Measuring the sensitivity of Gaussian processes to kernel choice

William T. Stephenson\textsuperscript{1,3}, Soumya Ghosh\textsuperscript{2,3}, Tin D. Nguyen\textsuperscript{1,3}, Mikhail Yurochkin\textsuperscript{2,3}, Sameer K. Deshpande\textsuperscript{1,3}, and Tamara Broderick\textsuperscript{1,3}

\textsuperscript{1}Computer Science and Artificial Intelligence Laboratory, MIT
\textsuperscript{2}IBM Research
\textsuperscript{3}MIT-IBM Watson AI Lab

June 14, 2021

Abstract

Gaussian processes (GPs) are used to make medical and scientific decisions, including in cardiac care and monitoring of carbon dioxide emissions. But the choice of GP kernel is often somewhat arbitrary. In particular, uncountably many kernels typically align with qualitative prior knowledge (e.g. function smoothness or stationarity). But in practice, data analysts choose among a handful of convenient standard kernels (e.g. squared exponential). In the present work, we ask: Would decisions made with a GP differ under other, qualitatively interchangeable kernels? We show how to formulate this sensitivity analysis as a constrained optimization problem over a finite-dimensional space. We can then use standard optimizers to identify substantive changes in relevant decisions made with a GP. We demonstrate in both synthetic and real-world examples that decisions made with a GP can exhibit substantial sensitivity to kernel choice, even when prior draws are qualitatively interchangeable to a user.

1 Introduction

Gaussian processes (GPs) enable practitioners to efficiently estimate flexible functional relationships between predictors and outcomes. GPs have been used to monitor physiological vital signs in hospital patients [e.g. 8, 9, 15, 16], to estimate the health effects of exposure to airborne pollutants [e.g. 13, 26, 32], and in many other medical and scientific settings. To use a GP for any application, a practitioner must choose a covariance kernel. The kernel determines the shape, smoothness, and other properties of the latent function of interest [see, e.g., 11 Chap. 2]. Ideally a user would specify a kernel that exactly encodes all of their prior beliefs about the latent function. In practice, a user often has only vague qualitative
prior information and typically selects a kernel from a relatively small set of commonly used families. It seems plausible that other kernels could have been equally compatible with the user’s beliefs. When a user has no reason to prefer one kernel over another given their prior beliefs, we call the kernels qualitatively interchangeable. We would worry, then, if substantive medical or scientific decisions changed when using a qualitatively interchangeable kernel. In this paper, we propose a workflow to assess the sensitivity of the GP posterior under qualitatively interchangeable choices of the kernel.

**Related work.** Choosing a prior at least partly out of convenience is not specific to GPs. There is a rich prior work on quantifying prior robustness in the Bayesian literature [3, 2, 4, 20, 36]. Most of this work has focused on finite-dimensional prior parameters, in contrast to the infinite-dimensional GP kernel. [18, 28] form a notable exception, e.g. showing how to perturb the functional form of parametric priors, including for the beta stick-breaking length in a Dirichlet process. Another option is to try to learn a GP kernel rather than specifying one a priori [1, 12, 37, 38]. But parameter learning and prediction can often be more efficient when prior information is used, so despite these advances, a great many practitioners still directly choose their kernels. [5, 7, 22, 23, 25, 30] have all studied robustness in GPs, but their work focuses on robustness to data perturbations whereas here we focus on robustness to kernel choice. To the best of our knowledge, [35] and [39] are the only other papers to assess sensitivity of GPs to kernel specification. They establish optimal convergence rates for GP regression when the smoothness implied by the selected covariance kernel is misspecified. But they do not study how kernel choice affects functionals like posterior predictive means or quantiles in finite samples, and they do not provide methodology for empirically assessing sensitivity.

**Our contributions.** Given a user-selected kernel, we propose a workflow to discover whether decisions based on a GP posterior are sensitive to the choice of kernel. Our workflow proceeds as follows. (A) Keep expanding a class of appropriate kernels around the original kernel until some kernel in this class yields a substantively different decision. (B) Assess if this decision-changing kernel is qualitatively interchangeable with the original kernel. If the two kernels are interchangeable, we conclude the decision is not robust; a different decision could be reached with the same prior information. If the two kernels are not interchangeable, we cannot conclude non-robustness. We demonstrate our workflow in various applications: (1) predicting whether an adult patient’s resting heart rate exceeds 130 beats per minute (bpm), a potentially dangerous level [14], is sometimes flagged as non-robust by our method and sometimes not; (2) predicted future carbon dioxide levels [31] can be substantively increased with a qualitatively interchangeable kernel; (3) predictions on MNIST handwritten digits can be altered with minor kernel perturbations.

## 2 Our Workflow

**Setup and notation.** Consider data \( D = \{(x_n, y_n)\}_{n=1}^N \), with covariates \( x_n \in \mathbb{R}^D \) and outcomes \( y_n \in \mathbb{R} \). Take a regression model \( y_n \sim N(f(x_n), \sigma^2) \) with \( \sigma^2 > 0 \) and a Gaussian
process (GP) prior on \( f \); we parameterize the GP with mean zero and covariance kernel \( k \):
\[
\mathcal{GP}(0, k).
\]
Let \( k_0 \) be the practitioner-chosen kernel. Typically, \( k_0 \) depends on hyperparameters \( \theta \) (including \( \sigma^2 \)); unless stated otherwise, we assume that \( \theta \) is estimated using maximal marginal likelihood estimation (MMLE). That is,
\[
\theta = \arg \max_{\tilde{\theta}} p(y_1, \ldots, y_N \mid x_1, \ldots, x_N, \tilde{\theta}).
\]

We make a decision based on the posterior predictive at a fixed test point. Let \( x^* \notin \{x_1, \ldots, x_N\} \) be the test point; the posterior predictive is \( f^* \mid \mathcal{D} \) with \( f^* := f(x^*) \). Let \( F^*(k) \) be a scalar functional, such as a mean or quantile, of this distribution; we make the dependence on the kernel \( k \) of the GP explicit. Let the level \( L \in \mathbb{R} \) represent a decision threshold in \( F^*(k) \). That is, we make one decision when \( F^*(k) \geq L \) and a different one when \( F^*(k) < L \). For example, let time be a single covariate, and let outcome be the resting heart rate of a hospital patient. \( F^*(k) \) could be the 95th percentile of the GP posterior at a particular time; an alarm might trigger if \( F^*(k) \) is greater than \( L = 130 \) bpm but not otherwise \[14\].

We want to assess whether our decision would change if we used a qualitatively interchangeable kernel. Without loss of generality, we assume that \( F^*(k_0) < L \) in what follows. Then we can define non-robustness.

**Definition 1.** For original kernel \( k_0 \), we say that our decision \( F^*(k_0) < L \) is non-robust to the choice of kernel if there exists a kernel \( k_1 \) that is qualitatively interchangeable with \( k_0 \) and \( F^*(k_1) \geq L \).

**Workflow overview.** Our workflow is summarized in Algorithm [1]. We start by defining a set \( \mathcal{K}_\varepsilon \) of kernels that are “\( \varepsilon \)-near” \( k_0 \). We solve the optimization problem:
\[
k_1 = \arg \max_{k \in \mathcal{K}_\varepsilon} F^*(k).
\] (1)

Then we increase \( \varepsilon \) until the optimum of Eq. (1) changes our decision. Finally, we check whether the decision-changing kernel, \( k_1 \), is qualitatively interchangeable with \( k_0 \). It remains to precisely define a set of “\( \varepsilon \)-near” kernels (Section 2.1), to show that we can efficiently solve Eq. (1) (Section 2.1), and to provide ways to assess qualitative interchangeability (Section 2.2). We provide a number of practical choices for each step in the remainder of this section and then illustrate on simulated and real data sets in subsequent sections.

