3\ell_{z_1}$
$3\ell_{x_1}$
$3\ell_{x_2}$
Z_1
z_1
x_2
x_1
x=[x$_1$,x$_2$]
z=[z$_1$]
point included for local GPR
point excluded for local GPR
sampling of aquisition function

Fig. 3 Exemplary local neighborhood definition for GP regression: A two dimensional action space is sampled with a different sub-spaces on the plane defined by the current context z_1. On the left side, the line sub-space leads to a cylindrical area whose inner points are used for the local regression to predict the candidate points (red). The ellipsoidal sub-space on the right side leads to a more complex shape for the determination of the local GP points

Sparse Gaussian process regression

In order to keep the computational effort nearly constant, we want to keep the number of points for the GPR constant. Methods involving inducing points are unsuitable for safe optimization, as the safety estimate can deviate to an unacceptable extent. Therefore, the iterative discarding of a point, which is replaced by a new point with a certain similarity to the discarded information, is the best class of sparse GPs for safe optimization. In this way, the sparseness leads to an overly cautious estimate of safety rather than a risky one. The constant number of points is called the *budget*. Here, we connect the budget to the iterative local neighborhood and iteratively discard the less informative points within this neighborhood until the budget is met. After this theoretical insight, the question arises which method is suitable for discarding data without endangering safe optimization.

Forgetting of less informative data

To discard less informative points, there are several methods with different levels of complexity. The easiest method is discarding random points, or the oldest points. While these approaches might work well for large amounts of points, they are unsuitable for a rather small amount of points, as the case considered here. In detail, discarding the oldest observation as well as randomly discarding it is not goal-directed, since at least one observation in each context should gradually lead to improvement or be retained. It is therefore more reasonable to use the Kullback–Leibler divergence (Seeger et al., 2003) or entropy (Herbrich et al., 2003) for the comparison of the information content of points. However, the required computational effort excludes this due to the imposed near-real-time requirements. In contrast, the normalized distance (normalized by ℓ of the GPR kernel) to all points in the local neighborhood can be computed very efficiently. To combine the advantages, we propose to first determine the two closest points, and then to calculate the entropy of these points to discard the less informative point.

The information along the context dimensions can be more important and should be weighted accordingly. If the budget is chosen sufficiently large, such a weighting is nor required. In the worst case, an insufficient budget can lead to the loss of the entire safe set for a certain context, which accordingly requires the reuse of the default policy.

Self-adaptive balancing

For continual optimization, a distinction must be made between exploration and exploitation in order to achieve the optimum in the long term. As the time step transition condition of StageOpt is not appropriate for contextual optimization, the selection of ϵ is required. This parameter is not intuitive to choose for an unknown objective with unknown scaling. However, since we scale iteratively, we can calculate ϵ_i based on the previously estimated uncertainty of the selected expanders or maximizers and the current neighborhood:

$$v_i = u_i(x_{i+1}) - l_i(x_{i+1}), \tag{15}$$

$$\epsilon_i = \begin{cases} 0 & \text{if size}(j) \leq J \\ \mu_{\mathbf{v}[j]} + 3\sigma_{\mathbf{v}[j]} & \text{else if } \sigma_{\mathbf{v}[j]} \leq 0.2 \\ Q_{\mathbf{v}[j]}(0.4) & \text{else.} \end{cases} \tag{16}$$

Here, j is a vector of indices indicating which points are within the current neighborhood and J is the amount of minimum exploration experiments per neighborhood (we chose 20% of the budget). First, a good initial exploration is ensured by $\epsilon_{1..J} = 0$ near to each context. Afterwards, the growing $\mathbf{v}$ with elements v_i is used to check whether the uncertainties near the current context are stable or unstable. If they are stable, we compute an ϵ_i that is significantly larger than the previous candidate uncertainties. In this way, we prefer expanders to maximizers only when the exploration is likely to be very informative. If the uncertainties are unstable near the current context, ϵ_i is computed as the quantile of the uncertainty vector. Too large values for the quantile lead to being greedy, while too small values prioritize exploration. We found that a quantile of 40 percent is a good compromise. Similar to StageOpt, points are considered only if their estimated uncertainty is above the threshold ϵ_i and thus indicates a potentially informative observation. Whenever the candidate uncertainty is too low, SafeUCB (13) is applied to determine the next candidate:

$$x_{i+1} = \begin{cases} \text{acc. to (11)} & \text{if } \max_{x \in \mathcal{E}_i \cup \mathcal{M}_i} (u_i(x) - l_i(x)) \geq \epsilon_i \\ \text{acc. to (13)} & \text{else.} \end{cases} \tag{17}$$

In contrast to the original idea of StageOpt, we allow a return to the exploration phase, which is essentially necessary because of the contextual setup of the optimization problem.

Implementation for self-learning machines

Since industrial control technology is highly regulated, a software module like an ML framework cannot be straightforwardly integrated into traditional control concepts. Therefore, these barriers shall be addressed in our work by a novel control concept and the development of a software module suitable for the purpose of self-learning machines.

Industrial control platform

As we aim to provide a generic tool for a wide range of applications, it is important that the modules can run on a stand-alone control unit, or via high-performance communication together with another network device. Bosch Rexroth's ctrlX CORE provides many of the capabilities for simplified industrial ML (De Blasi & Engels, 2020) and, in particular, provides an open platform with a shared memory concept for inter-process communication that allows different software modules to fully interact with each other. In the following, we present this edge device as the used ecosystem of our work. The data-driven decisions of a learning framework can be provided on-demand via a message broker service (Albrecht et al., 2019), enabling modules to subscribe to or provide information. The access path is stored within the control unit, which can be accessed by previously allowed network devices (Albrecht et al., 2019). By addressing the same node, the modules can obtain the access path,

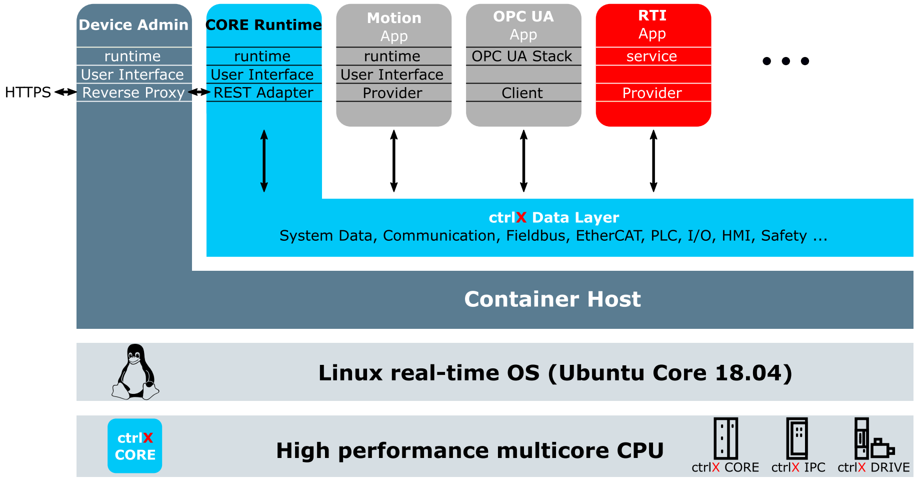

Fig. 4 The ctrlX CORE architecture is open and enables our RTI app (red highlighted) to provide a machine learning framework interaction with the system

which could even point to data on a different device. The used single board computer runs with an ultra-lightweight operating system (Ubuntu Core), enabling the container principle by using the package manager Snap from Canonical. Thus, installed programs run in isolated and immutable environments. The underlying system and other programs can only interact via the provided interfaces. The respective modules can be updated without the risk of affecting the stability of the system. A process scheduling allows the prioritization of different tasks. For example, the real-time automation program is prioritized and cannot be interrupted by the optional software. Furthermore, it runs on isolated CPU cores and is specially secured. Since the control unit can host a web server, the interface can be accessed with a standard web browser without installing additional software. In Fig. 4 the used software architecture is illustrated providing information about the communication.

