A "Fault Bucket" for Time-Series Prediction with Uncharacterised Faulty Observations

Anonymous Author(s)
Affiliation
Address
email

Abstract

Many signals of interest are corrupted by faults of unknown type. We propose an approach that uses Gaussian processes and a general "fault bucket" to capture *a priori* uncharacterised faults, along with an approximate method for marginalising the potential faultiness of all observations. This gives rise to an efficient, flexible algorithm. We demonstrate our method's relevance to problems drawn from environmental-monitoring applications.

1 Introduction

Many applications require inference about a signal, given only observations afflicted by multiple, uncharacterised faults. We are motivated by the fast-growing field of water-quality monitoring [**?**]. Here measurements are often corrupted in non-trivial ways by various intermittent faulty sensing mechanisms, giving rise to outliers, missing data, drift and exogenous disturbances. Further, signals are not well-modelled by simple parametric approaches, such as linear or Markovian models. Despite the enormous importance of such monitoring, appropriate machine learning techniques are yet to be deployed for this purpose. In particular, there is a clear need for flexible algorithms, able to cope with signals and faults of many different types without placing a significant model-building burden upon users. Such algorithms must also be able to run reliably in real-time on incoming data.

Our proposed method will rely on Gaussian processes (GPs). GPs have been used previously for fault detection in [**?**], but in a very different context, unsuitable for our problem. Previous work along similar lines has approached this problem by creating observation models that specify the anticipated potential fault types *a priori* [**?**], but this is usually an unreasonable assumption in highly variable or poorly understood environments. Here we suggest instead the use of a catch-all "fault bucket," which can identify and treat appropriately data readings suspected of being corrupted in some way. Notably, we do not require the specification of precise fault models. The result is a fast and efficient method for data-stream prediction that can manage a wide range of faults without requiring significant domain-specific knowledge.

The collection of literature on similar topics is vast, under labels such as fault detection [**? ? ? ?**], novelty detection [**?**], anomaly detection [**?**], or one-class classification [**?**]. Unfortunately, the problems solved by most of these techniques are of very different character to our own, rendering such techniques inapplicable. Further, most of the techniques are too simple (e.g. linear or Gaussian) or fail to produce good uncertainty estimates, important for the reliable monitoring of signals. Green-tech areas, including environmental monitoring and energy-demand prediction, are still far from full automation; the provision of uncertainty estimates is necessary to allow human operators to make appropriate decisions. For this reason, we focus on developing GP-based techniques to build probabilistic nonlinear models of the signal. In addition to providing posterior probabilities of observation faultiness, we are able to perform effective prediction for the latent process even in the presence of faults.

2 Gaussian Processes

Gaussian processes provide a simple, flexible framework for performing Bayesian inference about functions [?]. A Gaussian process is a distribution on the functions $f\colon \mathcal{X} \to \mathbb{R}$ (on an arbitrary domain $\mathcal{X}$) with the property that the distribution of the function values at a finite subset of points $F \subseteq \mathcal{X}$ are multivariate Gaussian distributed. A Gaussian process is completely defined by its first two moments: a mean function $\mu\colon \mathcal{X} \to \mathbb{R}$ and a symmetric positive semidefinite covariance function $K\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. The mean function describes the overall trend of the function and is typically set to a constant for convenience. The covariance function describes how function values are correlated as a function of their locations in the domain, thereby encapsulating information about the overall shape and behavior of the signal. Many covariance functions are available to model a wide variety of anticipated signals.

Suppose we have chosen a Gaussian process prior distribution on the function $f\colon \mathcal{X} \to \mathbb{R}$, and a set of input points $\mathbf{x}$, the prior distribution on $\mathbf{f} = f(\mathbf{x})$ is

$$p(\mathbf{f} \mid \mathbf{x}, \theta) = \mathcal{N}\big(\mathbf{f}; \mu(\mathbf{x}; \theta), K(\mathbf{x}, \mathbf{x}; \theta)\big),$$

where $K(\mathbf{x}, \mathbf{x}; \theta)$ is the Gram matrix of the points $\mathbf{x}$, and θ is a vector containing any parameters required of μ and K, which form hyperparameters of the model.

Exact measurements of the latent function are typically not available. Let $f(x)$ represent the value of an observation of the signal at x and $f(x)$ represent the value of the unknown true latent signal at that point. When the observation mechanism is not expected to experience faults, the usual noise model used is

$$p(y \mid f, x, \sigma_n^2) = \mathcal{N}(y; f, \sigma_n^2), \tag{1}$$

which represents additive i.i.d. Gaussian observation noise with variance σ_n^2. Note that this model is inappropriate when sensors can experience faults, which corrupt the relationship between y and f.