Note that although Algorithm [1] can detect non-robustness, it cannot certify robustness; it is possible, even though it may be unlikely, that there exists a qualitatively interchangeable kernel that the methodology has not detected but that still changes the decision. This point is generally true of sensitivity analyses, and the present workflow is no exception. This observation is also similar in spirit to classical hypothesis tests: a user can reject – but not accept – a null hypothesis.
Algorithm 1 Workflow for assessing robustness of GP inferences to kernel choice

1: Choose initial kernel $k_0$ using prior information.
2: Choose posterior quantity of interest $F^\star$. ▶ E.g. Posterior mean at test point $x^\star$
3: Define decision threshold $L$. ▶ E.g. 130 bpm is an alarming resting heart rate
4: Define “$\varepsilon$-near” kernels $K_\varepsilon$, for $\varepsilon > 0$. ▶ Section 2.1
5: while $F^\star(k_1) < L$ do
6: Expand $K_\varepsilon$ (increase $\varepsilon$) and re-solve Eq. (1) to get $k_1$. ▶ Section 2.1
7: end while
8: Assess whether $k_0$ and $k_1$ are qualitatively interchangeable. ▶ Section 2.2
9: if $k_0$ and $k_1$ qualitatively interchangeable then return “$F^\star$ is non-robust to the choice of kernel.”
10: else return “Failed to find that $F^\star$ is non-robust to the choice of kernel.”
11: end if

2.1 Nearby kernels and efficient optimization

We give two practical examples of how to choose $K_\varepsilon$ in the present work and detail how to solve Eq. (1) in each case. In both cases, we assume we have prior information that our kernel $k$ is smooth. First, we consider the case where we assume $k$ should be stationary. Second, we allow non-stationary kernels $k$.

Stationary kernels. By Bochner’s theorem, every stationary kernel can be represented by a positive measure [31, Thm. 4.1]. In the kernel discovery literature, it is common to make the additional assumption that this measure has a density $S(\omega) = \int e^{-2\pi i \omega k(\tau)} d\tau$, where $\omega = x - x'$. [37, 1, 38]. These authors show that the class of stationary kernels with a spectral density is a rich, flexible class of kernels. Following these works, we optimize over spectral densities $S(\omega)$ – which are just positive integrable functions on the real line – to recover stationary kernels. To make this optimization problem finite dimensional, we optimize the spectral density over a finite grid of frequencies $\omega$ and use the trapezoidal rule to recover $k$. To guarantee that our recovered kernels are not overly dissimilar from $k_0$, we constrain ourselves to an $\varepsilon$ ball in the $\ell_\infty$ norm around the spectral density of $k_0$ for some $\varepsilon > 0$. We summarize this constraint set and the resulting optimization objective in Algorithm 2; see Appendix A for more details.

Non-stationary kernels. In many modeling problems, stationarity may be a choice of convenience rather than prior conviction, or one may believe non-stationarity is probable. In either case, we wish to allow non-stationary kernels in the neighborhood $K_\varepsilon$. While there are several methods for constructing non-stationary kernels, a particularly convenient technique for our purposes relies on input warping [31 Sec 4.2.3]. Given a kernel $k_0$ and a non-linear mapping $g$, we define a perturbed kernel $k(x, x') = k_0(g(x), g(x'))$. This construction guarantees that the perturbed kernel $k$ is a kernel function as long as $k_0$ is a valid kernel. We let the function $g$ have parameters $w$ and set $g(x; w) := x + h(x; w)$, where $h : \mathbb{R}^D \to \mathbb{R}^D$ is a small neural network with weights $w$. By controlling the magnitude of $h$, we can control
Algorithm 2 Objective and $\mathcal{K}_\varepsilon$ for stationary kernels

**Objective**

1: **Input:** Frequencies $\omega_1, \ldots, \omega_G$, and density values $S(\omega_1), \ldots, S(\omega_G)$.
2: Use trapezoidal rule to approximate the integral $k(\tau) = \int e^{2\pi i \tau \omega} S(\omega)d\omega$.
3: return $F^*(k)$.

**Spectral constraint defining $\mathcal{K}_\varepsilon$**

1: **Input:** Frequencies $\omega_1, \ldots, \omega_G$, density values $S(\omega_1), \ldots, S(\omega_G)$, constraint set size $\varepsilon > 0$.
2: Compute spectral density of $k_0$, $S_0(\omega_1), \ldots, S_0(\omega_G)$ via trapezoidal rule or exact formula.
3: Constrain spectral density $S$ of $k$ as:

$$\max \left(0, (1 - \varepsilon)S_0(\omega_g)\right) \leq S(\omega_g) \leq (1 + \varepsilon)S_0(\omega_g), \quad g = 1, \ldots, G.$$ 

the deviations from $k_0$.

We could optimize over the set of non-stationary kernels with respect to the weights $w$, under the constraint $\|h(x; w)\|_2^2 \leq \varepsilon$. However, it is unclear how to enforce this constraint. Instead, we select a grid of points $\tilde{x}_1, \ldots, \tilde{x}_M \in \mathbb{R}^D$ and add a regularizer to our objective, $\frac{1}{\varepsilon M} \sum_{m=1}^M \|h(\tilde{x}_m; w)\|_2^2$, where $\varepsilon$ controls the regularization strength. While we would ideally regularize using the entire function $h$, we find using a grid of points to be a computationally cheap, mathematically simple, and empirically effective approximation. We summarize our objective as a function of the network weights $w$ in Algorithm 3. Note that we have also changed our objective to include a generic loss $\ell$; some care needs to be taken to ensure that the optimal $k_1$ is finite. See Sections 4 and 5 for specific implementations of $\ell$. Given the $\hat{w}$ minimizing the objective in Algorithm 3, we set $k_1(x, x') = k_0(g(x; \hat{w}), g(x'; \hat{w}))$.

Algorithm 3 Objective for non-stationary kernels

1: **Input:** Regularizer grid points $\tilde{x}_1, \ldots, \tilde{x}_M$, regularizer strength $\varepsilon > 0$, neural network weights $w \in \mathbb{R}^D$.
2: Define neural network $h(x; w)$ parameterized by weights $w$.
3: Define $k(x, x') := k_0(x + h(x; w), x' + h(x'; w))$.
4: return $\ell(k; F^*, L) + \frac{1}{\varepsilon M} \sum_{m=1}^M \|h(\tilde{x}_m; w)\|_2^2$

2.2 Assessing qualitative interchangeability

We introduce two visual assessments, similar to prior predictive checks [17], to assess qualitative interchangeability.

**Visual comparison of prior predictive draws.** When the covariates $x$ are one-dimensional, we can plot a small collection of functions drawn from each of the two distributions $\mathcal{GP}(0, k_0)$
and $\mathcal{GP}(0, k_1)$. To ensure that visual differences between the priors are due to actual differences in the kernels and not randomness in the draws, we use noise-matched prior draws. To define noise-matched draws, recall that one can draw from an $N$-dimensional Gaussian distribution $\mathcal{N}(0, \Sigma)$ by computing the Cholesky decomposition $L L^\top = \Sigma$; we then have $L z \sim \mathcal{N}(0, \Sigma)$, where $z \sim \mathcal{N}(0, I_N)$. We say that draws from two multivariate Gaussians are noise-matched if they use the same $z \sim \mathcal{N}(0, I_N)$.

If the user believes the two plots express the same qualitative information, the kernels are qualitatively interchangeable under this test. Two potential drawbacks to this method are as follows. (1) When covariates are high-dimensional, it may be difficult to effectively visualize prior draws. (2) The initial motivation for our paper was that some users may have difficulties expressing their prior beliefs; it might not be surprising, then, if some users may have difficulty in visually assessing prior beliefs. To address these concerns, we provide a second test next.

**Gram matrix comparison.** No matter the dimension of the covariates, we can compare the Gram matrices $k_0(X, X)$ and $k_1(X, X)$, whose $(i, j)$ entries are, respectively, $k_0(x_i, x_j)$ and $k_1(x_i, x_j)$. The two matrices $k_0(X, X)$ and $k_1(X, X)$ are the prior covariance matrices of the vector $f = (f(x_1), \ldots, f(x_N))^\top$ implied by the $\mathcal{GP}(0, k_0)$ and $\mathcal{GP}(0, k_1)$ priors. These two matrices fully characterize the prior distribution of $f$.