Real-time intelligence snap

To feed the hard real-time system with intelligent decisions, interactions between the ML framework and the process data are necessary. In contrast to usual communication architectures (Schwung et al., 2017, 2019; Jaensch et al., 2019; Schmidl et al., 2020), which run the ML framework and the process control on two different devices due to the inflexible control unit, we can run both on a single device. This increases data security, communication performance and application variability, for instance. Following the container principle, a snap was developed in the course of this project, which is referred to as Real-Time Intelligence (RTI) snap. We implemented the presented algorithm based on GPy (2012) within a Python-based framework. Furthermore, the RTI snap provides the used shared memory for configuration and data exchange over a lightweight C++ application as provider. Thus, the learning framework can be adjusted by each software component or simply over the web interface. This not only allows the connection to user interfaces of existing plants, but also enables the learning of different models for several use cases on the same machine. For the data exchange, we chose the binary buffer FlatBuffer (Google, 2014) over the JSON data format because FlatBuffers are superior regarding encoding/decoding time and memory usage. The nested objects within FlatBuffer are predefined via a schema. Thus, the predefined offsets reduce the time to access data with no parsing required. Whenever the data exchange should be possible for a human to be easily interpreted, the schema has to be adjusted for each use case. Three data exchange nodes are required to enable closed-loop learning. Before each experiment, a context is sent from any application. The RTI snap reacts on-demand via the subscribe functionality on a new context at the context node by running the learning iteration, building an action FlatBuffer based on the context and sending the extended FlatBuffer to the action node. Here, the application reads the decision and can run the desired experiment. Afterwards, the required data is recorded to calculate the performance of the experiment. The regarding action FlatBuffer is extended by this data and send to the result node. This node is in turn subscribed by the RTI snap and used as learning feedback. Accordingly, an applicator can also use the result node to train the model by historical data or manually executed experiments. To avoid multiple processing, the scheme includes an ID for each experiment. In this way the RTI snap interacts with any snap to include an intelligent decision if desired in firm real-time. During our project, we let the RTI snap communicate with the real-time operating system for process control based on Codesys, see Fig. 5.

Interaction modes

To support a wide variety of projects, several interaction modes are provided.

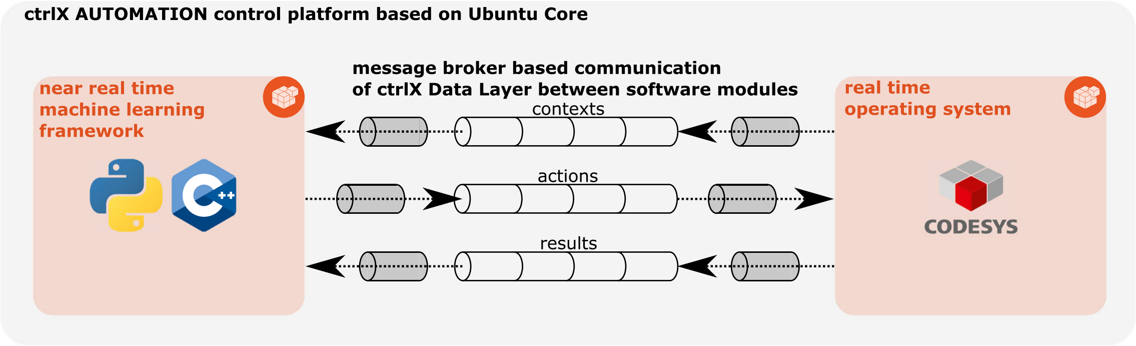
ctrlX AUTOMATION control platform based on Ubuntu Core
near real time machine learning framework
message broker based communication of ctrlX Data Layer between software modules
contexts
actions
results
real time operating system
CODESYS

Fig. 5 Interprocess communication between the process control and the near real-time machine learning framework on the ctrlX CORE. By using a message queue, the concept is flexibly applicable to, for example, a command buffer

- *maxaction / minaction*: This mode aims to optimize weighted action parameters while ensuring the safety criterion. This can be any combination, for example, that the first parameter should be maximized, the second minimized and the remaining parameters can be chosen arbitrarily.
- *exploit*: This mode runs UCB for objective maximization problems, which can be especially helpful in situations when a good result has already been found, but final fine-tuning is desired.
- *explore*: This mode selects amount the identified expanders to increase system understanding. In this way the interpretability of our presented approach can be valuable for industrial applications (Liu et al., 2022). It is not to be understood as an optimization mode.
- *learn*: This mode runs adaptive StageOpt as presented in this paper. By trading off between exploration and exploitation this mode can be used as default selection for objective maximization problems.
- *perform*: This mode selects the maximum of the LCB for objective maximization problems leading to a minimized risk of weakly performing the experiments. When stable process quality is highest priority, this mode is recommended after optimization for daily usage.

Evaluation and application

To the authors' knowledge, the presented approach is the first continual safe contextual optimization method and therefore cannot be compared to other methods, as they would not meet the requirements in at least one aspect (continual usable, context involving, safe). As real-world applications include natural perturbations appearance regarding sensor signals and actor command results, the approach is required to be robust. This can only be ensured if the GPR hyper-parameters are chosen correctly. For each application, the respective hyper-parameters of the GPR were determined by applying SASBO (De Blasi & Gepperth, 2020). Furthermore, it is recommended that the affecting safety threshold is not defined as the hard threshold but is set as a theoretical threshold slightly above the the actual desired threshold. To achieve a firm real-time capability for the real-world application, SafeMixedBO (De Blasi et al., 2021) with $D = 3$ is applied as the underlying algorithm for the efficient safe contextual Bayesian optimization approach.

Synthetic evaluation

For evaluation, the approach is applied to five test problem functions, ten times each. These suitable test problems for contextual safe optimization were adapted from well-known test functions (Momin & Yang, 2013) and extended by a contextual transformation, which in our case is a sinusoidal oscillating addition. For details regarding the test functions, see appendix. For each iteration, the context is randomly drawn from an uniform distribution with values larger than the corresponding length scale parameters. The used budget was chosen $\leq$ 1000 depending on the complexity of test problem. The evaluation metrics are regret-based and are adapted to contextual problems. The cumulative regret indicates the exploitation:

$$R_i = \sum_{j=1}^{i} f(x^*, z_i) - f(x_i, z_i). \tag{18}$$

Here, a horizontal asymptotic behaviour would mean an optimal exploitation for each context. Furthermore, the simple regret indicates the currently best solution:

$$\hat{x}_i = \arg\max_{x \in \hat{S}_i}(l_i(x, z_i)), \tag{19}$$

$$r_i = f(x^*, z_i) - f(\hat{x}_i, z_i). \tag{20}$$

The optimization is likely to select $x_i \neq \hat{x}_i$ during exploration.

