Elsevier Editorial System(tm) for ISA Transactions
Manuscript Draft

Manuscript Number: ISATRANS-D-20-00221

Title: Gaussian Processes for Shock Test Emulation

Article Type: Research article

Section/Category: Application

Keywords: Gaussian Process; Machine Learning; Uncertainty Quantification; Structural Dynamics; Bernoulli Trials; Virtual Shock Testing

Corresponding Author: Mr. Christophe Pol Andre Bonneville,

Corresponding Author's Institution: Cornell University

First Author: Christophe Pol Andre Bonneville

Order of Authors: Christophe Pol Andre Bonneville; Maxwell Jenquin; Juan Londono; Alex Kelly; Jeffrey Cipolla; Christopher Earls

Abstract: Certifying favorable performance of mechanical components with experimental tests is time consuming and expensive, which motivates the development of efficient approaches for predicting the outcomes from such testing, a priori. In this paper, we propose two such methods based on Gaussian processes (GP) to estimate the probability that new components will pass future certification tests, while assessing the confidence in our predictions. The first method (applicable to experimental training data) processes a set of Bernoulli trials into a suitable machine learning dataset and infers the probability of performing satisfactorily for new components using heteroscedastic bounded GP regression. The second method (for use with surrogate training data) uses GP classification with linear kernels. We demonstrate that linear kernels are well suited for datasets representing snapshots of mechanical system responses as they accurately reproduce the underlying physics by reflecting monotonic trends in the data. This yields consistent probabilities of passing and provides high labeling accuracy, even with small datasets. We demonstrate these techniques on synthetic datasets consistent with ship cabinet certification tests. We achieve between 96% and 100% accuracy using all of the training data, and at least 92% with only 10% of the available data. With a highly corrupted training set, we obtain between 93% and 100% accuracy. In the regression framework, we demonstrate that introducing heteroscedasticity into a bounded GP helps achieving significantly better accuracy than frequentist machine learning methods (up to r2=0.981 with GPs, compared to r2=0.873 and r2=0.866 with linear regression and neural networks, respectively).

Suggested Reviewers:

*Cover Letter

Christophe Bonneville
Cornell University
cpb97@cornell.edu
411 Hollister Hall
14850 Ithaca NY, USA

Dear editor,

We wish to submit our manuscript ("Gaussian Processes for Shock Test Emulation") for consideration by your Journal, *ISA Transactions*. This manuscript has not been published elsewhere, and to the best of our knowledge represents an original contribution to engineering research.

Performing shock certification tests of mechanical systems in real conditions is both expensive and time consuming. Relying on deterministic numerical simulation instead is usually not enough to fully capture the uncertainty in the tests outcome, which motivates the development of novel probabilistic approaches. In this manuscript, we show that Gaussian processes (GPs) are particularly well suited for predicting the probability of failure of mechanical tests involving structural dynamic systems, using either experimental data or surrogate synthetic data.

- In the case of experimental data with repeated testing, we develop a novel data processing approach to encode data uncertainty within a bounded GP regression framework. This approach allows us to obtain excellent prediction accuracy (up to $r^2=0.981$), even with a small and noisy dataset and outperform other non-Bayesian machine learning models.
- In the case of surrogate data or experimental data with single testing, we show that using linear kernels within the GP classification framework permits to accurately capture the manifestation of mechanical failure modes. Hence it allows us to achieve excellent labeling accuracy (between 93% and 100%), even with small dataset and/or highly corrupted data, and typically outperform other choices of GP covariance kernels.

Our manuscript stands at the intersection between uncertainty quantification, applied machine learning and virtual testing, with application to structural mechanics. It introduces both novel theoretical methods as well as practical application examples. As a result, we believe that this manuscript appropriately matches the editorial line of your journal.

Thank you for considering this manuscript for review.

Sincerely,
Christophe Bonneville

***Title page showing Author Details**

GAUSSIAN PROCESSES FOR SHOCK TEST EMULATION

A PREPRINT

Christophe Bonneville
Civil & Environmental Engineering
Cornell University
Ithaca, NY 14850
cpb97@cornell.edu

Maxwell Jenquin
Center for Applied Mathematics
Cornell University
Ithaca, NY 14850

Juan Londono
Applied Science
Thornton Tomasetti
New York, NY, 10005

Alex Kelly
Applied Science
Thornton Tomasetti
New York, NY, 10005

Jeffrey Cipolla
Applied Science
Thornton Tomasetti
Washington DC, 20006

Christopher Earls
Center for Applied Mathematics
Civil & Environmental Engineering
Cornell University
Ithaca, NY 14850
earls@cornell.edu

January 27, 2020

ABSTRACT

Certifying favorable performance of mechanical components with experimental tests is time consuming and expensive, which motivates the development of efficient approaches for predicting the outcomes from such testing, *a priori*. In this paper, we propose two such methods based on Gaussian processes (GP) to estimate the probability that new components will pass future certification tests, while assessing the confidence in our predictions. The first method (applicable to experimental training data) processes a set of Bernoulli trials into a suitable machine learning dataset and infers the probability of performing satisfactorily for new components using heteroscedastic bounded GP regression. The second method (for use with surrogate training data) uses GP classification with linear kernels. We demonstrate that linear kernels are well suited for datasets representing snapshots of mechanical system responses as they accurately reproduce the underlying physics by reflecting monotonic trends in the data. This yields consistent probabilities of passing and provides high labeling accuracy, even with small datasets. We demonstrate these techniques on synthetic datasets consistent with ship cabinet certification tests. We achieve between 96% and 100% accuracy using all of the training data, and at least 92% with only 10% of the available data. With a highly corrupted training set, we obtain between 93% and 100% accuracy. In the regression framework, we demonstrate that introducing heteroscedasticity into a bounded GP helps achieving significantly better accuracy than frequentist machine learning methods (up to $r^2 = 0.981$ with GPs, compared to $r^2 = 0.873$ and $r^2 = 0.866$ with linear regression and neural networks, respectively).

***K*eywords** Gaussian Process · Machine Learning · Uncertainty Quantification · Structural Dynamics · Bernoulli Trials · Virtual Shock Testing

***Highlights (for review)**

- Gaussian processes (GP) are powerful tools for estimating the probability of failure of mechanical systems
- Experimental data can be processed into Gaussian process training data with valuable uncertainty information
- Bounded Gaussian process regressors with RBF kernels trained on properly processed experimental data provide excellent accuracy and outperform deterministic machine learning models such as neural network
- Gaussian process classifiers with linear kernels trained on surrogate data provide excellent prediction accuracy, even with noisy or small datasets

GAUSSIAN PROCESSES FOR SHOCK TEST EMULATION

A PREPRINT

January 31, 2020

ABSTRACT

Certifying favorable performance of mechanical components with experimental tests is time consuming and expensive, which motivates the development of efficient approaches for predicting the outcomes from such testing, *a priori*. In this paper, we propose two such methods based on Gaussian processes (GP) to estimate the probability that new components will pass future certification tests, while assessing the confidence in our predictions. The first method (applicable to experimental training data) processes a set of Bernoulli trials into a suitable machine learning dataset and infers the probability of performing satisfactorily for new components using heteroscedastic bounded GP regression. The second method (for use with surrogate training data) uses GP classification with linear kernels. We demonstrate that linear kernels are well suited for datasets representing snapshots of mechanical system responses as they accurately reproduce the underlying physics by reflecting monotonic trends in the data. This yields consistent probabilities of passing and provides high labeling accuracy, even with small datasets. We demonstrate these techniques on synthetic datasets consistent with ship cabinet certification tests. We achieve between 96% and 100% accuracy using all of the training data, and at least 92% with only 10% of the available data. With a highly corrupted training set, we obtain between 93% and 100% accuracy. In the regression framework, we demonstrate that introducing heteroscedasticity into a bounded GP helps achieving significantly better accuracy than frequentist machine learning methods (up to $r^2 = 0.981$ with GPs, compared to $r^2 = 0.873$ and $r^2 = 0.866$ with linear regression and neural networks, respectively).

Keywords Gaussian Process · Machine Learning · Uncertainty Quantification · Structural Dynamics · Bernoulli Trials · Virtual Shock Testing

1 Introduction

Mechanical components implemented inside advanced technological systems are often subject to a strict certification process, to ensure survivability against vibrations and shock loading conditions. Such certifications are required in a variety of engineering fields, like in computer hard drive design (Tandon et. al., 2006 [1]), spacecraft electronic protective cases (Aggarwal, 2010, [2]), or in US Navy ships and submarines, where components such as electronic racks need to pass a battery of mechanical tests designed to simulate extreme shock scenarios (naval warfare, blasts, *etc.*), as described in military specifications (MIL-S-901D, 1989, [3]). Conducting these certification tests is time consuming and expensive, which motivates the development of alternative certification approaches. In many mechanical engineering applications, employing numerical techniques, such as finite element method, may provide excellent predictions; thus allowing in some cases for completely avoiding experimental validation testing. However, finite element methods yielding the required accuracy for use in certification may sometimes be as costly as the physical certification test, itself. Moreover, standard finite element methods are inherently deterministic, and thus do not account for realistic and unavoidable uncertainties. For example, while a certain numerically simulated test might be successful (regardless of how many time the simulation is repeated), the same test conducted experimentally may exhibit a small probability of failing. Hence, numerical simulations may be insufficient in practice when uncertainty dominates, like in shock testing. In addition to knowing the tests outcomes (passed or failed) we ultimately want to estimate the probability of each outcome. Moreover, to make informed decisions, it is desirable to assess the reliability of our predictions; thus we also wish to output confidence intervals.