We cannot directly assess a matrix norm of the difference $k_1(X, X) - k_0(X, X)$ because the scale is uncalibrated. To understand whether the difference between $k_0(X, X)$ and $k_1(X, X)$ is large, we make $R$ draws, $\{\theta^{(r)}\}_{r=1}^R$, from the posterior distribution $p(\theta | D)$ (or an approximation thereof) of the original kernel hyperparameters. For each $r$, we compute the difference between $k_0(X, X)$ and $k^{(r)}(X, X)$, where $k^{(r)}$ has the same functional form as $k_0$ but with hyperparameters $\theta^{(r)}$ instead of $\theta$. If the difference between $k_1(X, X)$ and $k_0(X, X)$ is small relative to the natural posterior variability in the difference between $k_0(X, X)$ and $k^{(r)}(X, X)$ across $r$, then we conclude that $k_1$ is qualitatively interchangeable with $k_0$.

In our experiments, we make the following concrete choices (though others are possible). Unless otherwise stated, we use a Laplace approximation to the posterior around the MMLE hyperparameters. And we compare Gram matrices using a relative Frobenius norm: namely, $\|k_1(X, X) - k_0(X, X)\|_F/\|k_0(X, X)\|_F$, where $\| \cdot \|_F$ is the Frobenius norm. We construct a histogram of the relative Frobenius norms between $k^{(r)}$ and $k_0$ across $r$, with a marker indicating the position of the relative norm between $k_1$ and $k_0$. If the marker lies to the right of the mass of the histogram, we conclude that $k_1$ and $k_0$ are not qualitatively interchangeable; otherwise, we conclude $k_1$ and $k_0$ are qualitatively interchangeable.

### 2.3 Workflow illustration on synthetic data

**Data and prior.** Before turning to real data, we illustrate our workflow with a synthetic-data example. We consider $N = 35$ data points with a single covariate; see the leftmost panel of Fig. [I]. We assume we have qualitative prior beliefs that (i) $f$ is smooth and (ii) our beliefs about $f$ are invariant to translation along the single covariate (stationarity). In this
Figure 1: Synthetic data (far left). Draws from the original prior $GP(0, k_0)$ (center left) and alternative priors $GP(0, k_1)$ for extrapolation at $x^* = 5.29$ (center right) and interpolation at $x^* = 2.00$ (far right). The prior draws are noise-matched (Section 2.2). The gray-shaded area in the right three plots is the 99.7% uncertainty interval for the original prior ($GP(0, k_0)$).

case a standard kernel choice is squared exponential: $k_0(x, x') = \exp[-(1/2)(x - x')^2/\theta^2]$ [see, e.g., 31, Chap. 4]. We estimate $\theta$ via maximum marginal likelihood estimation (MMLE). Four draws from the resulting prior are shown in the second panel of Fig. 1.

Decision boundary. For the purposes of this illustration, we will look at two separate decisions. One is at $x^* = 2.00$, which is within the range of the training data (interpolation). And one is at $x^* = 5.29$, which is outside the range of the training data (extrapolation). Our functional of interest at either point will be the relative change in posterior mean

$$F^*(k) := \frac{\mu(x^*, k_0) - \mu(x^*, k)}{\sigma(x^*, k_0)},$$

where $\mu(x^*, k)$ and $\sigma(x^*, k)$ are the posterior mean and standard deviation, respectively, of $f(x^*)$ corresponding to kernel $k$. We suppose that we would make a different decision if the posterior mean were two posterior standard deviations away from its current value: $L = 2$.

Nearby kernels and efficient optimization. Since we assume stationarity, we choose Algorithm 2 at line 4 of Algorithm 1. The left panel of Fig. 2 shows what happens in the while loop as we increase $\varepsilon$. The black dots (extrapolation) quickly cross the decision threshold line, so the while loop in that case is complete. The orange triangles (interpolation) do not cross the decision threshold line, even for very large $\varepsilon$.

Qualitative interchangeability: Visual comparison of prior predictive draws. Since this example is one-dimensional, it is straightforward to visualize the prior predictive draws. For our extrapolation example ($x^* = 5.29$), let $k^{(ex)}$ be the solution to Eq. [1] with the smallest $\varepsilon$ achieving $F^*(k^{(ex)}) \geq L = 2$. The third panel of Fig. 1 shows prior predictive draws with $k^{(ex)}$; the draws are noise-matched with the second panel. Visually, the two sets of prior predictive draws (second and third panels) are qualitatively similar. Both are smooth and stationary by constraint; the length scale (cf. the number of “wiggles” in each function) and amplitudes seem unchanged. We say that $k^{(ex)}$ is qualitatively interchangeable with $k_0$. 

7
Figure 2: (Left): Maximal value of the function $F^\star$ as a function of constraint set $\varepsilon$. Comparison of the distance between $k_0(X,X)$ and $k_1(X,X)$ to the posterior variation due to hyperparameter uncertainty for extrapolation (middle) and interpolation (right). The red line corresponds to our decision-changing kernel $k_1$.

Since even the largest constraint size value considered ($\varepsilon = 10$) is not enough to induce the required change, we define $k^{(in)}$ for our interpolation example ($x^\star = 2.00$) to be the solution of Eq. (1) with the final value of $\varepsilon = 10$. The fourth panel of Fig. 1 shows prior predictive draws with $k^{(in)}$; the draws are noise-matched with the second panel. Again, by design, both sets of draws (second and fourth panels) are stationary and smooth. However, the magnitudes of peaks and troughs with $k^{(in)}$ are much larger than those with $k_0$; cf. the gray uncertainty bands, which depict the 99.7% uncertainty interval from the marginal of the $k_0$ GP. Thus, we say that $k^{(in)}$ is not qualitatively interchangeable with $k_0$ under this test. We expect the difference would be even more pronounced at higher $\varepsilon$.

**Qualitative interchangeability: Gram matrix comparison.** Our relative Frobenius norm histograms appear in Fig. 2. $k^{(ex)}$ sits within the histogram of alternative kernels generated via hyperparameter uncertainty (center), whereas $k^{(in)}$ sits far outside of this uncertainty region (right). As in the prior predictive comparison, we say that $k^{(in)}$ is not qualitatively interchangeable with $k_0$ under the Gram matrix comparison, whereas $k^{(ex)}$ is.

Finally, following our workflow, we conclude that our extrapolation example is non-robust to the choice of kernel in the sense of Definition 1. On the other hand, in our interpolation example, we fail to find non-robustness.

## 3 Stationary perturbations to a model of heart rates

We now apply our workflow (Section 2) to assess the sensitivity of GP predictions of hospital patient deterioration. [9] use a GP to model individual patients’ heart rates and predict potentially troubling behavior at a future time $x^\star$. We check whether this prediction can be sensitive to the choice of kernel.

**Data, model, and decision.** [9] observe an outcome, heart-rate data measured in beats
Figure 3: Sensitivity of heart rate analysis in Section 3. (Top row): Observed data (left). Noise-matched draws from original prior $GP(0, k_0)$ (middle) and alternative prior $GP(0, k_1)$ (right). (Bottom row): Comparison of the difference between $k_0$ and $k_1$ (red line) to posterior hyperparameter uncertainty (histogram; left). Once we expand the constraint set to $\varepsilon = 0.24$ the predicted 95% quantile of heart rate at $x^*$ exceeds 130 bpm (red line; middle). Comparison of posterior distributions of $f^*$ computed using $k_0$ (blue) and $k_1$ (red; right).

per minute (bpm), as a function of one covariate, time. The authors choose their GP model to have mean equal to zero and a kernel equal to the sum of a squared exponential and Matern-5/2 kernel; see Appendix C. We fit the overall kernel’s hyperparameters via MMLE and refer to the resulting kernel as $k_0$. Some standard hospital alarm systems activate at 130 bpm [14], a threshold describing a worryingly-high resting heart rate. So we consider the task of predicting whether the 95th percentile of the GP posterior is above $L = 130$ bpm.