With the observation model above, given a set of observations $\mathcal{D} = \big\{\big(x, f(x)\big)\big\} = (\mathbf{x}, \mathbf{f})$, the posterior distribution (itself given by a Gaussian Process) of $f_\star = f(x_\star)$ given these data is

$$p(f_\star \mid \mathbf{y}, \theta) = \mathcal{N}\big(f_\star; m(f_\star \mid \mathbf{y}, \theta), C(f_\star \mid \mathbf{y}, \theta)\big),$$

where the posterior mean and covariance are

$$\begin{aligned} m(f_\star \mid \mathbf{y}, \theta) &= \mu(x_\star; \theta) + K(x_\star, \mathbf{x}; \theta)\big(K(\mathbf{x}, \mathbf{x}; \theta) + \sigma_n^2 \mathbf{I}\big)^{-1}\big(\mathbf{y} - \mu(\mathbf{x}; \theta)\big) \\ C(f_\star \mid \mathbf{y}, \theta) &= K(x_\star, x_\star; \theta) - K(x_\star, \mathbf{x}; \theta)\big(K(\mathbf{x}, \mathbf{x}; \theta) + \sigma_n^2 \mathbf{I}\big)^{-1} K(\mathbf{x}, x_\star; \theta). \end{aligned}$$

We now make some definitions for the sake of readability. Henceforth, we assume that our observations $\mathbf{y}$ have already been scaled by the subtraction of the prior mean $\mu(\mathbf{x}; \theta)$. We will also make use of the covariance matrix shorthand $K_{m,n} = K(\mathbf{x}_m, \mathbf{x}_n)$. Finally, for now, we'll drop the explicit dependence of our probabilities on the hyperparameters θ (it will be implicitly assumed that all quantities are conditioned on knowledge of them), and will return to them later. Similarly, we drop the dependence of our probabilities on the values of inputs x, which we assume are always known.

3 Fault Bucket

We propose an algorithm that is designed to deal with faults of many different, unspecified types. We use a sequential scheme, applicable for ordered data such as time series, partitioning the data available at any point into old and new halves. We then approximately marginalise the faultiness of old observations, storing and then updating our results for future use. This gives rise to an efficient and fast algorithm. In order to effect our scheme, we make four key approximations:

1. **Fault bucket:** Faulty observations are assumed to be generated from a Gaussian noise distribution with a very wide variance.
2. **Single-Gaussian marginal:** A mixture of Gaussians, weighted by the posterior probabilities of faultiness of old data, is approximated as a single moment-matched Gaussian.
3. **Old/new independence:** We assume that noise contributions are independent, and that the contributions for new data are independent of old observations.

4. **Affine precision:** The precision matrix over both old and new halves is assumed to be affine in the precision matrix over the old half.

We will detail and justify these approximations further below.

Approximation 1 gives us a single catch-all "fault bucket". It is built upon the expectation that points that are more likely to have been generated by noise with wide variance than under the normal predictive model of the GP can reasonably be assumed to be corrupted in some way, assuming we have a good understanding of the latent process. It is hoped that a very broad class of faults can be captured in this way. To formalise this idea, we choose an observation noise distribution to replace that in (1) that models the noise as independent but not identically distributed with separate variances for the non-fault and fault cases.

$$\begin{aligned} p(z \,|\, y, x, \neg\, \text{fault}, \sigma_n^2) &= \mathcal{N}(z; y, \sigma_n^2) \\ p(z \,|\, y, x, \text{fault}, \sigma_f^2) &= \mathcal{N}(z; y, \sigma_f^2), \end{aligned}$$

where $\text{fault} \in \{0, 1\}$ is a binary indicator of whether the observation $z(x)$ was faulty and $\sigma_f > \sigma_n$ is the standard deviation around the mean of faulty measurements. The values of both σ_n and σ_f form hyperparameters of our model and are hence included in θ.

Of course, *a priori*, we do not know whether an observation will be faulty. Unfortunately, managing our uncertainty about the faultiness of all available observations is a challenging task. With N observations, there are 2^N possible assignments of faultiness; it is infeasible to consider them all.