The above mentioned aims clearly require a stochastic model (rather than a deterministic one); thus we propose to use a Bayesian machine learning approach, based on Gaussian processes (GPs). Machine learning is a natural choice as we can predict a pass/fail probability for new and previously untested components, by leveraging either experimental or synthetic (numerically-generated) training data. GPs specifically embody qualities that are germane to our aims in the present work: First, a GP naturally outputs a confidence interval describing uncertainty in the prediction it deems the most likely; secondly, as a Bayesian method, we can encode prior beliefs about the expected behavior of our system of interest, given the underlying physics generating the data; thirdly, and as consequence of the second point, GPs can give very good results even with small datasets, which is often convenient when experiments are expensive.

In this paper, we present two different approaches for estimating the probability of passing a certification test. The first approach, based on GP regression, is suited for experimentally-generated training data, in which a certification test is repeated several times for the same mechanical component. The second approach, based on GP classification, is suited for either numerically generated training data or experimental training data where only a single test is performed per component. In the following sections, we first introduce elements of GP theory. We then introduce strategies for making predictions on experimental or synthetic datasets, and finally, we present and discuss our results.

2 Gaussian Process Theory

In this section of the manuscript, we introduce salient theoretical elements involved in GP regression and classification.

2.1 Gaussian Process Regression

The current subsection introduced GPs but primarily has as its focus GP regression.

2.1.1 Standard Regression Model

Popularized by Rasmussen & Williams (2006) [4], GPs are a set of supervised machine learning tools based on the *discriminative approach*, in which the goal is to learn a latent function, f, that maps a set of input vectors, $X = \{\vec{x_1}, \ldots, \vec{x_n}\}$, into a corresponding set of scalar outputs, $y = \{y_1, \ldots, y_n\}$.

For simplicity, in the following we will drop the arrows on $\vec{x_i}$. In the regression framework, y_i is a continuous real value. Every training sample (x_i, y_i) is assumed to be independently and identically distributed (*i.i.d*) from an unknown probability density function. In standard discriminative models, the outputs are assumed to have been generated from f and corrupted by Gaussian noise, ϵ, with variance, σ^2 (equation 1). Assumptions must be made about the functional form of the latent function, and in Bayesian approaches it is modeled as a stochastic function. With GPs in particular, the discrete form of f, expressed a collection of neighbouring points is conceived as a vector sampled from a prior multivariate Gaussian distribution (equation 2), with a zero mean and an input-dependant covariance kernel.

$$\begin{cases} y_i = f_i + \epsilon_i \\ \epsilon_i \sim \mathcal{N}(0, \sigma^2) \end{cases} \tag{1}$$

$$f \sim \mathcal{GP}(0, k(X, X)) \tag{2}$$

2.1.2 Covariance Kernel

The covariance of f is expressed through a kernel function, $k(X, X)$, which upon discretization leads to a covariance matrix. The kernel is an arbitrary function that plays a key role in GPs, as it *a priori* strongly influences the behavior of the model; hence, its importance cannot be overstated. One may choose a kernel that encodes prior knowledge about the data, or some other behaviors that should be included in the model (smoothness, linearity, *etc*). For example, a popular choice is the RBF kernel (equation 3) which describes the intuitive idea that data points close to each other are likely to be more correlated than points further away.

$$k_{\text{RBF}}(x_i, x_j; \theta) = \theta_1 \exp\left(- \frac{\|x_i - x_j\|^2}{2\theta_2^2} \right) \tag{3}$$

The hyperparameters, $\theta = \{\theta_1, \theta_2\}$, scale the covariance between data points, and therefore significantly influence the model behavior. Indeed, the hyperparameters instantiate the kernel function and are learned when the GP is trained on the previously available data. In the RBF kernel, θ_2 acts as a lengthscale hyperparameter, describing how much data points close to each other are related (*e.g.* a small lengthscale means that points far away from each other are less correlated, eventually leading to faster variations in output predictions). Note that existing kernels may be combined together to form new ones (*e.g.* sum or product of kernels), as long as the resulting kernel function yields a positive semi-definite matrix upon discretization (Rasmussen & Williams, 2006, [4]).

2.1.3 Marginal Likelihood Optimization

During GP training, the hyperparameters $\theta = \{\theta_1, \ldots, \theta_k\}$ can be learned automatically from the data for any kernel type, by maximizing the marginal likelihood $p(y|X, \theta)$. The marginal likelihood is the probability that the model generated the observed data, given θ. Hence, it quantifies how well the current hyperparameters describe the dataset, when contextualized with the associated covariance function. The marginal likelihood is derived by a combined application of the sum and product rule of probability:

$$\begin{aligned} p(y|X, \theta) &= \int p(y|f)p(f|X, \theta)df \\ &= \int \mathcal{N}(y|f, \sigma^2 I_n)\mathcal{N}(f|0, k(X, X; \theta))df \end{aligned} \tag{4}$$

The expression for the likelihood $p(y|f)$ is an immediate consequence of equation 1. The integral above is analytically tractable, and is itself Gaussian. Instead of maximizing directly $p(y|X,\theta)$ with respect to θ, it is equivalent, but computationally more efficient, to minimize the negative marginal-log-likelihood (MLL):

$$\log p(y|X,\theta) = -\frac{1}{2}y^\top(k(X,X;\theta)+\sigma^2 I_n)^{-1}y - \frac{1}{2}\log(|k(X,X;\theta)+\sigma^2 I_n|) - \frac{n}{2}\log(2\pi) \quad (5)$$

The term $\log(|k(X,X;\theta) + \sigma^2 I_n|)$ acts as a penalty term that prevents the hyperparameters from taking either too small or too large values (for simplicity in the following, we will omit the θ notation in $k(X,X;\theta)$). Note that this penalty term depends on the size of the available data, so a GP model automatically adapts its complexity to the dataset. As a result, GPs are naturally less prone to overfitting, unlike frequentist machine learning methods (Neal, 1995 [5], Rasmussen & Ghahramani, 2001 [6]).

In most applications, the variance σ^2 of the noise corrupting the data is not known in advance and is an arbitrary choice. For example, it may be chosen through an iterative process such that $k(X,X) + \sigma^2 I_n$ is invertible (this is always the case for σ^2 large enough). In the context of shock test validation however, a more sophisticated approach may be employed for selecting the noise variance, as we shall see in the following developments.

2.1.4 Predictive Distribution

The prior distribution, defined in equation 2, is a theoretical representation of the GP model and is not to be used on its own for making predictions. Bayesian approaches do not output a single prediction value, but rather a probability distribution over the prediction (latent posterior). In the GP framework, a new input point, x_*, is padded to the training data in the prior covariance kernel (equation 2) and the noise variance in the data is added to the prior covariance matrix $k(X,X)$, yielding the joint distribution $p(y, f_*|X, x_*)$ written below:

$$\begin{bmatrix} y \\ f_* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} k(X,X)+\sigma^2 I_n & k(X,x_*) \\ k(x_*,X) & k(x_*,x_*) \end{bmatrix}\right) \quad (6)$$

As shown in Rasmussen & Williams (2006) [4], the resulting predictive distribution is also Gaussian and can be derived by applying the sum and product rule to the joint distribution above:

$$p(f_*|X,y,x_*) = \frac{p(y,f_*|X,x_*)}{\int p(y,f_*|X,x_*)df_*} = \mathcal{N}(f_*|\mu_*,\Sigma_*) \quad (7)$$

$$\begin{cases} \mu_* = k(x_*,X)(k(X,X)+\sigma^2 I_n)^{-1}y \\ \Sigma_* = k(x_*,x_*) - k(x_*,X)(k(X,X)+\sigma^2 I_n)^{-1}k(X,x_*) \end{cases} \quad (8)$$

Given the training data and the kernel choice, the most likely predictive output is μ_*, with confidence Σ_*. Notice that in the GP framework, although the data are modeled with a discrete vector, that has no analytic expression (equation 2), it is nonetheless possible to make predictions for any new feature vector x_*.

2.2 Binary Gaussian Process Classification

The present subsection extends the notion of GPs for use in classification applications.