Most predictions in [9] take place 1.5 hours in the future, so we set $F^*$ to be the 95th quantile at 1.5 hours hours after the last observed data point. Since the data from [9] is confidential, we use heart-rate data from the 2019 Computing in Cardiology Challenge [33, 19].

Prior beliefs. [9] note that $k_0$ encodes the belief that “longer trends (on the order of hours) are governed by the smooth RBF kernel, while minutely variations in [heart-rate] are governed by a twice-differentiable Matérn(5/2) kernel.” Although [9] are not explicit about assuming stationarity, we presume it is a reasonable prior belief here; while we expect that a patient’s heart rate may change while in the hospital, our prior beliefs about the timing of any changes may be roughly uniform. We thus choose $K_\varepsilon$ according to the stationary specification in Section 2.1.

Robustness. Fig. 3 depicts our workflow (Algorithm 1) in action. We increase $K_\varepsilon$ until the solution to Eq. (1) gives a kernel, $k_1$, with $F^* \geq L$. We then compare noise-matched samples from the priors using $k_0$ and $k_1$. The noise-matched samples do not clearly represent different pieces of prior information; both prior plots display functions that are fairly rough with similar length scales. Finally, we compare the relative Frobenius norm between $k_0(X, X)$
and $k_1(X, X)$ to that between $k_0(X, X)$ and kernels from the natural variability around the MMLE hyperparameters. The difference between $k_0$ and $k_1$ is small compared to typical variation around $k_0$ due to hyperparameter uncertainty. Our tests suggest $k_0$ and $k_1$ are qualitatively interchangeable; we conclude that the prediction that $F^*$ will fall below the alarm threshold is non-robust in the sense of Definition 1.

While this outcome may be surprising, it is not entirely unintuitive. The patient’s heart rate is trending up toward the end of the observed data. In Appendix C, we show an example where the observed data is trending downward at the end of the observed data. In the latter case, when we expand $K_\varepsilon$ to obtain $F^* \geq 130$, the resulting kernel $k_1$ fails both of our tests of qualitative interchangeability.

4 Non-stationary perturbations to a model of carbon dioxide emissions

In a now-classic analysis of carbon dioxide (CO_2) levels at Mauna Loa, [31] predicted future CO_2 levels based on data up to 2003. With data up to 2021, we can now see that the [31] analysis substantially underestimates present-day CO_2 levels; compare the gray region (99.7% quantile of the original predictions) to the the green (true levels) in Fig. 4. In this section, we show that this model’s predictions of modern CO_2 levels are non-robust to kernel choice.

**Data, model, and decision.** We start by recreating the analysis of [31]. At the present day, monthly data for CO_2 emissions is available from the year 1958 through 2021. But [31] ran their original analysis in 2006 and used training data up to 2003. [31] use a kernel that is a sum of four basic kernels, where each term plays a specific role; e.g. a periodic term models the periodic seasonal trend in CO_2 levels, and a rational quadratic term models small “sub-seasonal” trends. See Appendix D for a full description of the kernel. We take this kernel with hyperparameters fit via MMLE as our $k_0$. We will consider prediction at the point $x^*$ corresponding to June 2020 and let $F^*$ be the posterior mean at $x^*$. We will say the posterior has substantively changed if $F^* = L$, where $L$ is the actual observed CO_2 level in June of 2020.

**Prior beliefs.** While $k_0$ is a stationary kernel we might also have non-stationary prior information here such as known historical (or expected future) developments in climate policy or technology. We therefore choose $K_\varepsilon$ according to the non-stationary specification in Section 2.1. In particular, we generate $K_\varepsilon$ using our input-warping methodology. For our regularizer grid, we use 600 evenly spaced points $\tilde{x}_1, \ldots, \tilde{x}_{600}$ between 1958 and January of 2021 to control the behavior of $h$ throughout our time period of interest. Input warping the entire kernel as $k = k_0(g(x), g(x'))$ would violate an important piece of prior information that we have about CO_2 levels: we know CO_2 has a regular seasonality, where minimal levels occur in the winter and maximal levels occur in the summer. The original $k_0$ accounts for this feature of the data with a periodic term, with period estimated via MMLE; as shown...
in Fig. 4 (top), this periodicity lines up very well with the training data. To produce an alternative kernel that accounts for this piece of prior knowledge, we leave the periodic portion of the kernel unwarped. To parameterize $g$, we use a fully connected network with two hidden layers, 50 units, and ReLU nonlinearities. To ensure the optimal $k_1$ is finite, we take the loss in Algorithm 3 to be $\ell(k; F^*, L) = (F^*(k) - L)^2$, which guarantees that our objective is bounded below.

**Robustness.** We now use our workflow to ask whether qualitatively interchangeable kernels might have better predicted the observed data from the year 2020; see Fig. 4 for our results. We lower the regularization strength (i.e. increase $\varepsilon$ in Algorithm 3) until $F^* \geq L$. In the bottom of Fig. 4, we plot noise-matched prior draws from $k_0$ alongside draws from the first $k_1$ to meet $F^* \geq L$. Differences between draws from $k_0$ and $k_1$ are almost visually indistinguishable on the scale of all prior draws. A closer inspection in Appendix D confirms that the two priors capture the same yearly periodic trend. These same zoomed-in plots show that the priors are not completely indistinguishable; however, in our opinion, the priors are qualitatively interchangeable. The Gram matrix comparison in Fig. 8 of Appendix D further confirms that the perturbed kernel sits well within the histogram of alternate kernels stemming from hyperparameter uncertainty. We conclude that future predictions of CO$_2$ levels using the original $k_0$ are non-robust to the choice of kernel in the sense of Definition 1.
Figure 5: Sensitivity of MNIST analysis in Section 5. (Left): $F^*$ as a function of regularizer strength. (Center): Histogram of the relative Frobenius norms of the 1000 (one for each test image) input-warped kernel Gram matrices (in red) plotted with those arising from kernel hyperparameter uncertainty (in black). (Right): Example test images $x^*$ (upper) and their warps $g(x^*)$ (lower) and predicted class labels (above and below).

5 Non-stationary perturbations in classifying MNIST digits

So far we have restricted our attention to low-dimensional covariates. To evaluate our approach in a high-dimensional setting, we reproduce the MNIST image classification experiment of [27]. It is rare to have well-specified prior beliefs about high-dimensional functions, so in this case we do not consider different kernels as arising from prior beliefs. Rather we imagine $k_0$ is used purely for convenience and predictive quality – but that a malicious actor is interested in changing the kernel to achieve different test predictions without detection.

Data, model, and decision. [27] use 1000 randomly sampled MNIST images for a training set, and a separate 1000 images for a test set. Given a test image $x^*$, [27] predict the class label $c \in \{1, \ldots, C\}$ by using a $C$-output GP with compositional structure, considered as the infinite-width limit of a sequence of Bayesian neural networks [27, 10]. The authors classify any image $x^*$ by picking the class $c$ that has the posterior mean of $f_c(x^*)$, i.e. $\mu_c(x^*)$, closest to 0.9; see Appendix E for details. We use both the same kernel and hyperparameters for $k_0$ as in [27]. We imagine that the malicious actor wants to change the label of a single test image $x^*$ from its current label $c_0$ to a different label $c_1$. For concreteness, we set $c_1 := |c_0 - 1|$. We consider 1000 separate iterations of this exercise, once for each of the 1000 test images. For a particular $x^*$, we set our posterior quantity of interest to be $F^* = \mu_{c_1}(x^*) - \mu_{c_0}(x^*)$. Since $F^* = 0$ implies that we have changed the prediction for $x^*$, we set our decision threshold $L = 0$.

Malicious actor. Instead of considering a range of priors that match prior beliefs, we here consider a range of priors that will allow the malicious actor to avoid detection. Since we have no prior belief of stationarity, we use the non-stationary construction from Section 2.1. We find that optimizing $F^*$ directly leads to kernels where for $c \neq c_1$, $\mu_c(x^*)$ takes on values
at least an order of magnitude higher than for the original kernel. This change could be easily detected by a user with an automated system. For the purposes of the malicious actor, we therefore consider these kernels to not be qualitatively interchangeable. We instead optimize a surrogate loss that maximizes the log probability of $c_1$ being correct and all other classes being incorrect; see Appendix E for details. We find that optimizing this surrogate loss leads to more benign-looking $\mu_c(x^*)$ and allows us to achieve $F^* \geq L$.