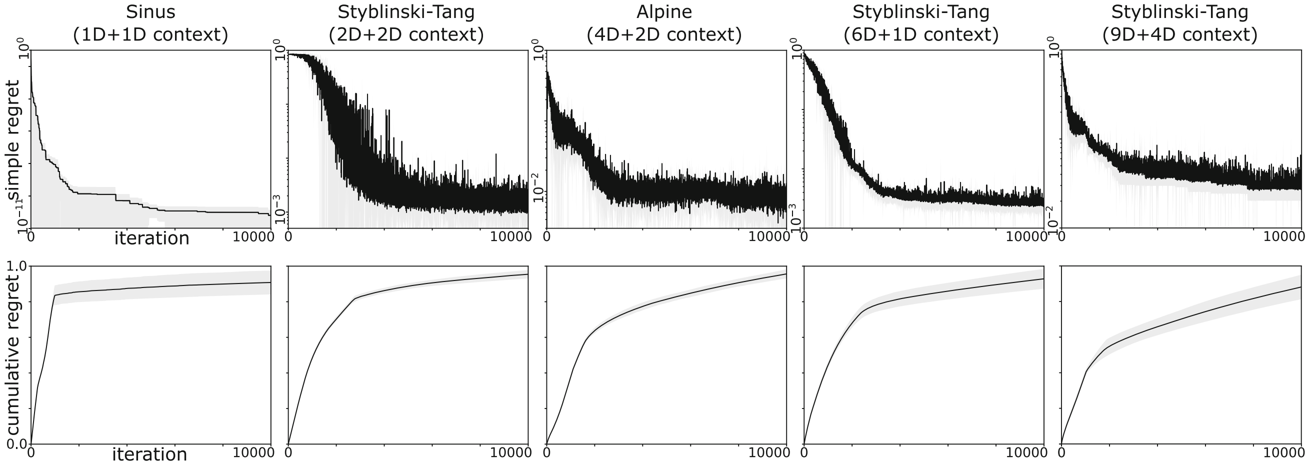

Fig. 6 Normalized regrets of synthetic evaluation test problems for contextual safe optimization

Since the budget for a high-dimensional context with wide range should be accordingly large, and this strongly affects the computation time, we decided to use a small range of only 166.7% of the corresponding length scale for the four-dimensional context of the 9-dimensional test function. In this particular case, the local GP approach equals a full GP approach because the same neighborhood is selected and a total of 9000 points are discarded. For all other tests, the global models contain between 2239 and 5244 points after 10,000 iterations. While the discarding accelerates the calculation and enables continual online learning, it does not lead to any violation of the safety restriction. The results are illustrated in Fig. 6. Despite the reduction of the context range for the highest-dimensional test function, the chosen budget is still too small for solving this complexity with changing context. Although no horizontal asymptotic behavior of the cumulative regrets is achieved, in all cases an oblique asymptotic behavior with low slope can be found. This means that the optimum is not permanently reached for every context, but that the test functions were solved adequately on average. At the time the respective budget is reached and points are started to be discarded, the subsequent optimum has not been reached in any of the test cases. This confirms that exploration-exploitation balancing for long-term optimization is achieved despite discarding points.

Industrial application

Industrial applications assign a higher compliance priority to certain aspects before deployment is appropriate. First, optimization should not prevent a system from changing or even being shut down, but reverse causality is possible (Ribeiro et al., 2016). Furthermore, it is important to ensure correct operation of the system, even if, for example, decision making leads to computational problems due to overload (Amodei et al., 2016). We use a default policy, which is applied whenever decision making took too long. As a result, learning efficiency suffers in the worst case, but operation is not negatively affected by turning on the learning procedure, compared to conventional operation. Third, a misleading definition of learning feedback is a threat to industrial learning, as inadequate definitions can lead to supposedly good results numerically because the model found an undesirable way to determine an optimum of the feedback without satisfactorily solving the actual problem (Ribeiro et al., 2016). To minimize this risk, interviews were conducted with domain experts and several example cases (including rather unrealistic cases) were analyzed to derive a cost function which received full approval by the experts.

As an application example, we enable self-optimization of the straightening machine HAMMERHEAD 3000 by the manufacturer Kohlbacher GmbH for saw blades which are delivered all over the world, as shown in Fig. 1. The general process of the straightening machine is explained on https://youtu.be/LiaJlKzYnxw. Accordingly, the working conditions vary strongly. A saw blade must be in shape and have the right tension so that it can cut wood efficiently and with as little wear as possible. The straightening machine aims to perform this preparation automatically (exclusive of the sharpening step). As the process of working the steel is affected by conditions like steel thickness, target shape or environmental temperature, a domain expert is usually required to tune the optimal parameters. Here, four parameters are particularly difficult to choose: The upper and lower limits for the machining pressure with respect to the back and the general tensioning process. In addition to commissioning, which can take several days, this adjustment may become necessary again and again even years later when, for example, the steel supplier is changed. The target vision is for the user to be able to insert a saw blade, which will then be worked on automatically. While the first virgin machining takes a long time, but does not destroy the saw blade, the nec-

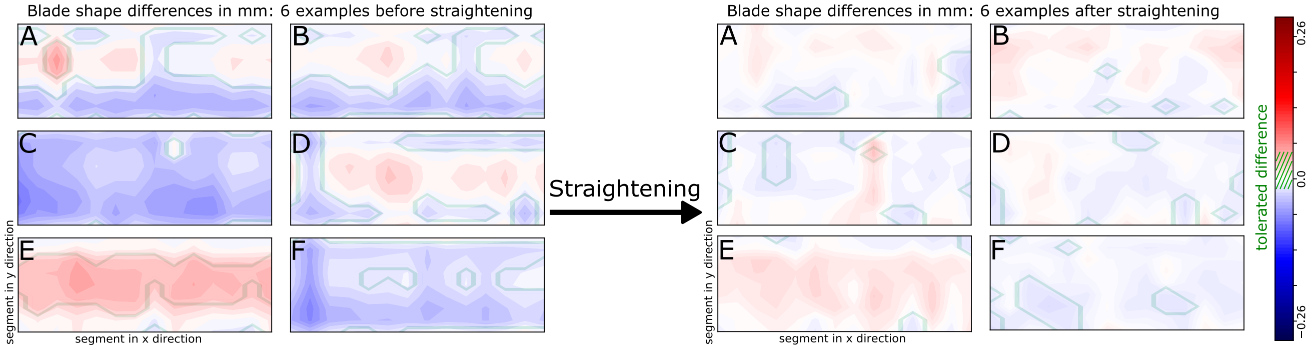

Fig. 7 The exemplary blade shape differences before and after the straightening process illustrate the achievements during the experiments. The initial states of the blades, which can be found on the left side, have larger under- (blue) and also sometimes over-tension (red). After the machining process, the majority of the blade areas is within the tolerance range and the overall target quality for each blade has been achieved

essary machining cycles are to be continuously minimized. In this way, the system adapts to the respective environment. In the end, the system has optimized itself and no domain expert commissioning is necessary. Furthermore, a separate model can be trained for each different type of saw blade.