Our solution is founded upon Approximation 2. For time series, the value to be predicted $f_\star$ typically lies in the future, and old observations are typically less pertinent for this task than new ones. We hence approximately marginalise the faultiness of old observations, representing the mixture of different Gaussian predictions (each given by a different combination of faultiness) as a single Gaussian. We prefer this approximate marginalisation over faultiness to heuristics that would designate all observations as either faulty or not; we acknowledge our uncertainty about faultiness.

More formally, imagine that we have partitioned our observations $\mathcal{D}_{a,b}$ into a set of old observations $\mathcal{D}_a = (\mathbf{x}_a, \mathbf{y}_a)$ and a set of newer observations $\mathcal{D}_b = (\mathbf{x}_b, \mathbf{y}_b)$. Define σ_a to be the (unknown) vector of all noise variances at observations $\mathbf{y}_a$, and define σ_b similarly. Because we have to sum over all possible values for these vectors, we will index the possible values of σ_a by i (each given by a different combination of faultiness over $\mathcal{D}_a$) and the values of σ_b similarly by j. We now define the covariances $V_a^i = K_{a,a} + \operatorname{diag} \sigma_a^i$, $V_b^j = K_{b,b} + \operatorname{diag} \sigma_b^j$ and $V_{a,b}^{i,j} = K_{\{a,b\},\{a,b\}} + \operatorname{diag}\{\sigma_a^i, \sigma_b^j\}$, where $\operatorname{diag} \sigma$ is the diagonal matrix with diagonal σ.

To initialise our algorithm, imagine that a identifies a small set of data, such that we can readily compute the likelihood of our hyperparameters

$$p(\mathbf{y}_a) = \sum_i p(\mathbf{y}_a | \sigma_a^i) p(\sigma_a^i) = \sum_i \mathcal{N}(\mathbf{y}_a; 0, V_a^i) p(\sigma_a^i) \tag{2}$$

and hence the hyperparameter posterior, $p(\sigma_a | \mathbf{y}_a)$. This distribution specifies the probability of our observations $\mathcal{D}_a$ being faulty; for a single observation $\mathcal{D}_a$, $p\big(\text{fault}(\mathcal{D}_a) \,|\, \mathbf{y}_a\big) = p(\sigma_a = \sigma_f \,|\, \mathbf{y}_a)$. If we were to perform predictions for some $f_\star$ using $\mathcal{D}_a$ alone, we would need to evaluate

$$p(f_\star \,|\, \mathbf{y}_a) = \sum_i p(\sigma_a^i \,|\, \mathbf{y}_a) p(f_\star \,|\, \mathbf{y}_a, \sigma_a^i) = \sum_i p(\sigma_a^i \,|\, \mathbf{y}_a) \mathcal{N}\big(f_\star; m(f_\star \,|\, \mathbf{y}_a, \sigma_a^i), C(f_\star \,|\, \mathbf{y}_a, \sigma_a^i)\big),$$

the weighted sum of Gaussian predictions made using the different possible values for σ_a. We now use Approximation 2. It is our hope that our predictions for $f_\star$ are not so sensitive to the noise in our observations that all the Gaussians in this sum become dramatically different. In any case, the quality of this approximation will improve over time—if $f_\star$ is far removed from our old data $\mathcal{D}_a$, then our predictions really will not be very sensitive to σ_a. So, we take

$$p(f_\star \,|\, \mathbf{y}_a) \simeq \mathcal{N}\big(f_\star; K_{\star,a} \tilde{V}_a^{-1} \mathbf{y}_a, K_{\star,\star} - K_{\star,a}(\tilde{V}_a^{-1} - \tilde{W}_a^{-1}) K_{a,\star} - (K_{\star,a} \tilde{V}_a^{-1} \mathbf{y}_a)^2\big),$$

where[1]

$$\tilde{V}_a^{-1} = \sum_i p(\sigma_a^i \,|\, \mathbf{y}_a)(V_a^i)^{-1},$$
$$\tilde{W}_a^{-1} = \sum_i p(\sigma_a^i \,|\, \mathbf{y}_a)(V_a^i)^{-1}\mathbf{y}_a\mathbf{y}_a^{\mathsf{T}}(V_a^i)^{-1}. \tag{4}$$

With these calculations performed, imagine receiving further data $\mathcal{D}_b$. To progress, we make Approximation 3; we assume that faults will not persist longer than $|\mathcal{D}_b|$. To be precise, we assume

$$p(\sigma_{a,b}^{i,j}, \mathbf{y}_{a,b}) \simeq p(\mathbf{y}_a)\, p(\sigma_a^i \,|\, \mathbf{y}_a)\, p(\sigma_b^j)\, p(\mathbf{y}_b \,|\, \sigma_{a,b}^{i,j}, \mathbf{y}_a) \tag{5}$$