**Robustness.** Fig. 5 shows the results of our workflow applied to this problem. We find a sufficiently large setting of $\varepsilon$ that allows us, across all 1000 test-image problems, to change every decision. In particular, for $\varepsilon = 10^{-4}$, we are able to find perturbed kernels that change the predicted class label in every case. It is not clear how to visualize our priors in this application. So, of the approaches in Section 2.2, we use only the hyperparameter uncertainty visualization to assess qualitative interchangeability. [27] optimize the hyperparameters of $k_0$ over a grid, so it is not clear that the hyperparameters are sufficiently close to a global minimum (or that the global minimum is sufficiently representative) to make the Laplace approximation reasonable. Instead, we note that, given that the authors optimized the hyperparameters with grid search, the size of the grid defines a natural variability in the hyperparameters $\theta$. To be conservative, we sample $\theta^{(r)}$ from an area around the MMLE that is over 10 times smaller than the full grid. (If we used the full original grid, we would find only more non-robustness.) The Gram matrices corresponding to our perturbed kernels are much closer to $k_0(X, X)$ than are the Gram matrices corresponding to each $\theta^{(r)}$. We conclude that classification of handwritten digits with the compositional kernel proposed by [27] is non-robust to the choice of kernel in the sense of Definition 1.

6 Discussion

In this paper, we proposed and implemented a workflow for measuring the sensitivity of GP inferences to the choice of the kernel function. We used our workflow to discover substantial non-robustness in a variety of practical examples, but also showed that many analyses are not flagged as non-robust by our method. We feel that robustness quantification, as in the present paper, will primarily have a positive impact: we hope that robustness quantification can help overcome brittle or cherry-picked machine learning analyses that will fail to yield stable conclusions in real practice. However, a potential negative outcome of our robustness quantification machinery is that it may be repurposed for adversarially changing the predictions of a GP model.

There are many exciting directions for expanding on the present work – both within our existing workflow and beyond. Many of our choices in the present paper were made for mathematical convenience. For instance, the constraint in our stationary objective in Algorithm 2 and the regularizer in our non-stationary objective in Algorithm 3 might be replaced by other notions of “nearby” spectral densities or “small” input warpings, respectively.

Additionally, our framework flags robustness but does not show how to make an analysis
more robust. The instances of non-robustness we have found suggest it might be worthwhile to develop methods to robustify GP inferences to the choice of kernel. One challenge would be understanding how to best balance sensitivity, which can be desirable in that a method should adapt to the data at hand, and robustness, to encourage stability and generalization of results.

Acknowledgments

This work was supported by the MIT-IBM Watson AI Lab, an NSF CAREER Award, an ARPA-E project with program director David Tew, and an ONR MURI.

References

[1] Gregory W Benton, Wesley J Maddox, Jyson P Salkey, Julio Albinati, and Andrew Gordon Wilson. Function-space distribution over kernels. In Advances in Neural Information Processing Systems (NeurIPS), 2019.

[2] James O. Berger. Bayesian Robustness, pages 1–32. Springer New York, New York, NY, 2000.

[3] James O. Berger and L. Mark Berliner. Robust Bayes and empirical Bayes analysis with $\epsilon$-contaminated priors. Annals of Statistics, 14(2):461–486, 1986.

[4] James O. Berger, Elías Moreno, Luis Raul Pericchi, M. Jesús Bayarri, José M. Bernardo, Juan A. Cano, Julián De la Horra, Jacinto Martín, David Ríos-Insúa, Bruno Betró, A. Dasgupta, Paul Gustafson, Larry Wasserman, Joseph B. Kadane, Cid Srinivasan, Michael Lavine, Anthony O’Hagan, Wolfgang Polasek, Christian P. Robert, Constantinos Goutis, Fabrizio Ruggeri, Gabriella Salinetti, and Siva Sivaganesan. An overview of robust Bayesian analysis. Test, 3(1):5 – 124, 1994.

[5] Ilija Bogunovic, Jonathan Scarlett, Stefanie Jegelka, and Volkan Cevher. Adversarially robust optimization with Gaussian processes. In Advances in Neural Information Processing Systems (NeurIPS), pages 5760 – 5770, 2018.

[6] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018.

[7] Luca Cardelli, Marta Kwiatkowska, Luca Laurenti, and Andrea Patane. Robustness guarantees for Bayesian inference with Gaussian processes. In AAAI Conference on Artificial Intelligence, pages 7759 – 7768, 2019.

[8] Li-Fang Cheng, Bianca Dumitrascu, Gregory Darnell, Corey Chivers, Michael Druagelis, Kai Li, and Barbara E. Engelhardt. Sparse multi-output Gaussian processes for online medical time series prediction. BMC Medical Informatics and Decision Making, 20(1):152 – 174, 2020.

[9] Glen Wright Colopy, Marco A F Pimentel, Stephen J Roberts, and David A Clifton. Bayesian Gaussian processes for identifying the deteriorating patient. In 2016 38th Annual International
[10] Alexander G. de G. Matthews, Jiri Hron, Mark Rowland, Richard E. Turner, and Zoubin Ghahramani. Gaussian process behaviour in wide deep neural networks. In *International Conference on Learning Representations (ICLR)*, 2018.

[11] David Duvenaud. *Automatic model construction with Gaussian processes*. PhD thesis, University of Cambridge, 2014.

[12] David Duvenaud, James Robert Lloyd, Roger Grosse, Joshua B. Tenenbaum, and Zoubin Ghahramani. Structure discovery in nonparametric regression with compositional kernel search. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, 2013.

[13] Federico Ferrari and David B Dunson. Identifying main effects and interactions among exposures using Gaussian processes. *Annals of Applied Statistics*, 14(4):1743 – 1758, 2020.

[14] Richard L. Fidler, Michele M. Pelter, Barbara J. Drew, Jorge Arroyo Palacios, Yong Bai, Daphne Stannard, J. Matt Aldrich, and Xiao Hu. Understanding heart rate alarm adjustment in the intensive care units through an analytical approach. *PLoS One*, 12(11):1 – 10, 2017.

[15] Joseph Futoma, Sanjay Harirahan, and Katherine Heller. Learning to detect sepsis with a multitask Gaussian process RNN classifier. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70, pages 1174 – 1182, 2017.

[16] Joseph Futoma, Sanjay Harirahan, Katherine Heller, Mark Sendak, Nathan Brajer, Meredith Clement, Armando Bedoya, and Cara O’Brien. An improved multi-output Gaussian process RNN with real-time validation for early sepsis detection. In *Proceedings of the 2nd Machine Learning for Healthcare Conference (MLHC)*, pages 243 – 254, 2017.

[17] Jonah Gabry, Daniel Simpson, Aki Vehtari, Michael Betancourt, and Andrew Gelman. Visualization in Bayesian workflow. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 182(2):389 – 402, 2019.

[18] Ryan Giordano, Tamara Broderick, and Michael I. Jordan. Covariances, robustness, and variational Bayes. *Journal of Machine Learning Research*, 19(51):1 – 49, 2018.

[19] Ary L Goldberger, Luis A. N. Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch. Ivanov, Roger G. Mark, Joseph E. Mietus, George B. Moody, Chung-Kang Peng, and H. Eugene Stanley. PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. *Circulation*, 101(23):e215 – e220, 2000.

[20] Paul Gustafson. Local sensitivity of posterior expectations. *Annals of Statistics*, 24(1):174 – 195, 1996.

[21] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant,
Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357 – 362, 2020.

[22] Daniel Hernández-Lobato, José Miguel Hernández-Lobato, and Pierre Dupont. Robust multi-class Gaussian process classification. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2011.

[23] Pasi Jylänki, Jarno Vanhatalo, and Aki Vehtari. Robust Gaussian process regression with a Student-t likelihood. *Journal of Machine Learning Research*, 12(99):3227 – 3257, 2011.