Problem statement and setup

The optimization goal is to straighten the saw blades with the HAMMERHEAD 3000 as fast as possible without damaging the blades by choosing the best combination of the four mentioned process parameters. This problem is to be solved with the presented approach as an industrial application in the course of the presented architecture. As a standard policy, a parameterization is chosen in consultation, which has hardly any effect even on thin steel. The safety threshold is defined by shapes of blade that can no longer be processed by the system in a targeted manner. Before and after each processing cycle, we measure at 14 up to 15 segments and 9 up to 11 lanes (depending on the size of the blade). For each segment, the mean value of the difference between the target shape and the current shape is determined and serves as one input variable. The second input variable for the model is the measured difference on the back of the blades. A whole work cycle consists of three rounds in which each lane is pressed with a calculated force (based on the difference and the model power) to form the blade correctly. These cycles take between 15 and 25 min depending on the required intermediate steps. After each cycle, the process performance is calculated from the performed measurements and provided to the model which chooses the action values for iterative learning.

A saw blade counts as completed as soon as the back tension as well as the blade tension have less than a 15% deviation to the target shape. This can be complicated by the fact that the processing of one criterion impacts the others, often leading to ambiguous behaviour. For example, in the case of an over-tensioning blade, the back is shortened by removing the tension with pressing on the according point. This might lead to a shorter back. If the back then has to be lengthened again by the corresponding pressure points, this increases the tension.

Experiments and results

To prove the generality, different saw types are used for evaluation. In Fig. 7, six examples of blade shapes before and after straightening are compared. The differences between the blade shape before the machining can vary very strongly. While Fig. 7A and B are very similar, for example, 7C has under-tension and Fig. 7E has strong over-tension. To take this into account, we define a two-dimensional context where one dimension summarizes the current overall tension, and the second dimension is the measured back tension of the saw blade.

We evaluated the implementation by straightening three blade types with different thicknesses and lengths. The results are summarized in Table 1. The improvement between the first blade to the second blade is remarkable. Especially with the type 6200x100x1.0, it is very pronounced due to the similarity of these two blades (Fig. 7A and B). Unfortunately, no other blades of this type were available. For the other types, it can be seen how a very different context has an effect and thus restricts the learning transfer between processed saw

Table 1 Required straightening cycles during learning until done criteria reached

Type of blade	Blade number							
	1st	2nd	3rd	4th	5th	6th	7th	8th
6160 × 100 × 0.9	23	10	13	11	7	4	4	4
6200 × 100 × 1.0	31	2						
6500 × 120 × 1.2	45	6	13	8	11	10	2	2

blades. However, a process optimization was achieved in all of the three blade types, as the model was able to complete them within a few cycles. We hypothesize that the blade type 6160x100x0.9 is harder to optimize, due to the thin steel, and that the parameterization to deal with the ambivalent behavior is more sensitive. A correlation between blade thickness and the required processing cycles for the initial blade can be identified. This is logically justifiable, since the same default policy is applied for all cases, and thicker blades require higher pressure parameters for the desired straightening. For the field application, it is therefore recommended to query the blade thickness and to adjust the default policy accordingly. This will lead to an even quicker learning progress.

All in all, the system was able to optimize itself and learn different models for different types of saw blades without incurring material costs, even without a cloud connection. This eliminates the need for domain experts to travel to the site and the associated time delays in production, and availability is practically on-demand.

To validate the generality in different application domains, the snap was also applied to another machine for optimizing industrial processes. Here, the acceleration and deceleration of a large-scale Cartesian robot is safely optimized over several days to minimize its pick-and-place time without slipping or sliding. For this application, the *maxaction* interaction mode of the snap was used.

Conclusion

As novel control architectures enable the use of learning frameworks close to the process control, the logical next step is intelligent process optimization. In line with the target vision of industry 4.0, we contribute to the desired self-optimization machines by several achievements. Safe optimization methods are accordingly extended to optimize process parameters with an environment in recurring experimental runs but varying context, avoiding violation of industrial regulations w.r.t. process quality. Therefore, we introduced an appropriate data discard strategy, local approximation techniques, and an ongoing exploration-exploitation trade-off approach. After evaluating the algorithm with synthetic test problems, we applied the approach to a saw blade straightening machine and achieve convincing results. Therefore, a snap was implemented that allows the learning framework to be executed directly in edge devices as a software module. Since the process control is also realized via a software module, communication can take place directly, as part of the common ecosystem. The evaluation results suggest that especially a high-dimensional context and, accordingly, a high data budget requires computing power. Here, fog or cloud computing, particularly in combination with high-performance communication standards (especially TSN and 5 G), will broaden the application spectrum. Depending on the application, the optimization can take place continuously or be triggered by the user or even automatically by an online anomaly detection. The latter suggestion is trivial to implement, since an expectation distribution already exists via the regression model. In this way, an autonomous system can differ between abrupt changes in system behavior, usual operation or optimization need.

In summary, we have introduced a novel software tool with a wide scope of applications and minimal setup effort for fine-tuning many industrial processes during daily production without additional interruption or increased failure rate. The demonstrated workflow can be used as a basis for further work. For example, the concept of snap communication and direct data access can be transferred to a wide variety of methods. Accordingly, the hope is that in the future, mid-sized machine builders will have the confidence to use the potential of ML for their very specific process optimization by using their domain knowledge to customize the optimization setup of our approach. In this way, the delivered systems can operate closer to their optimum on site in the future and are not rigidly bound to the specified default process parameter sets.

Limitations

The industrial application of the proposed approach and its current implementation is limited by some factors described in this section as well as by the stated assumptions regarding the problem definition. First, all experiments used the same kernel selection (the universal RBF kernel). The concepts can be transferred to most kernels, but not all (e.g. periodic or constant kernel). Second, the observations are required to provide an optimum, which is significantly different to the safety threshold. Otherwise, an unsafe exploration could be caused (De Blasi & Gepperth, 2020). Third, the computing power of edge controllers limits the complexity of problems to be solved. Here, the presented approximations with forgetting enhances the computational efficiency. Fourth, the presented Python-based framework limits the applications to firm real-time capability. For process optimization, this can be easily achieved by default policies, which are available or can be obtained in most industrial setups. Fifth, the data-efficiency of the presented approaches is depends on the default policy. With our method, safe optimization can only be recommended in industrial environments for process control fine tuning. Most limitations are therefore quite straightforward to circumvent through hardware or preliminary experimentation to improve the setup conditions.

Funding Open Access funding enabled and organized by Projekt DEAL.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Appendix

Justification for local GPR usage in safe optimization

The local GPR uses a subset with n_i of $n = n_i + n_o$ points. The n_o points are at least 3ℓ apart to the coordinate to be predicted. Lets assume that the optimization is a maximization with $f_{\min} = 1$ and $1 < f_{\max} < 5$ as the data is iteratively normalized. Furthermore, assume that local GPR is only used when $n_i > 50$ with RBF kernel.