Our predictions are now

$$p(f_\star \,|\, \mathbf{y}_{a,b}) \simeq \sum_j p(\sigma_b^j \,|\, \mathbf{y}_{a,b}) \sum_i p(\sigma_a^i \,|\, \mathbf{y}_a)\mathcal{N}\big(f_\star; m(f_\star \,|\, \mathbf{y}_{a,b}, \sigma_{a,b}^{i,j}), C(f_\star \,|\, \mathbf{y}_{a,b}, \sigma_{a,b}^{i,j})\big). \tag{6}$$

Before trying to manage these sums, we will determine $p(\sigma_b \mid \mathbf{y}_{a,b})$. As before, this distribution gives us the probability of the observations $\mathcal{D}_b$ being faulty. For example, if we have only a single observation $\mathcal{D}_b$, $p\big(\mathrm{fault}(\mathcal{D}_b) \,|\, \mathbf{y}_{a,b}\big) = p(\sigma_b = \sigma_f \,|\, \mathbf{y}_{a,b})$. We define

$$\tilde{m}(\mathbf{y}_b \,|\, \mathbf{y}_a) = K_{b,a}\tilde{V}_a^{-1}\mathbf{y}_a$$
$$\tilde{C}(\mathbf{y}_b \,|\, \mathbf{y}_a, \sigma_b) = V_b - K_{b,a}(\tilde{V}_a^{-1} - \tilde{W}_a^{-1})K_{a,b} - \tilde{m}(\mathbf{y}_b \,|\, \mathbf{y}_{a,b})^2,$$

where both $\tilde{V}_a$ (or its Cholesky factor) and $\tilde{W}_a^{-1}$ were computed previously. By using Approximations 2 and 3,

$$p(\sigma_b \,|\, \mathbf{y}_{a,b}) = \frac{\sum_i p(\mathbf{y}_b \,|\, \mathbf{y}_a, \sigma_{a,b}^i)p(\mathbf{y}_a, \sigma_{a,b}^i)}{p(\mathbf{y}_{a,b})} \simeq \frac{\mathcal{N}\big(\mathbf{y}_b; \tilde{m}(\mathbf{y}_b \,|\, \mathbf{y}_a), \tilde{C}(\mathbf{y}_b \,|\, \mathbf{y}_a, \sigma_b)\big)p(\sigma_b)}{p(\mathbf{y}_b \,|\, \mathbf{y}_a)},$$

where we have

$$p(\mathbf{y}_b \,|\, \mathbf{y}_a) = \sum_i \sum_j p(\mathbf{y}_b \,|\, \mathbf{y}_a, \sigma_{a,b}^i)p(\sigma_{a,b}^{i,j} \,|\, \mathbf{y}_a) \simeq \sum_j \mathcal{N}\big(\mathbf{y}_b; \tilde{m}(\mathbf{y}_b \,|\, \mathbf{y}_a), \tilde{C}(\mathbf{y}_b \,|\, \mathbf{y}_a, \sigma_b^j)\big)p(\sigma_b^j).$$

Note that the product of $p(\mathbf{y}_b \,|\, \mathbf{y}_a)$ and $p(\mathbf{y}_a)$ (previously computed in (2)) gives the likelihood of our hyperparameters. Now, returning to (6), we will once again use Approximation 2. We aim to reuse our previously evaluated sums over i to resolve future sums over i. As we gain more data, the faultiness of old data becomes less important. We arrive at

$$p(f_\star \,|\, \mathbf{y}_{a,b}) \simeq \mathcal{N}(f_\star; K_{\star,\{a,b\}}\tilde{V}_{a,b}^{-1}\mathbf{y}_{a,b}, K_{\star,\star} - K_{\star,a}(\tilde{V}_{a,b}^{-1} - \tilde{W}_{a,b}^{-1})K_{a,\star} - (K_{\star,\{a,b\}}\tilde{V}_{a,b}^{-1}\mathbf{y}_{a,b})^2)\,,$$

where we have

$$\tilde{V}_{a,b}^{-1} = \sum_j p(\sigma_b^j \,|\, \mathbf{y}_{a,b}) \sum_i p(\sigma_a^i \,|\, \mathbf{y}_a)(V_{a,b}^{i,j})^{-1}$$
$$\tilde{W}_{a,b}^{-1} = \sum_j p(\sigma_b^j \,|\, \mathbf{y}_{a,b}) \sum_i p(\sigma_a^i \,|\, \mathbf{y}_a)(V_{a,b}^{i,j})^{-1}\mathbf{y}_{a,b}\mathbf{y}_{a,b}^{\mathsf{T}}(V_{a,b}^{i,j})^{-1}.$$