2.2.1 Probit Classification Model

In binary classification, the outputs are discrete and correspond to labels (*e.g.* -1 for a failed test and $+1$ for a passed test). The GP classification procedure is similar to GP regression, however the goal is now to predict the probability that a new input belongs to either one of the two labels. To do so, the latent function is warped into the interval $[0, 1]$ using a standard normal cumulative distribution function (CDF). This yields a likelihood of the form:

$$p(y|f) = \prod_{i=1}^{n} p(y_i|f_i) = \prod_{i=1}^{n} \Phi(y_i f_i) \tag{9}$$

The likelihood is defined such that $y_i = +1$ if $f_i > 0$ and $y_i = -1$ if $f_i < 0$. Hence, if y_i and f_i have the same sign, $p(y_i|f_i) = \Phi(y_i \times f_i) > 0.5$ (Rasmussen & Williams (2006) [4]). Since the likelihood is no longer Gaussian, the integral in equation 4 is analytically intractable and needs to be approximated. As listed by Nickisch & Rasmussen (2008) [7], numerous approximation methods can be used to evaluate the MLL (Laplace, MCMC, variational bounds, *etc.*) and in the following developments, we will rely on variational inference (Blei et. al. 2017 [8]). Similarly to regression, the approximated negative MLL can then be minimized with respect to θ.

2.2.2 Label Prediction

The predictive distribution over the latent posterior f_* (equation 7) also needs to be approximated, but yields a predictive mean and variance analogous to equation 8. Subsequently, the probability that x_* belongs to the $+1$ label can be analytically approximated for normal CDF warping functions (Rasmussen & Williams, 2006 [4]).

$$p(y_* = +1|X, y, x_*) = \int \Phi(f_*) p(f_*|X, y, x_*) df_* \approx \Phi\left(\frac{\mu_*}{\sqrt{1 + \Sigma_*}}\right) \tag{10}$$

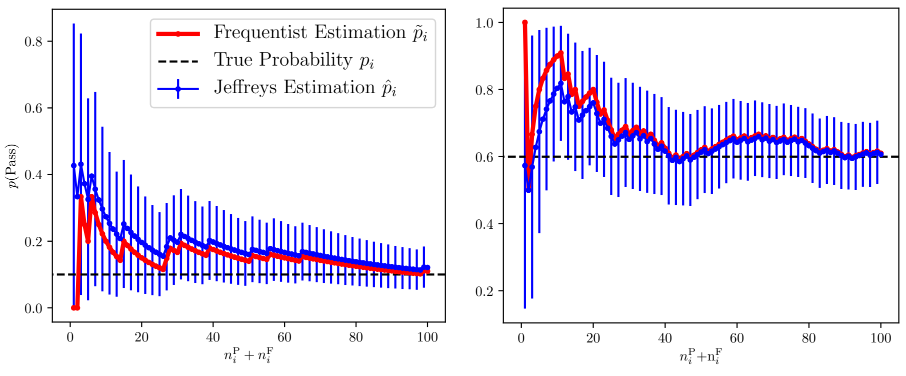
Likewise, the probability of failing is $p(y_* = -1|X, y, x_*) = 1 - p(y_* = +1|X, y, x_*)$.

3 GP Regression on Experimental Data

In this section, we introduce a GP regression method for estimating the probability that a new mechanical component will pass certification tests given previous experimental observations.

3.1 Raw Dataset

We assume that the same test has been performed numerous times for each component in the training dataset. In order to be suitable for machine learning inputs, we further assume that all the components have a similar geometry, such that they can be described by the same features (*e.g.* a girder where the 1st feature represents the length, the 2nd feature represents the type of cross-section, *etc.*). When the input features are very close to critical values, the same component may sometime pass and sometime fail (these are the ambiguous input regions of interest, since we want to assess experimental uncertainties). Subsequently, we consider raw datasets of the form $\mathcal{D}_i^{\text{raw}} = \{x_i, y_i^{\text{raw}} \Rightarrow (n_i^{\text{P}}, n_i^{\text{F}})\}$, where n_i^{P} and n_i^{F} are the number of times x_i passed and failed the test respectively.



(a) True $p_i = 0.1$ (b) True $p_i = 0.6$

Figure 1: Jeffreys interval convergence for an increasing number of trial. $\tilde{p}_i$ and $\hat{p}_i$ are estimated using n_i^{P} and $n_i^{\mathrm{F}} = n_i^{\mathrm{trial}} - n_i^{\mathrm{P}}$, where n_i^{P} is the number of times we observe $s = +1$, with s randomly sampled from $p_i^s(1-p_i)^{1-s}$ (over n_i^{trial} samplings in total).

3.2 Data Preprocessing

The form of the data presented above is not directly suitable for machine learning and requires preprocessing. Since the tests outcomes are binary, it may be tempting to create a new dataset that repeats each point n_i^{P} times with a $(+1)$ label, n_i^{F} times with a (-1) label, such that we may feed these processed data into a GP classifier. However, such an approach is not desirable in practice. Firstly, for each training point the predicted pass probability may be very different from the estimated frequentist probability (if $n_i^{\mathrm{P}} + n_i^{\mathrm{F}}$ is large enough, we should observe $p(y_i = +1|X, y, x_i) \approx \tilde{p}_i = n_i^{\mathrm{P}}/(n_i^{\mathrm{P}} + n_i^{\mathrm{F}})$). Secondly, a dataset that includes redundant inputs increases the risk of numerical instability when performing kernel inversion, because the rows and columns of $k(X, X) + \sigma^2 I_n$ may not all be linearly independent.

We propose a more efficient way to process the data by estimating directly the pass probability for each training point. Indeed, the data represent a Bernoulli trial situation, and the probability of passing can be modelled with a Bernoulli distribution of parameter p_i. To estimate this parameter, let us consider the Jeffreys intervals (Brown et. al., 2001 [9]), which provide credible intervals around the frequentist estimation $\tilde{p}_i$ of p_i. The Jeffreys intervals have a Bayesian derivation (using a Beta prior distribution) and account for the fact that too few trials may lead to a lack of confidence in the estimation of p_i. The credible interval over the parameter estimation $\hat{p}_i$ is bounded as:

$$\begin{cases} \mathrm{Low}_i = \beta^{-1}(\alpha/2, n_i^{\mathrm{P}} + 1/2, n_i^{\mathrm{F}} + 1/2) \\ \mathrm{Up}_i = \beta^{-1}(1-\alpha/2, n_i^{\mathrm{P}} + 1/2, n_i^{\mathrm{F}} + 1/2) \end{cases} \tag{11}$$

Where β^{-1} is the Beta distribution CDF inverse, with $\alpha = 0.05$ for 95% confidence. In the following, we will take $\hat{p}_i$ as the mid-distance between the two bounds (*i.e.* $\hat{p}_i = (\mathrm{Low}_i + \mathrm{Up}_i)/2$). Using $\hat{p}_i$, instead of the frequentist estimation $\tilde{p}_i$, helps to avoid an over-confident estimation when there are few trials, and also allows for introducing into the GP the uncertainty over p_i in the form of a Gaussian noise (as detailed in the next section). Figure 1 shows how the Jeffreys intervals converge given n_i^{P} and n_i^{F}. To the extent where $n_i^{\mathrm{P}} + n_i^{\mathrm{F}} \to +\infty$, we have $\hat{p}_i \to \tilde{p}_i$. As a result, we obtain a **regression** dataset of the form $\mathcal{D} = \{X, y = \hat{p}\}$.

3.3 Heteroscedastic Noise

Our data are uncertain, but this uncertainty is well quantified by the Jeffreys intervals. Therefore, we may account for it in our GP regression model. We can pass to the GP a variance term, σ^2, that controls the Gaussian noise that *a priori* corrupts the outputs (equation 1). We achieve this by converting the Jeffreys intervals into a variance term with the following expression:

$$\sigma^2(x_i) = \sigma_i^2 = \left(\frac{\mathrm{Up}_i - \mathrm{Low}_i}{2\delta}\right)^2 \tag{12}$$

If we chose 95% confidence for the Jeffreys intervals, then $\delta = 1.96$. The Jeffreys intervals typically become wider if p_i is close to 0.5, and the number of experimental tests available for each input may be different (fewer Bernoulli trials yields larger uncertainty). Therefore, the variance term is input-dependant (*heteroscedasticity*). In heteroscedastic GP regression, when the variance in the data is known (which is the case here) the inference can be performed exactly and the covariance matrix of $p(y|f)$ becomes $\mathrm{diag}(\sigma_1^2, \ldots, \sigma_n^2)$ instead of $\sigma^2 I_n$.

3.4 Kernel Selection

The probability that a component will pass a certification test is expected to vary smoothly with respect to the input features. Indeed even when mechanical failure happens suddenly and the pass probability undergoes fast variations, there is no physical reason to expect kinks in the probability of passing. As a result, we propose to use an RBF kernel (equation 3). The RBF kernel can approximate $\mathscr{C}^\infty$ functions and is therefore appropriate for modeling smooth physical phenomenon.