We differ between performance relevant impact and safety relevant impact of the approximation to the optimization. If $l_{\text{GP}}(x_i) \geq f_{\min} > l_{\text{LGP}}(x_i)$ then the approximation affects only the performance of the optimization because $\hat{\mathcal{S}}_i$ becomes smaller. This must be accepted to take advantage of the prediction acceleration by approximating. If $l_{\text{GP}}(x_i) < f_{\min} \leq l_{\text{LGP}}(x_i)$ then the approximation is safety critical because $\hat{\mathcal{S}}_i$ would increase. This risk cannot be accepted for safe optimization.

If one can prove that

$$l_{\text{GP}}(x_i) < f_{\min} \iff l_{\text{LGP}}(x_i) - \epsilon < l_{\text{GP}}(x_i) \tag{21}$$

with a small constant ϵ, then all safety critical cases can be transformed into performance relevant cases:

$$\hat{\mathcal{S}}_{\text{LGP}} = \{x \in \mathcal{X}^D | l_{\text{LGP}}(x) - \epsilon \geq f_{\min}\}. \tag{22}$$

In this way, $\hat{\mathcal{S}}_{\text{LGP}} \subseteq \hat{\mathcal{S}}_{\text{GP}}$ and the safe optimization using LGP is as safe as the standard one, but might be less effective.

Theorem *Let GP be defined by (4) with zero mean, RBF kernel and* $\mathbf{x} = [x_1, \ldots, x_n]$ *with* $\min(\mathbf{x}) > 1$ *and* $\max(\mathbf{x}) < 5$*. If LGP is obtained by removing an arbitrary point* x_n*, which is at least* 3ℓ *apart to the prediction coordinate* x^**, then (21) holds.*

Proof The lower confidence bound of the prediction point x^* is

$$\begin{aligned} l_{\text{GP}}(x^*) &= \mu_{\text{GP}}(x^*) - \beta\sigma_{\text{GP}}(x^*) \\ &= \Sigma_*^T \Sigma^{-1} \hat{\mathbf{f}}_n - \beta(\Sigma_{**} - \Sigma_*^T \Sigma^{-1} \Sigma_*) \end{aligned}$$

for GP and,

$$\begin{aligned} l_{\text{LGP}}(x^*) &= \mu_{\text{LGP}}(x^*) - \beta\sigma_{\text{LGP}}(x^*) \\ &= \Pi_*^T \Pi^{-1} \hat{\mathbf{f}}_{n_i} - \beta(\Pi{*}{*} - \Pi_*^T \Pi^{-1} \Pi{*}) \end{aligned}$$

for LGP. According to block form inversion (Chou & de Queiroz, 2016), Σ^{-1} can be obtained based on Π^{-1}, as LGP has one point less than GP. Hence, it is possible to obtain $\mu_{\text{GP}}(x^*)$ and $\sigma_{\text{GP}}(x^*)$ based on $\mu_{\text{LGP}}(x^*)$ and $\sigma_{\text{LGP}}(x^*)$ respectively.

$$\begin{aligned} &\Sigma_*^T \Sigma^{-1} \hat{\mathbf{f}}_n \\ &= \begin{bmatrix} \Pi_*^T & k(x^*, x_n) \end{bmatrix} \begin{bmatrix} \Pi^{-1} + \frac{1}{c}\Pi^{-1}\mathbf{k}_n\mathbf{k}_n^T\Pi^{-1} & -\frac{1}{c}\Pi^{-1}\mathbf{k}_n \\ -\frac{1}{c}\mathbf{k}_n^T\Pi^{-1} & \frac{1}{c} \end{bmatrix} \\ &\quad \times \begin{bmatrix} \hat{\mathbf{f}}_{n-1} \\ \hat{f}(x_n) \end{bmatrix} \\ &= \Pi_*^T \Pi^{-1} \hat{\mathbf{f}}_{n-1} + \frac{\mathbf{k}_n^T \Pi^{-1} \hat{\mathbf{f}}_{n-1}}{c} \left[\Pi_*^T \Pi^{-1} \mathbf{k}_n - k(x^*, x_n)\right] \\ &\quad -\frac{1}{c}\Pi_*^T \Pi^{-1} \mathbf{k}_n \hat{f}(x_n) + \frac{1}{c} k(x^*, x_n) \hat{f}(x_n) \end{aligned}$$

where $c = 1 - \mathbf{k}_n^T \Pi^{-1} \mathbf{k}_n$ and $\mathbf{k}_n^T = [k(x_n, x_1), k(x_n, x_2), ..., k(x_n, x_{n-1})]$.

Considering that $\mu_{\text{LGP}}(x_n) = \mathbf{k}_n^T \Pi^{-1} \hat{\mathbf{f}}_{n-1}$, and $\mathbf{k}_n^T \Pi^{-1} \hat{\mathbf{f}}_{n-1} \approx \hat{f}(x_n)$:

$$\begin{aligned} \Sigma_*^T \Sigma^{-1} \hat{\mathbf{f}}_n \approx{} & \Pi_*^T \Pi^{-1} \hat{\mathbf{f}}_{n-1} + \frac{\hat{f}(x_n)}{c}\left[-k(x^*, x_n)\right] \\ & + \frac{1}{c} k(x^*, x_n) \hat{f}(x_n). \end{aligned}$$

Hence:

$$\Sigma_*^T \Sigma^{-1} \hat{\mathbf{f}}_n \approx \Pi_*^T \Pi^{-1} \hat{\mathbf{f}}_{n-1}. \tag{23}$$

In the same way, $\sigma_{\text{GP}}(x^*)$ is obtained based on $\sigma_{\text{LGP}}(x^*)$:

$$\begin{aligned} &\Sigma_*^T \Sigma^{-1} \Sigma_* \\ &= \begin{bmatrix} \Pi_*^T & k(x^*, x_n) \end{bmatrix} \begin{bmatrix} \Pi^{-1} + \frac{1}{c}\Pi^{-1}\mathbf{k}_n\mathbf{k}_n^T\Pi^{-1} & -\frac{1}{c}\Pi^{-1}\mathbf{k}_n \\ -\frac{1}{c}\mathbf{k}_n^T\Pi^{-1} & \frac{1}{c} \end{bmatrix} \end{aligned}$$

$$
\begin{aligned}
&\times \begin{bmatrix} \Pi_* \\ k(x^*, x_n) \end{bmatrix} \\
&= \Pi_*^T \Pi^{-1} \Pi_* + \frac{\mathbf{k}_n^T \Pi^{-1} \Pi_*}{c} \big[\Pi_*^T \Pi^{-1} \mathbf{k}_n \\
&\quad - k(x^*, x_n) \big] - \frac{k(x^*, x_n)}{c} \Pi_*^T \Pi^{-1} \mathbf{k}_n \\
&\quad + \frac{1}{c} k(x^*, x_n) \times k(x^*, x_n)
\end{aligned}
$$

where $c = 1 - \mathbf{k}_n^T \Pi^{-1} \mathbf{k}_n$ and $\mathbf{k}_n = [k(x_n, x_1), k(x_n, x_2), ..., k(x_n, x_{n-1})]$.