Now, using the inversion by partitioning formula [**?** , Section 2.7],

$$(V_{a,b}^{i,j})^{-1} = \begin{bmatrix} S_a^{i,j} & -S_a^{i,j}K_{a,b}(V_b^j)^{-1} \\ -(V_b^j)^{-1}K_{b,a}S_a^{i,j} & (V_b^j)^{-1} + (V_b^j)^{-1}K_{b,a}S_a^{i,j}K_{a,b}(V_b^j)^{-1} \end{bmatrix},$$

[1] Note that for $\tilde{W}_a$, explicitly computing (unstable) matrix inverses can be avoided by solving the appropriate linear equations using Cholesky factors. For $\tilde{V}_a$, we can rewrite $(A^{-1} + B^{-1})^{-1} = A(A + B)^{-1}B$. If $i \in \{0, 1\}$ (as it would be if a identified a single observation which could be either faulty or not),

$$\tilde{V}_a = V_a^0\big(p(\sigma_a^1 \,|\, \mathbf{y}_a)V_a^0 + p(\sigma_a^0 \,|\, \mathbf{y}_a)V_a^1\big)^{-1}V_a^1. \tag{3}$$

If i takes more than two values, we can simply iterate using the same technique. We can then use the Cholesky factor of $\tilde{V}_a$ to compute our required equations.

where $S_a^{i,j} = (V_a^i - K_{a,b}(V_b^j)^{-1}K_{b,a})^{-1}$. Note that $(V_{a,b}^{i,j})^{-1}$ is affine in $S_a^{i,j}$, so that when $V_a \gg K_{a,b}V_b^{-1}K_{b,a}$, $(V_{a,b}^{i,j})^{-1}$ is effectively affine in $(V_a^i)^{-1}$. This is true if given $\mathcal{D}_b$, it is impossible to accurately predict $\mathcal{D}_a$. This might be the case if $\mathcal{D}_a$ represents a lot of information relative to $\mathcal{D}_b$ (if, for example, $\mathcal{D}_a$ is our entire history of observations where $\mathcal{D}_b$ is simply the most recent observation), or if $\mathcal{D}_b$ and $\mathcal{D}_a$ are simply not particularly well correlated. On this basis, we make Approximation 4. Additionally noting that $\sum_i p(\sigma_a^i \,|\, \mathbf{y}_a) = 1$, we have [2]

$$\tilde{V}_{a,b}^{-1} \simeq \sum_j p(\sigma_b^j \,|\, \mathbf{y}_{a,b}) \begin{bmatrix} \tilde{V}_a & K_{a,b} \\ K_{b,a} & V_b^j \end{bmatrix}^{-1},$$

$$\tilde{V}_{a,b} \simeq \begin{bmatrix} \tilde{V}_a & K_{a,b} \\ K_{b,a} & \tilde{V}_{b|a} + K_{b,a}\tilde{V}_a^{-1}K_{a,b} \end{bmatrix}$$

$$\tilde{V}_{b|a}^{-1} = \sum_j p(\sigma_b^j \,|\, \mathbf{y}_{a,b})(V_b^j - K_{b,a}\tilde{V}_a^{-1}K_{a,b})^{-1}\,.$$

Note that the lower right hand element of $\tilde{V}_{a,b}$ defines the noise variance to be associated with observations $\mathcal{D}_b$. In effect, we represent each observation as having a known variance lying between σ_n^2 and σ_f^2. The more likely an observation's faultiness, the closer its assigned variance will be to the (large) fault variance and the less relevant it will become for inference about the latent process. This approximate observation is then used for future predictions; we need never consider the full sum over all observations.