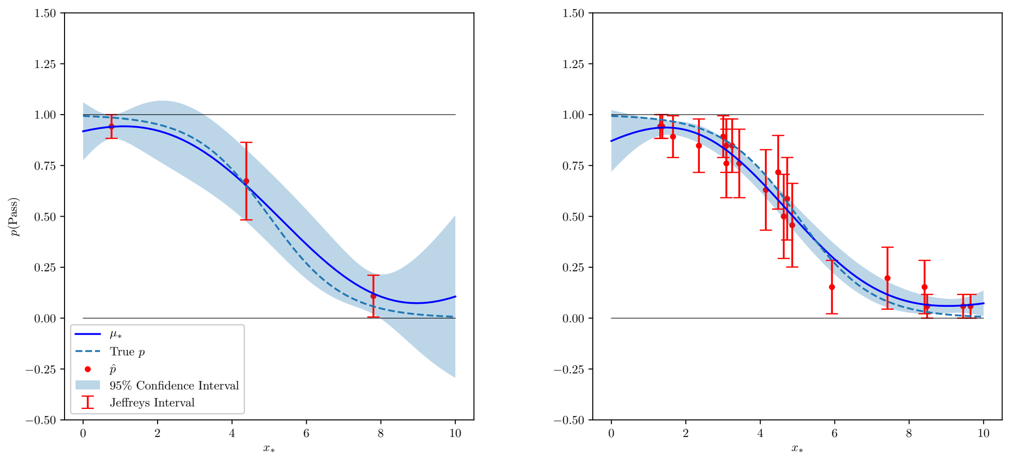
Figure 2 shows an example of synthetic 1D data fitting with the RBF kernel. When the data points are sparse (figure 2a) and far from one another, the GP relies heavily on the *a priori* variance that we passed to it (from the Jeffreys intervals), and at each training point the GP confidence intervals closely match the Jeffreys intervals. When more data are available (figure 2b), the GP can more easily estimate the true underlying patterns in the data (moving beyond the a priori Jeffreys intervals) and discriminate outlier data points from credible data points, yielding greater prediction confidence. Notice that in both cases, the true probability p falls almost always into the 95% confidence interval of the GP. As a result of the foregoing arguments and results, we adopt an RBF kernel in our work.

3.5 Bounded GP Regression

The method outlined in this section is intended to be used with highly uncertain datasets, where encountering a component with a high probability of either passing or failing should remain rare. However, since the method is based on the standard (unbounded) GP regression framework, there is always a risk of predicting a probability greater than 1 or lower than 0, which is mathematically inconsistent. This may occur if the dataset contains a significant number of data points with most, if not all, the tests having either passed or failed. Making predictions in these corresponding input regions of high certainty may result in outputs outside of the interval $[0, 1]$. This caveat can be easily overcome by using a *bounded* GP regression (Jensen et. al. 2013 [10]). The key idea is to introduce a warping function to constrain the latent function in the interval $[0, 1]$, (*e.g* using a standard normal CDF Φ), as shown in equation 13. Note that we still remain within regression framework and the likelihood is still Gaussian.

$$\begin{cases} y_i = \Phi(f_i) + \epsilon_i \\ \epsilon_i \sim \mathcal{N}(0, \sigma_i^2) \end{cases} \tag{13}$$

$$p(y|f) = \mathcal{N}(y|\Phi(f), \mathrm{diag}(\sigma_1^2, \ldots, \sigma_n^2)) \tag{14}$$



(a) 3 Training Points

(b) 20 Training Points

Figure 2: 1D example of probability fitting using an RBF kernel. In this synthetic example, $\hat{p}_i$ and the Jeffreys intervals are computed based on the procedure outlined in section 3.2, using for each input point 20 Bernoulli trials $s \sim p_i^s(1-p_i)^{1-s}$, where the true underlying probability is $p_i = 1/(1+\exp(-x_i+5))$. For representation purposes, we assume $x \in \mathbb{R}^1$.

Introducing a non-linear function to warp the mean of the likelihood renders the MLL analytically intractable; but can be easily approximated (like in classification). The predictive mean μ_* and variance Σ_* may also be approximated and warped back into Φ to obtain regression prediction bounded in the interval $[0,1]$. In the following developments, we will rely on Laplace method to approximate the MLL (Rasmussen & Williams, 2006, [4]), which is both easy to implement and a natural choice here. Indeed, the marginal likelihood integrand is a product of Gaussian probabilities and the Laplace method hinges on Gaussian approximation.

4 GP Classification on Synthetic or Experimental Data

In this section, we introduce a GP classification method for estimating the probability that a new mechanical component will pass certification tests using datasets where *only one test outcome per input is available.*

4.1 Dataset

We consider the case where each input component has been tested only once. The method detailed shortly is typically intended for synthetic datasets (numerically-generated), because numerical simulation tools like standard finite elements are deterministic and make repeated testing irrelevant. However, this method may also be used for experimental data where only one test was performed for each input (*e.g.* when testing is expensive and/or time consuming). Subsequently, each output is a binary label (-1 for a failed test and $+1$ for a passed test), and the dataset is directly suitable for GP classification.

4.2 Kernel Selection

We want our model to provide good labelling prediction accuracy (either pass or fail), but perhaps more importantly, we want it to output a probability of passing/failing that is consistent with engineering intuition. In GP classification, warping the latent function with a normal CDF has the effect of introducing smoothness and non-linearity, even if the chosen kernel doesn't naturally have these properties. Hence, we can use a kernel that describes the underlying physics in the data more closely and let the normal CDF take care of the non-linearity. The datasets on which this model is intended to be used represent a snapshot of solid and structural mechanics behaviors. Mechanical failure occurs when stresses or displacements exceed a critical value, which itself happens when one of the input features exceeds a critical threshold (*e.g.* elastic limit, failure load, *etc.*). Besides, once one or several of these thresholds have been reached, failure will occur in a more and more dramatic way, as we keep moving further beyond these thresholds. For example, consider a cantilever beam with a fixed point load at the free end, and we test whether or not the beam starts yielding for different beam lengths. The probability of passing should always decrease as we test longer and longer beams. Therefore, we expect a monotonic trend. Thus, we want our classification model to reflect it.

We propose using a linear kernel (equation 15). A GP that relies on such kernel is analogous to a linear regression with 1st order features, yielding a linear predictive mean. Since the mean is warped with a standard normal CDF, which is also monotonic, the probability of passing will have the required overall monotonicity. The linear kernel takes the form:

$$k_{\text{LIN}}(x_i, x_j) = \theta_1 x_i \cdot x_j + \theta_2 \tag{15}$$

Figure 3 illustrates the cantilever beam example (where $x \in [0, 10]$ is the overall beam length). With a linear kernel, we have $p(y_* = +1|X, y, x_* = 5) \approx 0.5$. This is intuitively satisfying, because $x = 5$ is the critical value below which all training labels are passing, and beyond which all training labels are failing. Hence, for this specific input we cannot make a confident call about the class label it belongs to. Moreover, we obtain the desirable monotonic trend as the shortest beams are the most likely to pass and the longest beams are the most likely to fail. Conversely, with an RBF kernel the model behavior is somewhat different. The pass probability show modest decrease as $x_* \to 0$ and increases as $x_* \to 10$, which is physically inconsistent. This is a classical artifact of the RBF kernel. Typically, a latent function based on an RBF kernel tends to the prior mean value in regions with fewer training data points, or when extrapolating outside of the original training data boundaries ($p(y_* = +1|X, y, x_* \to \pm\infty) \to \Phi(0) = 0.5$); see figure 3b.

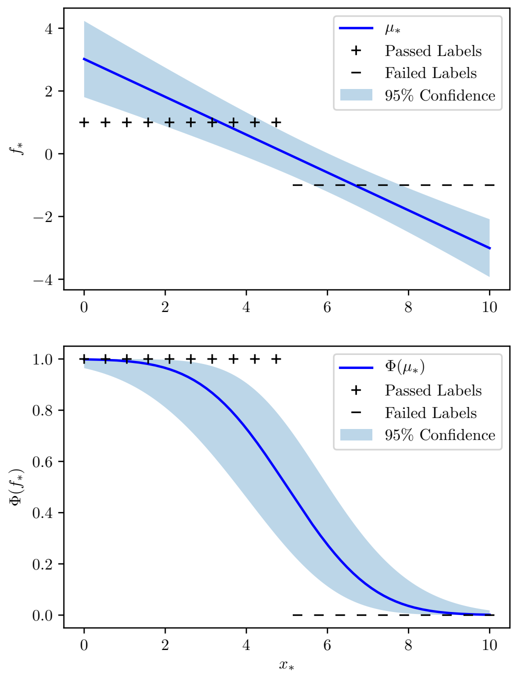
Note that we propose to use the linear kernel purely for its monotonic properties, and not to describe physical linearity. This model is intended to be suitable even if the underlying mechanics in the data are non-linear (*e.g.* in case of hyperelastic or plastic behavior).