Schur complement implies that $Cov(x^*, x_n) = k(x^*, x_n) - \Pi_*^T \Pi^{-1} \mathbf{k}_n$, and $Cov(x^*, x_n) \approx k(x^*, x_n)$. Considering that $\mathbf{k}_n^T \Pi^{-1} \Pi_* = \Pi_*^T \Pi^{-1} \mathbf{k}_n$, because Π is symmetric positive semi definite matrix:

$$
\Sigma_*^T \Sigma^{-1} \Sigma_* = \Pi_*^T \Pi^{-1} \Pi_* + \frac{1}{c} k(x^*, x_n) \times k(x^*, x_n). \tag{24}
$$

Considering that $\Sigma_{**} = \Pi_{**}$, according to (23) and (24):

$$
\begin{aligned}
l_{\text{GP}}(x^*) &= \Sigma_*^T \Sigma^{-1} \hat{\mathbf{f}}_n - \beta(\Sigma_{**} - \Sigma_*^T \Sigma^{-1} \Sigma_*) \\
&\approx \Pi_*^T \Pi^{-1} \hat{\mathbf{f}}_{n-1} - \beta(\Pi_{**} - \Pi_*^T \Pi^{-1} \Pi_*) \\
&\quad - \frac{\beta}{c} k(x^*, x_n) \times k(x^*, x_n).
\end{aligned}
$$

If $|x^* - x| \geq 3 * \ell$, then $\exp\left(-\frac{(3\ell)^2}{2\ell^2}\right) < 0.011$, which implies that $k(x^*, x) < 0.011$. Hence:

$$
l_{\text{GP}}(x^*) > l_{\text{LGP}}(x^*) - \epsilon
$$

By experiments with Schwefel test function we found that $\epsilon \ll 0.1$ for the applied normalization under the made assumptions. □

Definition of test functions

The origins of applied test functions can be found in literature (Momin & Yang, 2013). They are modified for evaluation of safe optimizations, which is explained in detail below. We chose the sensitivity of the transformation in such a way that each test function already contains higher frequent changes in its standard form than the one that will be added. The modified Styblinski-Tang test function is defined by

$$
\begin{aligned}
f_{\text{S.-T.}}(x, z) &= \frac{-1}{2d} \sum_{r=1}^{d} (x^{(r)4} - 16x^{(r)2} + 5x^{(r)}) \\
&\quad + \frac{f_{max} - f_{min}}{20 d_c} \sum_{r=1}^{d} \sin(\frac{z^{(r)} \pi r}{10}),
\end{aligned} \tag{25}
$$

with $f_{max} = 39.17$ and $f_{min} = -0.119d^2 + 3.452d + 3.571$ as safety threshold.

The modified Alpine test function is defined by

$$
\begin{aligned}
f_{\text{Alpine}}(x, z) &= 5 \prod_{r=1}^{d} \sqrt{x^{(r)}} \sin(x^{(r)}) \\
&\quad + \frac{f_{max} - f_{min}}{20 d_c} \sum_{r=1}^{d} \sin(\frac{z^{(r)} \pi r}{10}),
\end{aligned} \tag{26}
$$

with $f_{max} = 13.5$ and $f_{min} = 11.5$ as safety threshold.

Additionally, a rather easy test problem is introduced, which is called Sinus test function and defined by

$$
f_{\text{Sinus}}(x, z) = \sin((\sin(z) + x)\pi + 0.5\pi) + 1, \tag{27}
$$

with $f_{min} = 1.6$ as safety threshold.

References

Akametalu, A. K., Fisac, J. F., Gillula, J. H., Kaynama, S., Zeilinger, M. N.,& Tomlin, C. J. (2014). Reachability-based safe learning with Gaussian processes. In: 53rd IEEE Conference on Decision and Control, pp. 1424–1431. IEEE.

Albrecht, J., Burchardt, G.,& Kleinfeller, M. (Aug. 2019). Verfahren zum Adressieren von Datenobjekten in einem Steuerungssystem einer Maschine (German Patent 102019212471A1).

Albrecht, J., Burchardt, G.,& Kleinfeller, M. (Jul. 2019). Method for transmitting data in a control system of a machine (U.S. Patent US020210055704A1).

Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J.,& Mané, D. (2016). Concrete problems in ai safety. arXiv preprint arXiv:1606.06565

Azizi, A. (2019). Applications of artificial intelligence techniques in industry 4.0. Springer.

Azizi, A. (2020). Applications of artificial intelligence techniques to enhance sustainability of industry 4.0: design of an artificial neural network model as dynamic behavior optimizer of robotic arms. Complexity **2020**.

Berkenkamp, F., Krause, A.,& Schoellig, A. P. (2021). Bayesian optimization with safety constraints: safe and automatic parameter tuning in robotics. Machine Learning pp. 1–35.

Berkenkamp, F., Schoellig, A. P.,& Krause, A. (2016). Safe controller optimization for quadrotors with Gaussian processes. In: 2016 IEEE International Conference on Robotics and Automation (ICRA), pp. 491–496. IEEE.

Boedecker, J., Springenberg, J. T., Wülfing, J.,& Riedmiller, M. (2014). Approximate real-time optimal control based on sparse Gaussian process models. In: 2014 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL), pp. 1–8. IEEE.

Bouneffouf, D.,& Rish, I. (2019). A survey on practical applications of multi-armed and contextual bandits. arXiv preprint arXiv:1904.10040.

Bui, T. D., Nguyen, C.,& Turner, R. E. (2017). Streaming sparse Gaussian process approximations. In: Advances in Neural Information Processing Systems, pp. 3299–3307.

Bui, T. D., Nguyen, C. V., Swaroop, S.,& Turner, R. E. (2018). Partitioned variational inference: A unified framework encompassing federated and continual learning. arXiv preprint arXiv:1811.11206

Chatzilygeroudis, K., Vassiliades, V., Stulp, F., Calinon, S.,& Mouret, J. B. (2018). A survey on policy search algorithms for learning robot controllers in a handful of trials. arXiv preprint arXiv:1807.02303

Chilès, J., & Desassis, N. (2018). Fifty years of kriging. *Handbook of mathematical geosciences* (pp. 589–612). Springer.

Chou, P. A.,& de Queiroz, R. L. (2016). Gaussian process transforms. In: 2016 IEEE International Conference on Image Processing (ICIP), pp. 1524–1528. IEEE.

Cressie, N. (1990). The origins of kriging. *Mathematical Geology, 22*(3), 239–252.

Csató, L., & Opper, M. (2002). Sparse on-line Gaussian processes. *Neural computation, 14*(3), 641–668.

De Blasi, S. (2019). Active learning approach for safe process parameter tuning. Optimization, and Data Science*Machine Learning* (pp. 689–699). Cham: Springer International Publishing.

De Blasi, S.,& Engels, E. (2020). Next generation control units simplifying industrial machine learning. In: IEEE 29th International Symposium on Industrial Electronics (ISIE), pp. 468–473. IEEE.

De Blasi, S.,& Gepperth, A. (2020). Sasbo: Self-adapting safe bayesian optimization. In: 2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 220–225. IEEE.

De Blasi, S., Klöser, S., Müller, A., Reuben, R., Sturm, F., & Zerrer, T. (2021). Kicker: An industrial drive and control foosball system automated with deep reinforcement learning. *J. Intell. Robotic Syst., 102*(1), 20.

De Blasi, S., Neifer, A.,& Gepperth, A. (2021). Multi-pronged safe bayesian optimization for high dimensions. In: 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 1966–1973. IEEE.