We now turn to $\tilde{W}_{a,b}^{-1}$. Unfortunately, even if $V_a \gg K_{a,b}V_b^{-1}K_{b,a}$, $\tilde{W}_{a,b}^{-1}$ is quadratic in $(V_a^i)^{-1}$. We will nonetheless again make Approximation 4 and assume that $\tilde{W}_{a,b}^{-1}$ is affine in $(V_a^i)^{-1}$. The quality of our approximation for $\tilde{W}_{a,b}^{-1}$ is much less critical than for $\tilde{V}_{a,b}^{-1}$, because the former only influences the variance of our predictions for the current predictant; any flaws in that approximation will not be propagated forward. Further, of course, if one probability dominates, $p(\sigma_a^i \,|\, \mathbf{y}_a) \gg p(\sigma_a^{i'} \,|\, \mathbf{y}_a), \forall i' \neq i$, then the approximation is valid. With this,

$$\tilde{W}_{a,b}^{-1} \simeq \sum_j p(\sigma_b^j \,|\, \mathbf{y}_{a,b}) \begin{bmatrix} \tilde{V}_a & K_{a,b} \\ K_{b,a} & V_b^j \end{bmatrix}^{-1} \mathbf{y}_{a,b}\,\mathbf{y}_{a,b}^{\mathsf{T}} \begin{bmatrix} \tilde{V}_a & K_{a,b} \\ K_{b,a} & V_b^j \end{bmatrix}^{-1}.$$

If we now receive further data $\mathcal{D}_c$, our existing data is simply treated as old data ($a \leftarrow \{a, b\}$, $b \leftarrow c$), and another iteration of our algorithm performed.

3.1 Discussion

We return to the management of our hyperparameters θ. Unfortunately, analytically marginalising θ is impossible. Most of the hyperparameters of our model can be set by optimising their likelihood on a large training set, giving a likelihood close to a delta function. This is not true of the hyperparameters σ_n and σ_f, due to exactly the same problematic sums discussed earlier. Instead, we marginalise these hyperparameters online using Bayesian Monte Carlo [**?** , Chapter 7], taking a fixed set of samples in their values and using the hyperparameter likelihoods $p(\mathbf{y}_{a,b})$ to construct weights over them. Essentially, we proceed as described above independently in parallel for each sample, and combine the predictions from each in a final weighted mixture for prediction. Note that we can use a similar procedure [**?**] to determine the full posterior distributions of σ_n and σ_f, if desired. It would be desirable to use non-fixed samples, but, unfortunately, this would require reconstructing our full covariance matrix from scratch each time a sample is moved.

The proposal above can be extended in several ways. First, we may wish to sum over more than one fault variance, which could be useful if observations are prone to faultiness in more than one mode. If, instead of summing over a small number of known variances, we wished to marginalise with respect to a density over noise variance, we can simply replace the sums over i and j with appropriate

[2] $\tilde{V}_{b|a}^{-1}$ can be computed using the same trick as in (3) if b identifies a single observation and $j \in \{0, 1\}$. The Cholesky factor of $\tilde{V}_{a,b}$ required to solve the linear equations for our predictions can be efficiently determined [**?** , Appendix B] using the previously evaluated Cholesky factor of $\tilde{V}_a$.

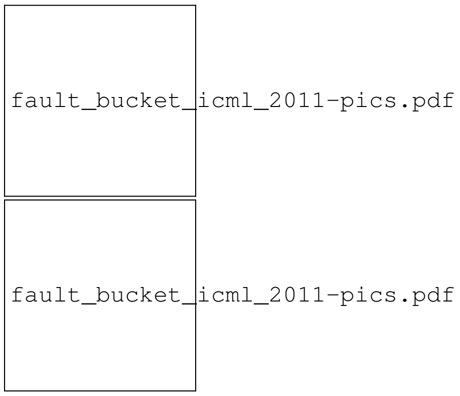
fault_bucket_icml_2011-pics.pdf
fault_bucket_icml_2011-pics.pdf

Figure 1: Mean and $\pm 3\sigma$ standard-deviation bounds for the predictions of the fault-bucket algorithm on (top), the painting dataset and (bottom), the "fishkiller" dataset. Detected faults are marked in black crosses, and the unobserved true values are marked in grey circles.

integrals. Obviously this will only be analytically possible if the posteriors for σ_a take appropriate, simple forms. In such a way our algorithm might tackle the general problem of heteroskedasticity.

Our proposed algorithm steps through our data one at a time, so that $\mathcal{D}_b$ always contains only a single observation. With Approximation 3, this means that our algorithm is not expecting faults to last more than a single observation. The results that follow, however, will show that we can nonetheless manage sustained faults. It would also be possible to step in larger chunks, evaluating larger sums. Although more computationally demanding, this might be expected to improve results. It would also allow us to consider non-diagonal noise contributions.

We have so far not specified our prior for faultiness (as expressed by $p(\sigma_a)$ and $p(\sigma_b)$). Within this paper, we consider exclusively a time-independent probability of faultiness, but our framework does not necessarily require this to be so.