5 Computer Implementation

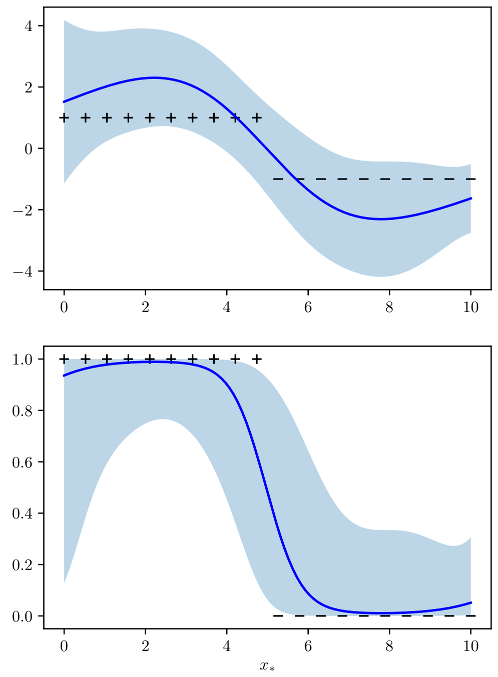
To implement our standard regression and classification models, we use `GPyTorch` (Gardner et. al., 2018, [11]), a GP Python library built on top of the deep learning framework `PyTorch` (Paszke et. al., 2017, [12]). `GPyTorch` is a flexible library: it is scalable to large datasets, and already includes variational inference for approximating the MLL in GP classification (Blei et. al. 2017 [8]). To implement our bounded regression model, we use our own GP code and rely on Laplace approximation to approximate the MLL. To minimize the negative MLL and obtain the GP hyperparameters, we will rely on the Adam optimizer (Kingma & Ba, 2014, [13]). Adam is a robust and highly efficient algorithm based on gradient-descent, where the hyperparameters are learned iteratively by subtracting the MLL gradient from the current value of θ (equation 16).

$$\theta_{t+1} = \theta_t - \alpha \nabla(-\log p(y|X, \theta_t)) \tag{16}$$

With Adam in particular, the arbitrary learning rate α is adaptive, and decreases as the optimizer progresses towards the minimum (which helps to prevent overshooting it), and the gradient is averaged over successive training iterations.



(a) Linear Kernel



(b) RBF Kernel

Figure 3: Comparison of model behaviors for the linear kernel and the RBF kernel. The upper plots represent the latent prediction f_* and the lower plots shows the warped predictions $\Phi(f_*)$ for probability of passing. For representation purposes, we assume $x \in \mathbb{R}^1$.

6 Results and Discussion

The present section demonstrates the proposed GP approaches to useful engineering contexts.

6.1 Application to Synthetic Data

6.1.1 Dataset

In this section, we consider an application to shock predictions for rack mounted electronic components, referred to as *cabinets*. The dataset reflects the behavior of a 7-level cabinet for different mechanical parameter inputs (figure 4). The associated response dataset is made of 873 data points generated from linear-elastic (deterministic) finite element simulations. Forced transient vibrations are induced at the base of the cabinet, and the resulting velocities, v_i, of each level are recorded. For each transient response, the shock response spectrum (SRS) is calculated (a standard measure of mechanical response). If these SRS spectrums exceed given threshold curves, the test is failed (we consider two pass/fail criteria, referred in the foregoing as *NATO* and *NoDev*). Consequently, for each cabinet there are 14 different pass/fail outputs (2 per level). Note that it is possible for a given cabinet to have some levels that pass according to one criteria, and fail according to the other. It is also possible to have some levels failing and others passing for the same cabinet simultaneously. The only varying features are the forced vibration frequency (8Hz, 14Hz, or 25Hz), and the mass of each level ($m_i \in [0, 36]$, noting that the masses add up to 36 lbs). Every other parameter is common to all cabinets (*e.g.*, elastic modulus, dimensions, *etc.*), and hence has no influence on the test outcome. Therefore we may neglect them, such that our input points contain 8 features in total (1 frequency and 7 masses). We also have 14 binary outputs associated with each levels/criteria that we will treat independently.

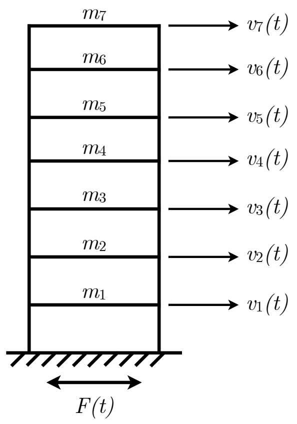


Figure 4: Cabinet Representation

Level	1	2	3	4	5	6	7
NATO	0.33	0.77	0.77	0.77	0.77	0.77	0.72
NoDev	0.66	0.78	0.77	0.78	0.78	0.77	0.72

Table 1: Ratio of passed labels in the dataset for each level and criteria

Note that our dataset is unbalanced, and for each level/criteria the outputs are not exactly split between 50% passed and 50% failed. The exact ratio of passed labels is given in Table 1.

6.1.2 Training

The dataset is randomly shuffled and split between a training set (80% of the data) and a test set (remaining 20%). Note that using a cross-validation set is less relevant in the current case than it may be in frequentist machine learning approaches (*e.g.* deep learning), because overfitting is typically not an issue within a Bayesian setting, and model selection here depends on our prior beliefs. Indeed, in the Bayesian paradigm, rather than using additional data for model selection, we may use it to update our current model instead (Neal, 1995 [5]). The data are also normalized: each feature has its mean value subtracted and the result divided by its standard deviation. We train 14 independent GP classifiers for each level/criteria with a linear kernel, an RBF kernel, and a combined RBF$\times$Linear kernel. For baseline comparison purposes only, we also train a fully connected deep neural network (2 hidden layers of 200 units with ReLU activation). For the optimization with Adam, we consider $\alpha = 0.1$ for the initial learning rate and 100 iterations.

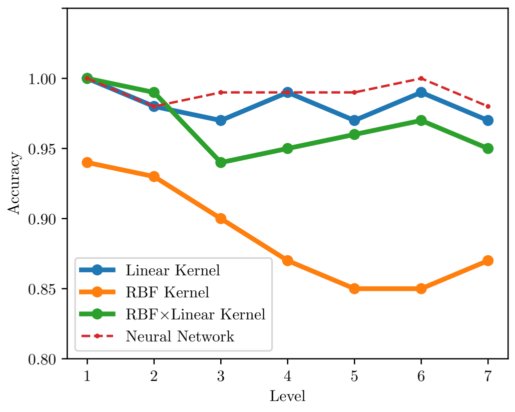
The training was performed on a mid-2010 MacPro, with dual 6-core 2.66GHz Intel Xeon CPUs, and 64Gb ram. Each GP took on average about one minute to train. The size of our training set is relatively small (698 points) so computational expenses are not an issue in the present case. However, the training process requires an inversion of the covariance matrix, which takes up to $\mathcal{O}(n_{\text{train}}^3)$ computations. Hence for large datasets ($n_{\text{train}} > 10,000$), the use of scalable kernel approximation methods may be desirable to speed up computations (Wilson & Nickisch, 2015 [14]).

6.1.3 Preliminary Results

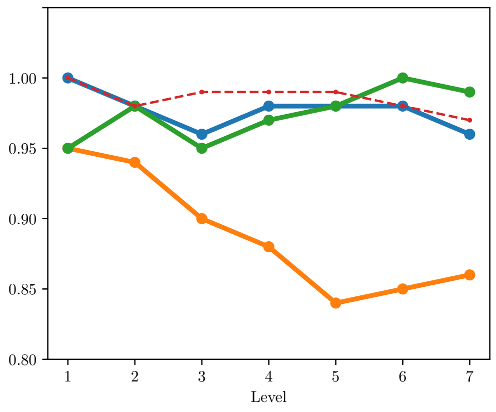
The test set accuracy for each model and level/criteria is shown in table 2 and figure 5. The accuracy is measured as the number of correctly labeled test inputs over the total number of points in the test set. Note that in this paper, we set the labeling threshold probability at 0.5 (*i.e.* x_* is predicted to pass the certification test if $p(y_* = +1|X, y, x_*) > 0.5$, and fail if $p(y_* = +1|X, y, x_*) \leq 0.5$).

Label	Linear Kernel	RBF Kernel	RBF×Linear Kernel	Neural Network
Level 1 - NATO	1.00	0.94	1.00	1.00
Level 1 - NoDev	1.00	0.95	0.95	1.00
Level 2 - NATO	0.98	0.93	0.99	0.98
Level 2 - NoDev	0.98	0.94	0.98	0.98
Level 3 - NATO	0.97	0.90	0.94	0.99
Level 3 - NoDev	0.96	0.90	0.95	0.99
Level 4 - NATO	0.99	0.87	0.95	0.99
Level 4 - NoDev	0.98	0.88	0.97	0.99
Level 5 - NATO	0.97	0.85	0.96	0.99
Level 5 - NoDev	0.98	0.84	0.98	0.99
Level 6 - NATO	0.99	0.85	0.97	1.00
Level 6 - NoDev	0.98	0.85	1.00	0.98
Level 7 - NATO	0.97	0.87	0.95	0.98
Level 7 - NoDev	0.96	0.86	0.99	0.97

Table 2: Classification accuracy of the different GP models and the neural network. The accuracy is defined as the number of correctly labelled input points (from the test set) over the total number of points in the test set. An accuracy of 1.00 mean that 100% of the predictions on the test set were correct.