[24] C. D. Keeling and T. P. Whorf. Atmospheric CO2 records from sites in the Scripps Institution of Oceanography (SIO) air sampling network (1985 - 2007). 10 2004.

[25] Hyun-Chul Kim and Zoubin Ghahramani. Outlier robust Gaussian process classification. In *Structural, Syntactic, and Statistical Pattern Recognition*, pages 896 – 905, 2008.

[26] Duncan Lee, Sabyasachi Mukhopadhyay, Alastair Rushwork, and Suji K Sahu. A rigorous statistical framework for spatio-temporal pollution predition and estimation of its long-term impact on health. *Biostatistics*, 2:370 – 385, 2017.

[27] Jaehoon Lee, Jascha Sohl-Dickstein, Jeffrey Pennington, Roman Novak, Sam Schoenholz, and Yasaman Bahri. Deep neural networks as Gaussian processes. In *International Conference on Learning Representations (ICLR)*, 2018.

[28] Runjing Liu, Ryan Giordano, Michael I. Jordan, and Tamara Broderick. Evaluating sensitivity to the stick breaking prior in Bayesian nonparametrics. *arXiv preprint arXiv:1810.06587v1*.

[29] Roman Novak, Lechao Xiao, Jiri Hron, Jaehoon Lee, Alexander A. Alemi, Jascha Sohl-Dickstein, and Samuel S. Schoenholz. Neural tangents: Fast and easy infinite neural networks in python. In *International Conference on Learning Representations (ICLR)*, 2020.

[30] Rishik Ranjan, Biao Huang, and Alireza Fatehi. Robust Gaussian process modeling using EM algorithm. *Journal of Process Control*, 42:125 – 136, 2016.

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

[32] Boyu Ren, Xiao Wu, Danielle Braun, Natesh Pillai, and Francesca Dominici. Bayesian modeling for exposure response curve via Gaussian processes: causal effects of exposure to air pollution on health outcomes. *arXiv preprint arXiv:2105.0354v1*, 2021.

[33] Matthew Reyna, Chris Josef, Russell Jeter, Supreeth Shashikumar, Benjamin Moody, M. Brandon Westover, Ashish Sharm, Shamim Nemati, and Gari Clifford. Early Prediction of Sepsis from Clinical Data – the PhysioNet Computing in Cardiology Challenge 2019 (version 1.0.0), 2019.

[34] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen,
E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[35] Wenjia Wang and Bing-Yi Jing. Convergence of Gaussian process regression: Optimality, robustness, and relationship with kernel ridge regression. *arXiv preprint arXiv:2104.09778v1*, 2021.

[36] Larry Wasserman, Michael Lavine, and Robert L. Wolpert. Linearization of Bayesian robustness problems. *Journal of Statistical Planning and Inference*, 37:307 – 316, 1993.

[37] Andrew Gordon Wilson and Ryan Prescott Adams. Gaussian process kernels for pattern discovery and extrapolation. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, 2013.

[38] Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P. Xing. Deep kernel learning. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2016.

[39] George Wynne, François-Xavier Briol, and Mark Girolami. Convergence guarantees for Gaussian process means with misspecified likelihoods and smoothness. *Journal of Machine Learning Research*, 22(123):1 – 40, 2021.
A Details of spectral density constraints

Here, we give the details of how we optimize over spectral densities to produce a stationary kernel as summarized in Algorithm 2. Our goal is to optimize over the set of stationary kernels. It is not immediately clear how to enforce this constraint; however, Bochner’s theorem [31, Thm. 4.1] tells us that every stationary kernel $k(\tau)$, where $\tau \in \mathbb{R}^D$ has a positive finite spectral measure $\mu$ such that:

$$k(\tau) = \int_{\mathbb{R}^D} e^{2\pi i \tau^T \omega} d\mu(\omega).$$  \hspace{1cm} (2)

A common assumption in the literature on kernel discovery [37, 1, 38] is to assume that $\mu$ has a density $S$ with respect to the Lebesgue measure; that is, we can write:

$$k(\tau) = \int_{\mathbb{R}^D} e^{2\pi i \tau^T \omega} S(\omega) d\omega.$$ \hspace{1cm} (3)

These works have shown that the class of stationary kernel with spectral densities is a rich, flexible class of kernels. In all of our examples optimizing over spectral densities, we have $D = 1$. We thus assume $D = 1$ in the rest of our development here. In this case, it must be that $S$ is symmetric around the origin to obtain a real-valued $k$. So, we can simply Eq. (3) further as:

$$k(\tau) = \int_0^\infty \cos(2\pi \tau \omega) S(\omega) d\omega.$$ \hspace{1cm} (4)

Optimizing over positive functions $S$ on the positive real line seems at least somewhat more tractable than optimizing over stationary positive-definite functions $k(\tau)$. However, this is still an infinite dimensional optimization problem. To recover a finite dimensional optimization problem, we follow [1] and choose a grid $\omega_1, \ldots, \omega_G$. We can then optimize over the finite values $S(\omega_1), \ldots, S(\omega_G)$ and use the trapezoidal rule to approximate the integral in Eq. (4). [1] find that $G = 100$ gives reasonable performance in their experiments; we find the same in ours, and fix $G = 100$ throughout. [1] recommend setting $\omega_g = 2\pi g/(8\tau_{max})$, where $\tau_{max}$ is the maximum spacing between datapoints. We find this to sometimes give inaccurate results in the sense that using the trapezoidal rule / an exact formula to compute the density of $k_0$, $S(\omega_1), \ldots, S(\omega_G)$ and then using the trapezoidal rule to recover the gram matrix $k_0(X, X)$ gives an inaccurate approximation to $k_0(X, X)$. This is problematic in our case, as it would imply $k_0(X, X)$ is not in the constraint set for small $\varepsilon$. Instead, we recommend setting our $\omega_g$’s as a uniform grid from $\omega_1 = 0$ up to an $\omega_G$ such that $S_0(\omega_G)$ is equal to the floating point epsilon ($10^{-15}$ in our experiments); some manual experimentation will be required to implement this rule.

As we are only interested in kernels nearby $k_0$, we will have to put some kind of constraint on $k_1$’s spectral density, $S_1(\omega_1), \ldots, S_1(\omega_G)$. We use a simple $\varepsilon$-ball given by:

$$\max \left( 0, (1 - \varepsilon)S_0(\omega_g) \right) \leq S(\omega_g) \leq (1 + \varepsilon)S_0(\omega_g), \quad g = 1, \ldots, G.$$ \hspace{1cm} (5)

As long as our posterior functional of interest $F^*$ is a differentiable function of the kernel matrix, we can compute gradients of $F^*$ with respect to our discretized spectral density. Rather than manually work out the derivatives of the trapezoidal rule combined with $F^*$, we use the automatic differentiation package jax[6]. Given a gradient of $F^*$, we take a step in the direction of the

\[1\] Note that jax does not use 64 bit floating point numbers by default. We found that the increased precision given by 64 bit floating point arithmetic to be important in our experiments.
gradient and then project the current iterate onto our constraint set in Eq. (5) by clipping the resulting spectral density.

B Additional details of synthetic-data experiment

We generated the $x$-component of the synthetic data by first drawing 25 uniformly random numbers in $[0, 5]$. To increase the density around the interpolation point $x^* = 2.00$, we then draw 10 uniformly random numbers in $[1.9, 2.1]$. The extrapolation point $x^* = 5.29$ lies 0.5 to the right of the largest $x$ value drawn. The $y$-component is defined to be

$$y_i = \frac{x_i^2}{2} + \cos(\pi x_i) + \epsilon_i,$$

where $\epsilon_i \sim \mathcal{N}(0, 0.01)$.

We chose the $\varepsilon$ grid for extrapolation ($x^* = 5.29$) to be 15 evenly-spaced values between 0.2 and 0.8. The grid for interpolation ($x^* = 2.00$) is 15 evenly-spaced on the log grid between $10^{0.1}$ and $10^1$.