Demirkıran, G., Erdener, Ö., Akpınar, Ö., Demirtaş, P., Arık, M. Y., & Güler, E. (2020). Control of an inverted pendulum by reinforcement learning method in plc environment. In: 2020 Innovations in Intelligent Systems and Applications Conference (ASYU), pp. 1–5. IEEE.

Duivenvoorden, R. R., Berkenkamp, F., Carion, N., Krause, A., & Schoellig, A. P. (2017). Constrained bayesian optimization with particle swarms for safe adaptive controller tuning. *IFAC-PapersOnLine, 50*(1), 11800–11807.

Fiducioso, M., Curi, S., Schumacher, B., Gwerder, M.,& Krause, A. (2019). Safe contextual bayesian optimization for sustainable room temperature pid control tuning. In: Proceedings of the 28th International Joint Conference on Artificial Intelligence, pp. 5850–5856.

Fisac, J. F., Akametalu, A. K., Zeilinger, M. N., Kaynama, S., Gillula, J., & Tomlin, C. J. (2018). A general safety framework for learning-based control in uncertain robotic systems. *IEEE Transactions on Automatic Control, 64*(7), 2737–2752.

Garcıa, J., & Fernández, F. (2015). A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research, 16*(1), 1437–1480.

Gelbart, M. A., Snoek, J.,& Adams, R. P. (2014). Bayesian optimization with unknown constraints. In: 30th Conference on Uncertainty in Artificial Intelligence, UAI 2014, pp. 250–259. AUAI Press.

Geramifard, A., Redding, J., & How, J. P. (2013). Intelligent cooperative control architecture: a framework for performance improvement using safe learning. *Journal of Intelligent & Robotic Systems, 72*(1), 83–103.

Ghafoorpoor Yazdi, P., Azizi, A.,& Hashemipour, M. (2018). An empirical investigation of the relationship between overall equipment efficiency (oee) and manufacturing sustainability in industry 4.0 with time study approach. Sustainability **10**(9), 3031

Gheibi, O., Weyns, D., & Quin, F. (2021). Applying machine learning in self-adaptive systems: A systematic literature review. *ACM Transactions on Autonomous and Adaptive Systems (TAAS), 15*(3), 1–37.

Google. (2014). Flatbuffers. https://github.com/google/flatbuffers

GPy. (2012). A Gaussian process framework in python. http://github.com/SheffieldML/GPy (since)

Gramacy, R. B. (2016). lagp: Large-scale spatial modeling via local approximate Gaussian processes in r. *Journal of Statistical Software, 72*(1), 1–46.

Gramacy, R. B., & Apley, D. W. (2015). Local Gaussian process approximation for large computer experiments. *Journal of Computational and Graphical Statistics, 24*(2), 561–578.

Gulde, R., Tuscher, M., Csiszar, A., Riedel, O.,& Verl, A. (2019). Reinforcement learning approach to vibration compensation for dynamic feed drive systems. In: 2019 Second International Conference on Artificial Intelligence for Industries (AI4I), pp. 26–29. IEEE.

Hameed, M.S.A.,& Schwung, A. (2020). Reinforcement learning on job shop scheduling problems using graph networks. arXiv preprint arXiv:2009.03836

Hensman, J., Fusi, N.,& Lawrence, N.D. (2013). Gaussian processes for big data. In: Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence, pp. 282–290.

Herbrich, R., Lawrence, N.D.,& Seeger, M. (2003). Fast sparse Gaussian process methods: The informative vector machine. In: Advances in neural information processing systems, pp. 625–632.

Hernández-Lobato, J. M., Gelbart, M. A., Adams, R. P., Hoffman, M. W., & Ghahramani, Z. (2016). A general framework for constrained bayesian optimization using information-based search. *The Journal of Machine Learning Research, 17*(1), 5549–5601.

Iwanek, P., Gausemeier, J.,& Dumitrescu, R. (2018). Potenzialanalyse zur steigerung der intelligenz mechatronischer systeme. In: Steigerung der Intelligenz mechatronischer Systeme, pp. 39–72. Springer.

Jaensch, F., Csiszar, A., Sarbandi, J.,& Verl, A. (2019). Reinforcement learning of a robot cell control logic using a software-in-the-loop simulation as environment. In: 2019 Second International Conference on Artificial Intelligence for Industries (AI4I), pp. 79–84. IEEE.

Jagerman, R., Markov, I., & Rijke, M. D. (2020). Safe exploration for optimizing contextual bandits. *ACM Transactions on Information Systems (TOIS), 38*(3), 1–23.

Kazerouni, A., Ghavamzadeh, M., Yadkori, Y. A.,& Van Roy, B. (2017). Conservative contextual linear bandits. In: Advances in Neural Information Processing Systems, pp. 3910–3919.

Kim, D. H., Kim, T. J., Wang, X., Kim, M., Quan, Y. J., Oh, J. W., Min, S. H., Kim, H., Bhandari, B., Yang, I., et al. (2018). Smart machining process using machine learning: A review and perspective on machining industry. *International Journal of Precision Engineering and Manufacturing-Green Technology, 5*(4), 555–568.

Kirschner, J., Mutny, M., Hiller, N., Ischebeck, R.,& Krause, A. (2019). Adaptive and safe bayesian optimization in high dimensions via one-dimensional subspaces. In: International Conference on Machine Learning, pp. 3429–3438.

Kirschner, J., Nonnenmacher, M., Mutnỳ, M., Krause, A., Hiller, N., Ischebeck, R.,& Adelmann, A. (2019). Bayesian optimisation for fast and safe parameter tuning of swissfel. In: FEL2019, Proceedings of the 39th International Free-Electron Laser Conference, pp. 707–710. JACoW Publishing.

Krause, A.,& Ong, C.S. (2011). Contextual Gaussian process bandit optimization. In: Advances in Neural Information Processing Systems (NIPS), pp. 2447–2455

Liu, A., Shi, G., Chung, S.J., Anandkumar, A.,& Yue, Y. (2019). Robust regression for safe exploration in control. arXiv preprint arXiv:1906.05819

Liu, H., Ong, Y.S., Shen, X.,& Cai, J. (2020). When Gaussian process meets big data: A review of scalable gps. IEEE Transactions on Neural Networks and Learning Systems.

Liu, J., Ye, J., Silva Izquierdo, D., Vinel, A., Shamsaei, N.,& Shao, S. (2022). A review of machine learning techniques for process and performance optimization in laser beam powder bed fusion additive manufacturing. Journal of Intelligent Manufacturing pp. 1–27.

Maier, M., Rupenyan, A., Bobst, C.,& Wegener, K. (2020). Self-optimizing grinding machines using Gaussian process models and constrained Bayesian optimization. The International Journal of Advanced Manufacturing Technology.

Micchelli, C. A., Xu, Y.,& Zhang, H. (2006). Universal kernels. *Journal of Machine Learning Research, 7*, 2651–2667.

Mockus, J. (2012). Bayesian approach to global optimization: Theory and applications, vol. 37. Springer.

Möhring, H. C., Wiederkehr, P., Erkorkmaz, K., & Kakinuma, Y. (2020). Self-optimizing machining systems. *CIRP Annals, 69*(2), 740–763.

Momin, J., & Yang, X. S. (2013). A literature survey of benchmark functions for global optimization problems. *Int. Journal of Mathematical Modelling and Numerical Optimisation, 4*(2), 150–194.