In some contexts it might be useful to perform inference about the fault contribution, rather than the signal of interest. To do so, we merely switch the roles of the fault and non-fault contributions. Note that, using our full posteriors for faultiness, we can also trivially use Bayesian decision theory to make hard decisions as required.

4 Results

We test the effectiveness of the fault bucket algorithm on several time-series that are indicative of problems found in environmental monitoring. In particular, we test on water-level readings; such data are often characterised by complex dynamics and will therefore provide a good indicator of our algorithm's performance on real-world tasks. We aim to improve upon the simple, human-supervised approaches to fault detection used in this field [**?**]. For a quantitative assessment, we used two semi-synthetic datasets where a typical fault has been injected into clean sensor data. We then analyzed qualitative performance on two real data sets with actual faults. All measurements are given in meters, with samples spaced in increments of approximately 30 minutes.

Our first synthetic example, a bias fault, concerns a simple sensor error where measurements are temporarily adjusted by a constant offset, but otherwise remain accurate. This could happen if the sensor undergoes physical trauma which results in a loss of calibration. The next dataset contains a synthetic anomaly where the water level rises quickly, but smoothly, before returning back to normal. This would be indicative of a genuine environmental event such as a flash flood. The next (real) dataset contains a fault type called "painting," which is an error that occurs when ice builds on a sensor obscuring some of the readings. It is characterised by frequent sensor spikes interlaced with the original, and still accurate, signal. Our final (real) dataset, which we dub "fishkiller", comes from a sensor near a dam on a river in British Columbia, Canada. It contains an otherwise normal water level-reading that is occasionally interrupted by a short period of rapid oscillation. This occurs when dam operators open and close the floodgates too quickly, leading to rapid water level drops followed by salmon becoming stranded and suffocating. Detecting these events is critical to proper regulation of dams.

We implemented the algorithm described in Section 3 in MATLAB to address the task of 1-step-lookahead time-series prediction. A sliding window of size 100 was used to predict the value of the next observation. Each dataset was recentered so that a zero prior mean function was appropriate, and the functions were all modeled using a Matérn covariance with parameter $\nu = 5/2$ [**?**]. The hyperparameters for this covariance, including the normal observation noise variance σ_n^2, were learned using training data nearby but not included in the test datasets. The unknown fault noise variance σ_f^2 was marginalised using Bayesian Monte Carlo, with 7 samples in this parameter. The prior probability of an observation being faulty was set to a constant value of 1% throughout.

We tested against a number of different methods in order to establish the efficacy of the fault bucket algorithm. All GP based approaches used the same hyperparameters employed by our algorithm. The training set used to learn those hyperparameters was also supplied to other methods for their respective model learning phases. Several methods identify a new observation y as a fault if

$$|y - m(y \,|\, \mathbf{y})| > 3\sigma_T \,, \tag{7}$$

where $m(y \,|\, \mathbf{y})$ is the method's *a priori* prediction for y, and σ_T is the noise standard deviation on the faultless training set. Of course, methods using (7) or similar can not provide the posterior probability of a point's faultiness, as our algorithm can. Methods tested include:

XGP: A GP in which we exhaustively search over the faultiness of the last 10 points, and approximate the noise variance of all previous points in the window as having the value $\sigma_f p(\text{fault} \,|\, \mathbf{y}) + \sigma_n p(\neg\, \text{fault} \,|\, \mathbf{y})$, fixed at the time the point was observed (when data $\mathcal{D}$ was available). Clearly, this method is very much more computationally expensive than the fault bucket algorithm (roughly 2^9 times more), but offers a useful way to quantify the influence of Approximations 2-4.

TGP: A GP in which a point was flagged as a fault using (7); if faulty, a point was treated as having noise variance σ_f.

MLH: The most likely heteroscedastic GP [**?**]. Note that for this method we perform retrospective prediction (so that all data is available to make predictions about even the first predictant), as the method is intended to be used. Clearly this allows the method a predictive advantage relative to sequential methods, and the multiple passes over the data effected by MLH cannot be readily applied to the sequential problem without requiring a great deal of expensive computation.

EKF: An autoregressive neural net trained with the extended Kalman filter to capture nonstationarity. Again, (7) was used to identify and discard faulty data.

Figures **??** and 1 show the performance of the fault-bucket algorithm on the four datasets, as well as the performance of other tested methods on the synthetic datasets for comparison. In all cases, the fault-bucket algorithm did an excellent job of identifying faults when they occur, and made excellent predictions. The faults in the synthetic bias, painting, and fishkiller datasets were identified almost perfectly, with a small number of false negatives in the latter two. Most of the fault in the synthetic anomaly dataset was identified; however, the algorithm required a number of readings at the beginning of the smooth fault interval before it was able to conclude conclusively that the sharp rise was not part of the normal signal.