(a) NATO criteria



(b) NoDev criteria

Figure 5: Classification accuracy with respect to each level, for the two criteria

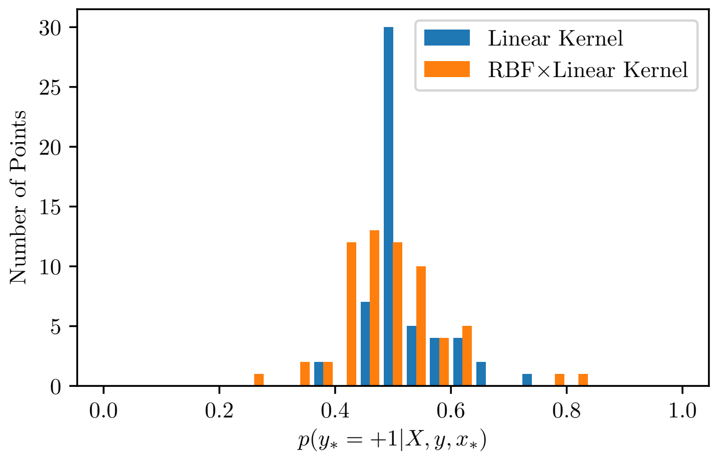


Figure 6: Histogram of the predicted probability for each test points that were incorrectly classified (over the 14 outputs). For example, it happened once that the RBF×Linear kernel predicted a probability of passing greater than 0.8, when in reality the input point would have failed the certification test.

The linear kernel provides very good accuracy and achieves nearly the same accuracy as the neural network. This was expected because as previously mentioned, it is the kernel that matches intuitions about the underlying physics the best. It clearly outperforms the RBF kernel and tends to perform better than the combined RBF×Linear kernel on most level/criteria. However, the latter is actually slightly more accurate than the simple linear kernel for 3 of the outputs (Level 2 – NATO, Level 6 – NoDev and Level 7 – NoDev). While it could be tempting to use the RBF×Linear kernel on these specific labels, instead, and a linear kernel for the rest, introducing an RBF kernel causes the data boundary effect described in figure 3b, which is inconsistent with engineering intuition.

Therefore, there is a tradeoff between using the simple linear kernel and the RBF×Linear kernel. If we are interested only in the correct labeling prediction (test pass or test failed), we should certainly choose the kernel that is the most accurate for each label (but in this case, we might as well choose a neural network or simply rely on finite element simulations if we are not interested in uncertainty considerations). However, if we are interested in a physically interpretable model, with meaningful pass/fail probabilities, we should rather choose the linear kernel, as it better reflects the underlying physics within the cabinet data. Note that in figure 5, the accuracy decreases for higher levels, especially with the RBF and the RBF×Linear kernel. This can be explained by the fact that these levels are coincidentally the most unbalanced ones, and unbalanced datasets negatively impact learning performance.

It is also insightful to take a closer look at the pass/fail predictive probabilities when the model is making a classification error. Figure 6 shows a histogram of the pass probability for all the test points that were misclassified (over the 14 labels) for the linear kernel and the RBF×Linear kernel. For the linear kernel, the pass probability is concentrated around 0.5, indicating that the model has a low confidence when it makes wrong predictions. For the RBF×Linear kernel, the pass probability for misclassified labels has a larger variance and can more easily output extreme probability. In other words, the RBF×Linear kernel can make mistakes with a higher confidence, which is definitely not desirable (when it is wrong, it can be *very* wrong).

Note that the neural networks are only used to provide a baseline comparison standpoint for our GP models. For practical use, neural networks would be of little interest here because they don't provide suitable confidence intervals, and as non-Bayesian models they cannot be easily used to encode prior knowledge about the data.

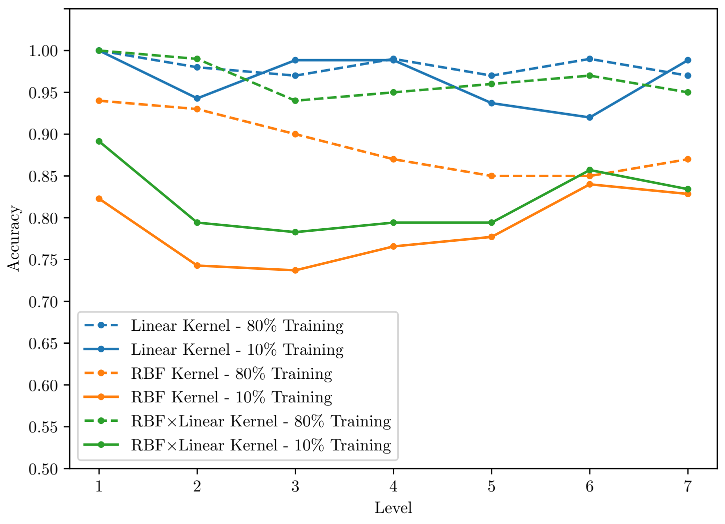


Figure 7: Accuracy comparison between model trained on 80% and 10% of the dataset for the NATO criteria

6.1.4 Robustness to Sparse Data

We investigate the robustness of our model given the size of the dataset. We retrain our 3 GP models on the NATO criteria outputs, but with only 10% of the dataset this time, and assess the prediction accuracy on the same test set as earlier. Figure 7 shows the accuracy when relying on only 10% of the data, compared to the orginal accuracy shown in figure 5a. The linear kernel still performs very well, and while there is a slight drop in accuracy for some levels (2, 5 and 6), this drop is not dramatic (the worst drop is only by 7% for level 6). Moreover, the other levels yields as good results as before and some (3 and 7) are even slightly better. Conversely, the RBF kernel and the RBF ×Linear kernel are subject to a much more significant drop in accuracy (up to 19% and 20% respectively), and are now both completely outperformed by the linear kernel. This indicates that the RBF kernel and the RBF×Linear kernel are highly sensitive to the amount of data available, unlike the linear kernel. This is easily explainable by the fact that the linear kernel already describes well the underlying physics in the data, and hence only need very few training points to correctly tune the hyperparameters.

6.1.5 Robustness to Mislabeled Data

Let us now consider the effect of mislabeled outputs in the training data. The data used so far are numerically-generated and purely deterministic, as each point has only one possible label. However, if the dataset had been based on experiments and only one test had been performed per input, it is likely that some outputs would have been labeled in a very random manner (*e.g.* if the true probability of passing for a component is close to 0.5, a single certification test is not trustworthy and may lead to unreliable output labeling). To investigate the effects of uncertain or mislabeled outputs in the dataset, we again consider the cabinet data, but we corrupt it by randomly flipping 20% of the outputs in the training set (*i.e.* we randomly pick 20% of the training data, for which we flip the passed label into a failed label or *vice-versa*). We then test the accuracy of our models on a the same (non-corrupted) test set as in section 6.1.3 and 6.1.4. Note that in this section, we consider the full training dataset (like in section 6.1.3).

Figure 8 shows the accuracy results on the test set for the three different kernels, trained on both the original (not corrupted) and the corrupted training set for the NATO criteria. Again, the linear kernel gives excellent and stable accuracy. The worst drop in accuracy is only by 4% for level 3, and the other levels maintain similar accuracy as with the non corrupted training set. We observe a much more significant drop for the RBF and the RBF×Linear kernel, where the worst drops are 15% and

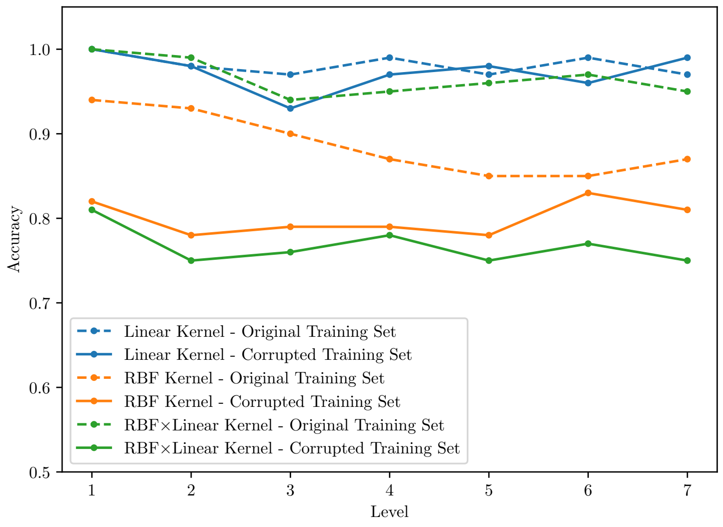


Figure 8: Accuracy comparison between model trained on the original (sane) training set and the corrupted training set for the NATO criteria

24% respectively. These results are not surprising, because the RBF kernel can vary more easily than the linear kernel, and thus accommodate input regions where the labels are changing quickly (which is not desirable in the present case, since these regions most likely correspond to mislabeled outputs). Conversely, it would take a colossal amount of mislabeled data points, located in nearby input regions, to significantly mislead the linear kernel.