Nguyen-Tuong, D., Peters, J.R.,& Seeger, M. (2009). Local Gaussian process regression for real time online model learning. In: Advances in Neural Information Processing Systems, pp. 1193–1200.

Nian, R., Liu, J.,& Huang, B. (2020). A review on reinforcement learning: Introduction and applications in industrial process control. *Computers & Chemical Engineering, 1*, 106886.

Oba, S., Sato, M. A.,& Ishii, S. (2001). On-line learning methods for Gaussian processes. In: International Conference on Artificial Neural Networks, pp. 292–299. Springer

Park, H. S., & Tran, N. H. (2014). Development of a smart machining system using self-optimizing control. *The International Journal of Advanced Manufacturing Technology, 74*(9–12), 1365–1380.

Parrot, C.,& Venayagamoorthy, G. K. (2008). Real-time implementation of intelligent modeling and control techniques on a plc platform. In: 2008 IEEE Industry Applications Society Annual Meeting, pp. 1–7. IEEE.

Permin, E., Bertelsmeier, F., Blum, M., Bützler, J., Haag, S., Kuz, S., Özdemir, D., Stemmler, S., Thombansen, U.,& Schmitt, R., et al. (2016). Self-optimizing production systems. Procedia Cirp **41**(C), 417–422.

Polymenakos, K., Abate, A.,& Roberts, S. (2019). Safe policy search using Gaussian process models. In: Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, pp. 1565–1573. International Foundation for Autonomous Agents and Multiagent Systems.

Qu, Y., Ming, X., Liu, Z., Zhang, X.,& Hou, Z. (2019). Smart manufacturing systems: State of the art and future trends. *The International Journal of Advanced Manufacturing Technology, 1*, 1–18.

Quiñonero-Candela, J.,& Rasmussen, C. E. (2005). A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research, 6*, 1939–1959.

Rasmussen, C., & Williams, C. (2006). Gaussian processes for machine learning, vol. 1. *MIT Press, 39*, 40–43.

Rattunde, L., Laptev, I., Klenske, E. D., & Möhring, H. C. (2021). Safe optimization for feedrate scheduling of power-constrained milling processes by using Gaussian processes. *Procedia CIRP, 99*, 127–132.

Rauch, E. (2020). Industry 4.0+: the next level of intelligent and self-optimizing factories. In: Design, Simulation, Manufacturing: The Innovation Exchange, pp. 176–186. Springer

Renard, D.,& Yancey, S. (1984). Smoothing discontinuities when extrapolating using moving neighbourhood. In: Geostatistics for natural resources characterization. NATO advanced Study Institute, pp. 679–690.

Ribeiro, M. T., Singh, S.,& Guestrin, C. (2016). "why should i trust you?" explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp. 1135–1144.

Schmidl, E., Fischer, E., Wenk, M.,& Franke, J. (2020). Knowledge-based generation of a plant-specific reinforcement learning framework for energy reduction of production plants. In: 2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), vol. 1, pp. 1–4. IEEE.

Schmidt, A., Schellroth, F.,& Riedel, O. (2020). Control architecture for embedding reinforcement learning frameworks on industrial control hardware. In: Proceedings of the 3rd International Conference on Applications of Intelligent Systems, pp. 1–6.

Schwung, D., Csaplar, F., Schwung, A.,& Ding, S. X. (2017). An application of reinforcement learning algorithms to industrial multi-robot stations for cooperative handling operation. In: 2017 IEEE 15th International Conference on Industrial Informatics (INDIN), pp. 194–199. IEEE.

Schwung, D., Kempe, T., Schwung, A.,& Ding, S. X. (2017). Self-optimization of energy consumption in complex bulk good processes using reinforcement learning. In: 2017 IEEE 15th International Conference on Industrial Informatics (INDIN), pp. 231–236. IEEE.

Schwung, D., Modali, M.,& Schwung, A. (2019). Self-optimization in smart production systems using distributed reinforcement learning. In: 2019 IEEE International Conference on Systems, Man and Cybernetics (SMC), pp. 4063–4068. IEEE.

Schwung, D., Schwung, A.,& Ding, S. X. (2018). On-line energy optimization of hybrid production systems using actor-critic reinforcement learning. In: 2018 International Conference on Intelligent Systems (IS), pp. 147–154. IEEE.

Seeger, M., Williams, C. K.,& Lawrence, N. D. (2003). Fast forward selection to speed up sparse Gaussian process regression. In: Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics (AISTATS).

Sergeyev, Y. D., Candelieri, A., Kvasov, D. E., & Perego, R. (2020). Safe global optimization of expensive noisy black-box functions in the δ-lipschitz framework. *Soft Computing, 24*(23), 17715–17735.

Sergeyev, Y.D., Candelieri, A., Kvasov, D. E.,& Perego, R. (2020). Safe global optimization of expensive noisy black-box functions in the δ-lipschitz framework. Soft Computing pp. 1–21.

Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., & De Freitas, N. (2015). Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE, 104*(1), 148–175.

Snelson, E.,& Ghahramani, Z. (2006). Sparse Gaussian processes using pseudo-inputs. In: Advances in neural information processing systems, pp. 1257–1264.

Sui, Y., Burdick, J.,& Yue, Y., et al. (2018). Stagewise safe bayesian optimization with Gaussian processes. In: International Conference on Machine Learning, pp. 4781–4789.

Sui, Y., Gotovos, A., Burdick, J.,& Krause, A. (2015). Safe exploration for optimization with Gaussian processes. In: International Conference on Machine Learning, pp. 997–1005.

Titsias, M. (2009). Variational learning of inducing variables in sparse Gaussian processes. In: Artificial Intelligence and Statistics, pp. 567–574.

Turchetta, M., Berkenkamp, F.,& Krause, A. (2019). Safe exploration for interactive machine learning. In: Advances in Neural Information Processing Systems, pp. 2891–2901.

Vargas, A., Cuenca, L., Boza, A., Sacala, I., & Moisescu, M. (2016). Towards the development of the framework for inter sensing enterprise architecture. *Journal of Intelligent Manufacturing, 27*(1), 55–72.

Weichert, D., Link, P., Stoll, A., Rüping, S., Ihlenfeldt, S., & Wrobel, S. (2019). A review of machine learning for the optimization of production processes. *The International Journal of Advanced Manufacturing Technology, 104*(5–8), 1889–1902.

Williams, C. K.,& Seeger, M. (2001). Using the nyström method to speed up kernel machines. In: Advances in neural information processing systems, pp. 682–688.

Wu, Y., Shariff, R., Lattimore, T., & Szepesvári, C. (2016). Conservative bandits. In: International Conference on Machine Learning, pp. 1254–1262.

Zeiser, A., van Stein, B., & Bäck, T. (2021). Requirements towards optimizing analytics in industrial processes. *Procedia Computer Science, 184*, 597–605.

Zhong, R. Y., Xu, X., Klotz, E., & Newman, S. T. (2017). Intelligent manufacturing in the context of industry 4.0: A review. *Engineering, 3*(5), 616–630.

Zuehlke, D. (2010). Smartfactory-towards a factory-of-things. *Annual Reviews in Control, 34*(1), 129–138.