Table 1 displays quantitative measures of performance for the various algorithms on the synthetic datasets. In addition to superior predictive performance, our detection rates for the faulty points is generally excellent. The results reveal that Approximations 2-4 do not result in significant loss of performance compared to exhaustive search.

5 Conclusion

We have proposed a novel algorithm, the "fault bucket," for managing time-series data corrupted by faults of type unknown ahead of time. Our chief contribution is a sequential algorithm for marginalising the faultiness of observations in a GP framework, allowing for fast, effective prediction in the presence of unknown faults and the simultaneous detection of faulty observations.

Table 1: Quantitative comparison of different algorithms on the synthetic datasets. For each dataset, we show the mean squared error (MSE), the log likelihood of the true data ($\log p(\mathbf{y} \mid \mathbf{x})$), and the true-positive and false-positive rates of detection for faulty points (TPR and FPR), respectively, with all methods permitted a 'burn-in' period of 50 points. The best value for each set of results is highlighted in bold.

	Bias dataset				Change-in-dynamics dataset			
Method	MSE	$\log p(\mathbf{y} \mid \mathbf{x})$	TPR	FPR	MSE	$\log p(\mathbf{y} \mid \mathbf{x})$	TPR	FPR
FB	**0.024**	334	**0.997**	0.031	0.069	-5.77×10^3	**0.829**	0.016
XGP	0.037	**439**	0.982	**0.022**	**0.042**	$\mathbf{-1.52 \times 10^3}$	0.805	**0.012**
TGP	0.033	278	**0.997**	0.031	0.075	-8.29×10^3	**0.829**	0.083
MLH	0.940	-5.43×10^7	0.065	0.031	2.369	-2.27×10^7	0.045	0.262
EKF	0.060	-1.26×10^4	0.551	0.258	0.613	-1.81×10^4	0.169	0.768

References

[1] R. J. Wagner and US Geological Survey. *Guidelines and standard procedures for continuous water-quality monitors: Station operation, record computation, and data reporting*. US Department of the Interior, US Geological Survey, 2006.

[2] L. Eciolaza, M. Alkarouri, N. D. Lawrence, V. Kadirkamanathan, and P. J. Fleming. Gaussian Process Latent Variable Models for Fault Detection. In IEEE *Symposium on Computational Intelligence and Data Mining*, pages 287—292, 2007.

[3] R. Garnett, M. A. Osborne, S. Reece, A. Rogers, and S. J. Roberts. Sequential Bayesian Prediction in the Presence of Changepoints and Faults. *The Computer Journal*, 53, 2010.

[4] N. de Freitas, I. M. MacLeod, and J. S. Maltz. Neural networks for pneumatic actuator fault detection. *Transactions of the South African Institute of Electrical Engineers*, 90:28–34, 1996.

[5] R. Isermann. Model-based fault-detection and diagnosis – status and applications. *Annual Reviews in Control*, 29(1):71–85, 2005.

[6] S. X. Ding. *Model-based Fault Diagnosis Techniques: Design Schemes, Algorithms, and Tools*. Springer, first edition, 2008.

[7] M. Markou and S. Singh. Novelty detection: a review – Part 1: Statistical approaches. *Signal Processing*, 83(12):2481–2497, 2003.

[8] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. ACM *Comput. Surv.*, 41:15:1–15:58, July 2009.

[9] S. Khan and M. Madden. A survey of recent trends in one class classification. In Lorcan Coyle and Jill Freyne, editors, *Artificial Intelligence and Cognitive Science*, volume 6206 of *Lecture Notes in Computer Science*, pages 188–197. Springer Berlin / Heidelberg, 2010.

[10] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning. The MIT Press, Cambridge, MA, USA, 2006.

[11] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, New York, NY , USA, 1992.

[12] M. A. Osborne. *Bayesian Gaussian Processes for Sequential Prediction, Optimisation and Quadrature*. PhD thesis, University of Oxford, 2010. Available at `www.robots.ox.ac.uk/~mosb/full_thesis.pdf`.

[13] K. Kersting, C. Plagemann, P. Pfaff, and W. Burgard. Most likely heteroscedastic Gaussian process regression. In *Proceedings of the 24th international conference on Machine learning*, pages 393–400. ACM, 2007.