To discretize the spectral density, we follow Appendix A in using 100 frequencies evenly-spaced from 0 to 2. To optimize over nearby spectral densities, we also perform constrained gradient descent with randomized initializations in the sense of Appendix A. For extrapolation ($x^* = 5.29$), using 25 random seeds, we find sensitivity. For interpolation ($x^* = 2.00$), even with 40 random seeds, we fail to detect non-robustness.

Computing is done using a computing cluster, which has xeon-p8 computing cores. We request 7 nodes, each using 15 cores to run parallel experiments across both $\epsilon$ and the random seed for initialization. Total wall-clock time comes to roughly 5 minutes.

C Additional details for the heart rate example

Here, we give additional details for our heart rate modeling example from Section 3. The assets from the Computing in Cardiology challenge [33, 19] are available under Creative Commons Attribution 4.0 International Public License. According to [33], the data was collected under the approval of appropriate institutional review boards, and personal identifiers were removed. Following [9], we first take the log transform of our heart rate observations $y_n$. We then zero-mean the observations ($\sum_{n=1}^N y_n = 0$) and set them to have unit variance ($\sum_{n=1}^N y_n^2 = 1$). The kernel used by [9] to model the resulting data log-scaled standardized data is a Matérn 5/2 kernel plus a squared exponential kernel:

$$k_0(x, x') = h_1^2 \left(1 + \frac{\sqrt{5}|x - x'|}{\lambda_1} + \frac{5|x - x'|^2}{3\lambda_1^2}\right) \exp \left[-\frac{\sqrt{5}|x - x'|}{\lambda_1}\right] + h_2^2 \exp \left[-\frac{|x - x'|^2}{2\lambda_2^2}\right],$$

where $h_1, h_2, \lambda_1, \lambda_2 > 0$ are kernel hyperparameters, which we set via MMLE. While all inferences are done on the zero-mean, unit-variance log-scaled data, all of our plots and discussion are given in the untransformed (i.e. raw bpm) scale for ease of interpretability.
Figure 6: Sensitivity of heart rate analysis in Appendix C for an example where we fail to find non-robustness. (Top-left): Heart rate data; notice the data is trending downwards at the end of the time series. (Top-right): Prior draws from our original kernel $k_0$ from Eq. (6). (Bottom-left): Prior draws from our decision-changing kernel $k_1$ that achieves $F^* = L$, noise matched by color to the draws from $k_0$. (Bottom-right): Comparison of the difference between $k_0$ and $k_1$ (red line) to posterior hyperparameter uncertainty (histogram).
In the main text, we showed an example where our workflow in Algorithm 1 discovered non-
robustness in predicting whether a patient’s heart rate would be likely to be above 130 BPM or
not 1.5 hours in the future. We noted that there was some evidence in the data supporting this
finding: the patient’s heart rate was trending upward towards the end of the observed data, so we
might expect that small changes to the prior could result in significant posterior mass being placed
on high heart rates. To demonstrate that we do not always find GP analyses non-robust to the
choice of the prior, we give an example here where we fail to find non-robustness. For our example,
we use a different patient from the Computing in Cardiology challenge [33, 19]. The heart rate for
this patient is plotted in Fig. 6; notice that their heart rate is trending down at the end of the
observed data.

As in Section 3, we use the constraint set and objective specified by Algorithm 2 (i.e. we constrain
ourselves to stationary kernels with spectral densities close to the density of \( k_0 \)). Following Algo-
rithm 1, we expand the constraint set size until we achieve \( F^* = \Delta \). We then assess whether the
recovered \( k_1 \) is qualitatively interchangeable with \( k_0 \). We plot noise-matched prior
draws from \( k_0 \) and \( k_1 \) in Fig. 6. We see that \( k_1 \) has obvious qualitative deviations from \( k_0 \); the
functions drawn from \( k_1 \) have noticeably larger variance (count the number of times the functions
from \( k_1 \) pass 130 bpm). Additionally, we see in Fig. 6 that the relative Frobenius norm between
\( k_0(X,X) \) and \( k_1(X,X) \) is much larger than the typical deviations around \( k_0 \) due to hyperparameter
uncertainty. We conclude that \( k_0 \) and \( k_1 \) are not qualitatively interchangeable. Thus, we say that
we fail to find non-robustness in the sense of Definition 1. Again, this conclusion is fairly sensible:
at the final observation, the patient’s heart rate is below 80 BPM and is trending downwards. It
thus seems reasonable that it would take a somewhat unusual prior to predict that the patient’s
heart rate would suddenly spike to 130.

The heart rate data we use necessarily contains information about individuals – each time series of a
heart rate comes from an individual patient. The creators of this dataset had their study approved
by an institutional review board, so we assume they obtained consent from all people in the dataset.
The dataset contains no immediately personally identifying information; all patients are referred to
only by a number. However, we are not aware of a proof that no personally identifying information
could be extracted from this dataset. We are confident that this dataset contains no offensive
information.

The heart rate experiments in were run on a laptop with a six-core i7-9750H processor. Each of
the two experiments took roughly five minutes to complete.

D Additional details for CO2 experiment

Here, we give additional details on the CO2 experiment from Section 4. Our dataset is a series of
monthly CO2 levels taken from Mauna Loa in Hawaii between 1958 and January of 2021 [24]; we
download our data from https://scrippsco2.ucsd.edu/assets/data/atmospheric/stations/in situ
co2/monthly/monthly_in situ_co2.csv. The data is freely available (no attached license). [31, Section 5.4.3] predict future CO2 levels using
Figure 7: Sensitivity analysis of Mauna Loa. Each row plots noise matched samples from a zero mean Gaussian process with original and perturbed kernel functions. These plots provide a zoomed in view of the prior samples shown in Fig. 4. We note that draws from $k_1$ are in-phase with those of $k_0$ (i.e. $k_1$ captures the seasonal maxima and minima of CO$_2$ just as well as $k_0$ does). Overall, there is high agreement between functions sampled from the two GPs.
a GP. Their kernel is the sum of four terms:

\[
k_0(x_1, x_2) = \theta_1^2 \exp\left(-\frac{(x_1 - x_2)^2}{2\theta_2^2}\right) + \theta_3^2 \exp\left(-\frac{(x_1 - x_2)^2}{2\theta_4^2} - \frac{2 \sin^2(\pi (x_1 - x_2))}{\theta_5^2}\right) + \theta_6^2 \left(1 + \frac{(x_1 - x_2)^2}{2\theta_7^2\theta_8}\right) + \theta_9^2 \exp\left(-\frac{(x_1 - x_2)^2}{2\theta_{10}^2}\right),
\]

where the $\theta_i$ comprise the kernel hyperparameters (in addition to the noise variance $\sigma^2$). The different components of this kernel encode different pieces of prior knowledge. The two squared exponentials encode long-term trends and small-scale noise, respectively. The rational quadratic kernel (Eq. (9)) encodes small seasonal variability in CO$_2$ levels between different years. The periodic kernel captures the periodic trend in CO$_2$ levels, which peak in the summer and reach their minimum in the winter. This periodic is multiplied by a squared exponential to allow deviations away from exact periodicity.

Similar to [31, Section 5.4.3], we first compute the empirical mean of the training data’s CO$_2$ levels. We use this as the mean function for the GP. To set the GP hyperparameters, we find that the hyperparameters values reported in [31, Section 5.4.3] are close, but not exactly, the MMLE solution on our data set, because the Jacobian of the marginal log-likelihood has an entry substantively different from zero.

We set hyperparameters by 10 random restarts of MMLE, where the solution iterates are initialized at the values reported in [31, Section 5.4.3]. The fitted values are $\theta_1 = 68.58$, $\theta_2 = 69.09$, $\theta_3 = 2.55$, $\theta_4 = 87.60$, $\theta_5 = 1.44$, $\theta_6 = 0.66$, $\theta_7 = 1.18$, $\theta_8 = 0.74$, $\theta_9 = 0.18$, $\theta_{10} = 0.13$, $\theta_{11} = 0.19$. They are, for the most part, within 5% of the values reported in [31, Section 5.4.3].