6.2 Application to Experimental Data

6.2.1 Dataset

Our GP regression model has been applied to an experimental dataset, but due to restrictions on the ownership of these data, we are unable to present results obtained with it in this paper. Consequently in the following section, we will show results obtained with a dataset generated numerically and contaminated with realistic random noise, to simulate experimental uncertainty (*cf.* appendix for more details).

We consider a cabinet dataset similar to the one presented in section 6.1, except that we now have 5 levels. On each level a random mass is set (such that the total mass over all the levels adds up to 100 kg). A sinusoidal displacement is enforced at the base of the cabinet, and the acceleration of each level is recorded during 1 second. For each cabinet, the experiment is repeated 20 times, but given the uncertainty in the accelerometer measurements, the maximum acceleration recorded may be different for each of the 20 experiment. If the maximum acceleration is above a critical threshold ($a_c = 0.00726$ m/s^2), the test is failed and we therefore obtain a set of 20 pass/fail outcomes for each set of input features. The features considered are the masses of each of the 5 levels (the geometric and mechanical parameters and the displacement enforced at the base are the same for every cabinet).

The training dataset contains 50 points, which is a feasible number had the data been generated in real experimental settings. The test dataset contains 1000 points and each point was tested 4000 times. Doing so furnishes very good estimations of the ground truth probabilities of passing in the test set, and therefore yields a quasi-noiseless test set. Hence we can better assess how well our regression model recovers the true probability of passing given a very limited number of trials in the training set.

6.2.2 Training

We train several GP models: A standard GP regression and a bounded GP regression with an RBF kernel and a Matérn kernel (Rasmussen Williams, 2006, [4]). Note that we train these models with both homoscedastic noise and heteroscedastic noise, to verify that the introduction of heteroscedasticity helps to achieve better accuracy. The Matérn kernel is similar to an RBF kernel but one can tune its smoothness, given a parameter ν. For $\nu = 0.5$ the Matérn kernel is equivalent to a pointwise linear interpolator and for $\nu \to +\infty$, $k_{\text{Matérn}} \to k_{\text{RBF}}$. In the following, we will take $\nu = 2.5$, which provides smoothness while making the GP capable of modelling small kinks in the data. For baseline comparison purposes, we also train a standard linear regression, a neural network regression (2 hidden layers of 200 units with ReLU activation) and a support vector machine regression with an RBF kernel. For the optimization with Adam, we consider $\alpha = 0.1$ for the initial learning rate and 200 iterations.

The training was performed with the same equipment described in section 6.1.2. The standard GP models took a few seconds to train each, and the bounded GP models took about a minute each (bounded GPs take longer as each optimization step requires an evaluation to approximate the MLL with the Laplace approach, which itself requires a full Newton-Raphson loop, as per Rasmussen & Williams, 2006 [4]).

6.2.3 Results

For each model, the Root Mean Squared Error (RMSE), the Mean Absolute Error (MAE), and the r^2 value obtained on the test set are shown in table 3. By all metrics, the bounded heteroscedastic GP regression with an RBF kernel provides the best accuracy (RMSE $= 0.0398$), closely followed by the same model with a Matérn kernel (RMSE $= 0.0438$). They both clearly outperform the three baseline comparison models as well as the other GP regressions. The standard heteroscedastic GP regression models also provide satisfactory results (RMSE $= 0.0834$ and 0.0843 for the RBF and the Matérn kernel respectively). The heteroscedastic bounded GP regressions are also the models yielding the smallest difference between the MAE and the RMSE, suggesting that the standard deviation within the absolute error distribution is reduced. Hence, they are less likely to make *very* wrong predictions.

While heteroscedastic GP regressions provide good if not excellent accuracy, this accuracy drops significantly when switching to arbitrary homoscedastic noise. In this later case, the baseline comparison models overall perform better than the GP regressions. Such observation points out the appropriateness of heteroscedasticity: Introducing prior knowledge about the uncertainty in the dataset allows for achieving significantly better accuracy than homoscedastic models (including non-Bayesian).

With the heteroscedastic GPs (bounded and standard), the RBF and the Matérn kernel both provide quasi-identical accuracy, which is no surprise because the two kernel are very similar. However, the Matérn kernel tends to perform slightly worse. A possible explanation is that the available training datas are only a partial snapshot of the underlying physics, and some input regions with sparse data points may wrongly imply discontinuity and non-smoothness in the physics. Since the Matérn kernel can more easily capture kinks, it is more prone to closely fit these troublesome input regions. Therefore, the slight difference in accuracy between the two kernels confirms our prior assumption made in section 3.4.

7 Conclusion

Our goal was to demonstrate non-physical but mathematically rigorous data-driven machine learning models in the context of mechanical test and failure prediction. We have presented two models based on Gaussian processes for estimating the probability that mechanical components will pass certification tests. In the first model, we used standard and bounded GP regression applied to experimental datasets. We demonstrated that estimating the uncertainty in the training data using confidence intervals and then introducing this uncertainty into the GP regression framework using heteroscedasticity yields better prediction accuracy than common frequentist machine learning

Model	**RMSE**	**MAE**	r^2
Standard GPR (RBF & Heteroscedastic Noise)	0.0834	0.0581	0.915
Standard GPR (RBF & Homoscedastic Noise $\sigma^2 = 10^{-3}$)	0.1502	0.1219	0.724
Standard GPR (Matérn 2.5 & Heteroscedastic Noise)	0.0843	0.0617	0.913
Standard GPR (Matérn 2.5 & Homoscedastic Noise $\sigma^2 = 10^{-3}$)	0.1227	0.0980	0.816
Bounded GPR (RBF & Heteroscedastic Noise)	**0.0398**	**0.0340**	**0.981**
Bounded GPR (RBF & Homoscedastic Noise $\sigma^2 = 10^{-3}$)	0.1482	0.1168	0.730
Bounded GPR (RBF & Homoscedastic Noise $\sigma^2 = 10^{-6}$)	0.1694	0.1376	0.649
Bounded GPR (Matérn 2.5 & Heteroscedastic Noise)	0.0438	0.0368	0.976
Bounded GPR (Matérn 2.5 & Homoscedastic Noise $\sigma^2 = 10^{-3}$)	0.1167	0.0908	0.833
Bounded GPR (Matérn 2.5 & Homoscedastic Noise $\sigma^2 = 10^{-6}$)	0.1300	0.1035	0.794
Linear Regression	0.1019	0.0889	0.873
Neural Network	0.1049	0.0800	0.866
Support Vector Regression (RBF)	0.1223	0.0925	0.817

Table 3: RMSE, MAE and r^2 value of the different GP models and the baseline comparison models. The RMSE indicates how far each predicted output is from the true value, on average, with an increasing penalty for the most inaccurate predictions. RMSE $= \left(\frac{1}{n}\sum_i(\mu_{*i} - y_i)^2\right)^{1/2}$. The MAE indicates the average absolute error between each predicted output and the true value. MAE $= \frac{1}{n}\sum_i |\mu_{*i} - y_i|$. The r^2 is a coefficient between 0 and 1 that measures how well the model explains the test set. $r^2 = 1 - \left(\sum_i(\mu_{*i} - y_i)^2 / \sum_i(\langle y\rangle - y_i)^2\right)$, with $\langle y\rangle = \frac{1}{n}\sum_i y_i$.

methods. In particular, we showed that bounded heteroscedastic GP regression with an RBF kernel is very well suited in the context of structural dynamics test validation, as it provides both excellent accuracy (RMSE $= 0.0398$, $r^2 = 0.981$) and mathematical consistency.

In the second model, we used GP classification applied to surrogate datasets. We demonstrated that using a linear kernel is well suited for describing the monotonic trend in certain underlying physics, and helps in achieving the best accuracy among other GP and frequentist models (we obtained between 96% and 100% classification accuracy). We also demonstrated that the high accuracy achieved with the linear kernel still holds even with significantly fewer data available and with highly corrupted training labels.

8 Acknowledgments

The authors acknowledge valuable conversation with NAVSEA engineers at NSWC Carderock, especially Domenic Urzillo, Randall Goodnight, Brian Lang, and Jacob Mason.

References

[1] N.Tandon, V.V.P.Rao, V.P.Agrawal. *Vibration and noise analysis of computer hard disk drives*, Measurement. 39. 16-25, 2006.

[2] K. Aggarwal, Pravin. *Dynamic (Vibration) Testing: Design Certification of Aerospace System*, `https://doi.org/10.1002/9780470686652.eae179`, 2010.

[3] Department of Defense, *MIL-S-901D, Shock Tests, H.I. (High Impact), Shipboard Machinery, Equipment and Systems, Requirements for*, 1989.

[4] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*, MIT Press, 2006.