When [31] ran their analysis, only data up to 2003 were available. As it turns out, their analysis significantly underestimates current CO$_2$ levels. We ask if a qualitatively interchangeable kernel could have changed this result. Thus, we let $F^*$ be the mean of the posterior in June 2020 and set our substantive change level $L$ to be the observed CO$_2$ level in June 2020. $k_0$ is stationary, so we could search for alternative stationary kernels using our spectral density framework from Section 2.1 (Algorithm 2). However, there is good reason to think we might want to consider non-stationary prior beliefs. Developments in technology and/or global policy could have a large impact on CO$_2$ levels. Thus, we might encode past / expected future changes in technology and policy into our prior beliefs, making our prior beliefs non-stationary.

Thus, we use the input warping approach from Section 2.1 (Algorithm 3). However, we do not input warp the entirety of $k_0$. As we know CO$_2$ data has a regular periodicity, we leave the periodic component of the kernel, $\exp[-\sin^2(\pi (x_1 - x_2))/(2\theta_5^2)]$ unwarped. In preliminary experiments, we input warped the entirety of $k_0$; the resulting prior draws sometimes had minima in the summer.

---

2This problem might be due to the existence of slightly different versions of the Mauna Loa data set. The originally link for the data is [http://cdiac.esd.ornl.gov/ftp/trends/co2/maunaloa.co2](http://cdiac.esd.ornl.gov/ftp/trends/co2/maunaloa.co2) is no longer responsive, for instance.
Figure 8: Relative Frobenius norms of Gram matrices constructed with samples drawn from a Laplace approximation (in blue) and our perturbed kernel (in red).

and maxima in the winter, a clear violation of our prior knowledge about CO$_2$ levels. We input warp all other parts of $k_0$ using a use a two hidden layer fully connected network, with 50 units and ReLU nonlinearities to parameterize $h$. Finally, to ensure the optimal $k_1$ is finite, we use $\ell(k_1; F^*, L) = (F^*(k_1) - L)^2$ in Algorithm 3, which guarantees that our objective is bounded below.

We plot noise matched prior draws for $k_0$ and the $k_1$ that achieves $F^*(k_1) = L$ in Fig. 7. The samples from $k_1$ appropriately line up with the expected maxima and minima of CO$_2$ levels (to see this, note that the draws from $k_1$ are in-phase with those from $k_0$, which correctly captures the seasonal maxima and minima). The deviations between the noise-matched samples do not seem significant, so we say that $k_1$ and $k_0$ are qualitatively interchangeable. We therefore conclude that the prediction of CO$_2$ levels under $k_0$ is non-robust to the choice of the kernel in the sense of Definition 1. We also show the relative Frobenius norms of samples drawn from a Laplace approximation about the MMLE estimate and our perturbed kernel’s Gram matrix in Fig. 8.

This dataset is not about people, and so issues of consent and personally identifying information are not relevant.

The Mauna Loa experiments were run on a laptop with a 2.3 GHz 8-Core Intel Core i9, with 64 GB of RAM. The experiment (which optimizes from five random initialization) took about 15 minutes to run, with each seed taking about 3 mins.

E More details on MNIST experiments

MNIST data is available under Creative Commons Attribution-Share Alike 3.0.

We use the publicly available neural-tangents package for constructing the kernels in our MNIST experiments. The package is available under Apache License Version 2.0. We follow the experimental setup of [27] where the authors use a Gaussian process with a kernel corresponding to a 20 layer, infinitely wide, fully connected, deep neural network with ReLU non-linearities. They place zero mean Gaussian priors over the weights, $\mathcal{N}(0, \sigma_w^2)$, and biases, $\mathcal{N}(0, \sigma_b^2)$, and set the hyper-parameters $\sigma_w^2 = 1.45$ and $\sigma_b^2 = 0.28$ via a grid search over parameters to maximize held-out predictive performance. [27] use a GP with $C = 10$ outputs (classes). They pre-process one-hot
Figure 9: Additional MNIST experiments. Here we visualize the training and test set performances along the hyperparameter grid used for assessing qualitative interchangeability. The train and test accuracies exhibit high performance and low variability across the grid.

encoded output vectors to have zero mean, i.e. \( y_{ic} = 0.9 \) if \( c \) is the correct class for the \( i \)th training point, and \( y_{ic} = -0.1 \) for all incorrect classes; input images are flattened and an overall mean is subtracted from every image. Test prediction is made by selecting a class corresponding to the GP output with mean closest to 0.9. The resulting GP trained on one thousand images from the MNIST training set and evaluated on the MNIST test set achieves an accuracy of 92.79%.

In our experiments we assess the robustness of their kernel. The 28 \( \times \) 28 MNIST images require a warping function \( g : \mathbb{R}^{784} \rightarrow \mathbb{R}^{784} \). We use a fully connected multi-layer perceptron with one 784 unit hidden layer, 784 input, and 784 output units with ReLU non-linearities to parametrize \( g \). Let \( c_0 \) be the prediction under the original kernel at a target test image \( x^\star \). We define \( c_1 := |c_0 - 1| \) and create a “fake” output \( y^\star \) with \( y^\star_{c_1} = 0.9 \) and \( y^\star_c = -0.1 \) for \( c \neq c_1 \). We find parameters of \( g \) by minimizing the objective in Algorithm 3 plugging in

\[
\ell(k; F^\star, L) = -\frac{1}{C} \sum_{c=1}^{10} \log p(y^\star_{c}|X, x^\star, Y),
\]

i.e. the negative log-likelihood of the “fake” output at a particular test image \( x^\star \) under the perturbed kernel; \( X \) and \( Y \) are the train inputs and outputs. As we discussed in the main text, directly optimizing the posterior quantity of interest \( F^\star = \mu_{c_1}(x^\star) - \mu_{c_0}(x^\star) \) produces unrealistic outputs, e.g. \( \mu_{c_0}(x^\star) \ll -0.1 \). Such predictions would look obviously suspicious to a user, so we would say that our supposed malicious actor has not achieved their goal in this case. Instead, we optimize the surrogate loss in Eq. (11). With this surrogate loss we are able to find kernel perturbations yielding benign-looking outputs and achieving the goal of the malicious actor to change the prediction at \( x^\star \) to \( c_1 \), i.e. \( \mu_{c_1}(x^\star) \approx 0.9 \) and \( \mu_{c}(x^\star) \approx -0.1 \) for all \( c \neq c_1 \). In this case, we feel that a user would not be able to identify these predictions as obviously wrong, and so we say that the malicious actor has achieved their goal of changing the predictions of \( k_0 \) without detection.

**Hyperparameter sensitivity grid** To quantify variability in the Gram matrices arising from hyperparameter uncertainty, we vary \( \sigma_w^2 \) over 30 uniformly spaced points between 1.4 and 1.5, and \( \sigma_b^2 \) over 30 uniformly spaced points between 0.23 and 0.33. Fig. 9 shows that over the 900 possible hyperparameter combinations the train and test accuracies remain high and exhibit low variability.
We do not know if the creators of the MNIST dataset obtained consent from each person who wrote the digits; however, we are not aware of any issues in this regard. The dataset contains no immediately personally identifying information, and we strongly suspect that no personally identifying information could be recovered given only pictures of an anonymous person’s handwriting. We are confident that this dataset contains no offensive information.

The experiment took approximately 55 minutes to run for a single test image. We ran the computations for the 1000 test images in parallel on a compute cluster with Intel Xeon E5-2667 v2, 3.30GHz cores, requesting one core each time.

## F Code assets used

Our experiments use the following dependencies which are listed alongside their license details:

1. **NumPy**[^21], which uses the BSD 3-Clause “New” or “Revised” License.
2. **jax**[^6], which uses the Apache License, Version 2.0.
3. **scipy**[^34], which uses the BSD 3-Clause “New” or “Revised” License.
4. **neural-tangents**[^29] package, which uses the Apache License, Version 2.0.

[^21]: https://numpy.org/
[^6]: https://jax.org/
[^34]: https://scipy.org/
[^29]: https://github.com/google/neural-tangents