[5] Radford M. Neal. *Bayesian Learning for Neural Networks*, PhD Thesis, University of Toronto, 1995.

[6] Carl Edward Rasmussen and Zoubin Ghahramani. *Occam's razor*, Advances in Neural Information Processing Systems 13, pages 294-300, MIT Press, 2001.

[7] Hannes Nickisch and Carl Edward Rasmussen. *Approximations for Binary Gaussian Process Classification*, Journal of Machine Learning Research, 9, 2035-2078, 2008

[8] David M. Blei, Alp Kucukelbir, Jon D. McAuliffe. *Variational Inference: A Review for Statisticians*, Journal of the American Statistical Association, Vol. 112 , Iss. 518, 2017

[9] Brown, L.D., Cai, T.T., DasGupta, A. *Interval Estimation For a Binomial Proportion*, Statistical Science,16, 101–133, 2001

[10] Bjørn Sand Jensen, Jens Brehm Nielsen, Jan Larsen. *Bounded Gaussian process regression*, IEEE International Workshop on Machine Learning for Signal Processing (MLSP), 1-6, 2013.

[11] Gardner, Jacob R and Pleiss, Geoff and Bindel, David and Weinberger, Kilian Q and Wilson, Andrew Gordon. *GPyTorch: Blackbox Matrix-Matrix Gaussian Process Inference with GPU Acceleration*, Advances in Neural Information Processing Systems, 2018

[12] Paszke, Adam and Gross, Sam and Chintala, Soumith and Chanan, Gregory and Yang, Edward and DeVito, Zachary and Lin, Zeming and Desmaison, Alban and Antiga, Luca and Lerer, Adam. *Automatic differentiation in PyTorch*, NIPS, 2017

[13] Diederik P. Kingma, Jimmy Ba. *Adam: A Method for Stochastic Optimization*, Proceedings of the 3rd International Conference on Learning Representations (ICLR), 2014

[14] Andrew Gordon Wilson, Hannes Nickisch. *Kernel interpolation for scalable structured Gaussian processes (KISS-GP)* Proceedings of the 32nd International Conference on Machine Learning (ICML), 1775-1784, 2015

A Appendix on Generating the Data Used for Testing the GP Regression Model in section 6.2

A.1 Introduction

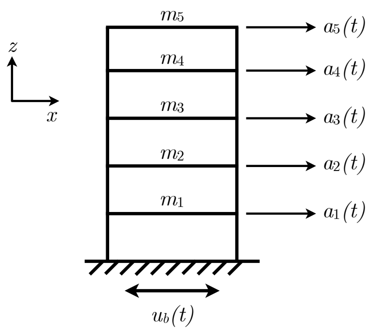
In this appendix we provide further details on numerically generating data that are representative of experimental testing of mechanical components conducted in real-life conditions. These data are then used for assessing the robustness and accuracy of our GP Regression model in section 6.2 of the present paper.

A.2 Experiment Description

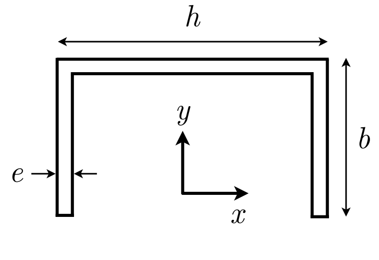
We consider a cabinet made of aluminum, with 5 vertically equispaced levels (or shelves), as shown in figure 9a. On each level, an arbitrary mass is set (adding up to a total of 100 kg over the 5 levels). We consider prismatic U-section beams to be present between the levels (figure 9b). A sinusoidal displacement is applied at the base of the cabinet (in the x-axis direction), and we compute the response acceleration of each levels during 1 second. For generating the training set, we consider 50 cabinets where the 100kg total mass has been randomly distributed over the 5 levels. The base displacement and the geometric and mechanical parameters of the cabinet remain the same for each point generated. Consequently, each raw data point has 5 input features (the mass set on each level), and the corresponding output is the maximum acceleration value recorded over 1 second (across all the levels). All the experiment parameters are summarized in table 4.

Parameter	Notation	Unit	Value
Aluminum Elasticity Modulus	E	GPa	70
Aluminum Density	ρ	kg/m^3	2700
Cabinet Height	L	m	2
Cabinet Width	h	m	1
Cabinet Depth	b	m	0.5
Cabinet Wall Thickness	e	m	0.04
2nd Moment of Inertia	I_y	m^4	0.00775
Cross Section Area	A	m^2	0.0584
Base Displacement	$u_b(t)$	m	$0.1\sin(16\pi t)$

Table 4: Mechanical and geometric parameters of the cabinets.



(a) Cabinet - Front representation



(b) Cabinet - Horizontal cross section

Figure 9: Representation of the experiment and the cabinet geometry. Note that $\sum_{i=1}^{5} m_i = 100$ kg, the height of the cabinet is $L = 2$ m, and $e = 0.04$ m, $b = 0.5$ m, $h = 1$ m.

A.3 Mass-Spring System for Generating Level Responses

The cabinet is modeled as a standard mass-spring system with 5 degrees of freedom. We neglect the damping effects. The system of equations is given by (Chopra, 2012 [1]):

$$\mathbf{M}\ddot{\mathbf{u}} + \mathbf{K}\mathbf{u} = -\mathbf{M}\mathbf{1}\ddot{u}_b(t) \tag{A.17}$$

Where M and K are respectively the standard mass and stiffness matrices associated with a MDOF in series and $\mathbf{1}$ is a vector of length 5 with unit components. Note that we take into account the mass of the cabinet itself, so the diagonal terms of M are $[M]_{ij} = \frac{\rho AL}{5} + m_i$ (where m_i is the random mass set on the level i, sampled from a uniform distribution). In K, we have $[K]_{ii} = 2k$ and $[K]_{i,i+1} = [K]_{i+1,i} = -k$, with $k = \frac{12 \cdot EI_y}{(L/5)^3}$. The displacements are solved analytically and we use them to compute the maximum acceleration across each level during 1 second of shaking.

```
initialization
$\mathbf{a^{max}}$ = vector containing the maximum computed acceleration of each cabinet;
$a_c = 0.00726$;
$n^{\text{trial}} = 20$;
$n^{\text{cabinet}} = 100$;
$y^{\text{raw}} = n^{\text{cabinet}} \times 2$ empty array;
for $i = 1 \to n^{\text{cabinet}}$ do
    $n_i^{\text{P}} = 0$
    for $j = 1 \to n^{\text{trial}}$ do
        Sample $\epsilon_j \sim \mathcal{N}(0, (0.02 \times \mathbf{a}_i^{\mathbf{max}})^2)$
        if $\mathbf{a}_i^{\mathbf{max}} + \epsilon_j < a_c$ then
            $n_i^{\text{P}} = n_i^{\text{P}} + 1$
        end if
    end for
    $y_{i,0}^{\text{raw}} = n_i^{\text{P}}$
    $y_{i,1}^{\text{raw}} = n^{\text{trial}} - n_i^{\text{P}}$
end for
```

Algorithm 1: Algorithm for contaminating the cabinet accelerations with noise

A.4 Noise Contamination

To make the dataset generated above representative of an experimental dataset, we assume that the level accelerations were measured experimentally using accelerometers. Even high quality accelerometers present irreducible uncertainty in their measurement outputs. This uncertainty varies given multiple factors (acceleration frequency, temperature, *etc.*), but for MEMS accelerometers, a realistic approximation is to model the noise in the measurement as a Gaussian, with a standard deviation scale factor between 0.05% and 3% of the quantity measured (Martin et. al., 2016 [2]). For example, if the true measurement is 100 and the scale factor is 2%, one can assume the noise to be $\epsilon \sim \mathcal{N}(0, (0.02 \times 100)^2)$. For each cabinet, we contaminate the computed acceleration n^{trial} times with Gaussian noise (with a 2% scale factor) in order to simulate repeated independent measurements (we assumed that each cabinet was tested $n^{\text{trial}} = 20$ times). If a contaminated acceleration exceeded an arbitrary threshold (set to $a_c = 0.00726$ m/s^2), the corresponding test is failed. Consequently, for each cabinet, we obtain a set of 20 Bernoulli trials outcomes suitable for our GP regression model. The noise contamination routine is summarized in algorithm 1.

Note that we assume 20 repeated tests for each cabinet, for generating the GP training set, because it corresponds to a number of experimentation that is reasonably feasible in real settings. However, for generating the GP test set we assume 4000 tests per cabinet. Indeed with such a high number of trials, we can better assess if the predictions of our GP model are close to the *ground truth* probability of passing.

References

[1] Chopra, Anil K. *Dynamics of Structures: Theory and Applications to Earthquake Engineering*, Pearson Higher Education, 2012.

[2] Martin, H., Groves, P., and Newman, M. *The Limits of In-Run Calibration of MEMS Inertial Sensors and Sensor Arrays*, J Inst Navig, 63: 127– 143, 2016

***Conflict of Interest Statement**

Declaration of interests

☒ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: