

Bayesian Change Point Detection for Satellite Fault Prediction

Ryan Turner

PhD Student, University of Cambridge, Cambridge, UK
rt324@cam.ac.uk

Keywords: Machine learning, Time series, Bayesian, Change point, Reliability, Statistics.

Abstract

Change point detection makes predictions in data whose structure changes over time and belongs to a class of methods called time series methods. Machine learning is the design of algorithms whose performance improves with data. Techniques from statistics, such as change point detection, have received increased interest in recent years in machine learning. Many problems have change point structure: Stock markets exhibit change points when some significant economic event causes an increase in volatility, weather systems may exhibit change points during years of El Niño, and electronic systems show change points when devices begin to fail or configurations change.

This paper focuses on detecting changes in the operation of satellite base stations. The aim is to provide predictions of device lifetimes or time until maintenance will be necessary. Typical reliability engineering uses survival analysis to find lifetime distributions and mean time between failures (MTBF). However, using change point detection we can make predictions that include information dynamically. For instance, a change in the measured weather conditions could signal an impending satellite signal loss due to an imminent storm. Modeling changing dependencies is also important; a change in the relationship between the external and internal temperature on a device should mean there is a problem with the cooling unit. In the mentioned examples, the weather change point occurred when the weather conditions changed and when the cooling unit began to malfunction, respectively. Unlike MTBF, change point detection provides probability distributions describing the likelihood of a failure, which is necessary in most cases since there is usually significant remaining uncertainty over the time until failure. The two key aspects are that change point detection can include information dynamically and model changing dependencies.

We show how to use techniques from Bayesian statistics, namely change point detection, to model and understand electronic systems.

1 Introduction

Change point models are used to extend simple time series methods to cases where the appropriate model changes. In this paper we look at modeling sensor measurements from a satellite base station using change point methods. We provide examples where events, such as power supply problems or changes in the amplifier power, correspond to change points in measurements taken by a logging system. We do this by extending the Bayesian online change point detection (BOCPD) algorithm introduced by Adams and MacKay (2007).

This paper is organized as follows: In Section 2 we describe the notion of time series data and what it means for an algorithm to be online. In Section 3, we give some background on what machine learning is and how it is used. We also explain its relationship to statistics as well as how ideas from Bayesian statistics carry over to Bayesian machine learning. In Section 4, we explain the BOCPD algorithm both mathematically and through illustrative examples. We demonstrate the use of BOCPD on real world data, namely satellite fault logs, in Section 5.

2 Time Series Methods

Change point detection is designed to deal with time series data. The time series measurements we deal with are real valued and can be sampled at any point in time. Examples include voltage, temperature, and current measurements. In more precise terms time series data maps from a time index $t \in \mathcal{T}$ to a measurement $x_t \in \mathbb{R}$. Time series can be in discrete time ($\mathcal{T} = \mathbb{Z}$) or continuous time ($\mathcal{T} = \mathbb{R}$), which means discrete time is a special case of continuous time. For example, if we measure the voltage exactly once an hour we can model it as discrete time where the time index is how many samples have been taken so far. We call this case *uniform sampling*. However, if voltages are measured at arbitrary times or with changing sampling rates it is more convenient to model it as a continuous time process. To simplify the analysis we focus on discrete time processes.

2.1 Online Algorithms

We also assume that our application demands *online* change point detection: an autonomous robot must detect change points in its sensors as they happen, an automatic trading system must detect change points as soon as possible, and electronic monitoring applications must alert to change points in measurements as soon as possible. The alternative to online algorithms are *retrospective* algorithms. A retrospective fault detection system would be one where it waits until the end of the year, or some other period of time, and then uses all the data throughout the year to locate where the change points are. By contrast, in the online system if a change point happens in March, the system should alert as soon as it can, and not wait till the end of the year to collect all the data it can before doing any analysis. Prior to the work by Adams and MacKay (2007), and similar by Fearnhead and Liu (2007), previous Bayesian approaches to change point detection have been retrospective (Barry and Hartigan, 1993; Xuan and Murphy, 2007).

3 Machine Learning

Machine learning is the study of algorithms whose performance improves with exposure to data. The functionality of typical software can be completely specified in advance. However, in machine learning software must learn to match its output to training examples. The classic example is optical character recognizers (OCR): images of characters, say zero to nine, are provided and the software must output which digit is in the image. As opposed to trying to specify rules about what makes a two a two, example images are provided along with labels (this is known as the *training set*). A good machine learning algorithm will predict the correct character in test when novel images are provided. An OCR is an example of an *iid data set*, this paper is focused on the more complex task of *time series data*.

It was realized early on that machine learning must have closer ties to statistics than other parts of computer science; functionality was being specified by real data not abstract specifications on paper as is the case with data structures like stacks and queues. However, machine learning still maintains some independence from statistics. Machine learning is more concerned about new efficient algorithms for complex tasks while statistics is more concerned about the details of data sets. Central to machine learning is *test set* performance.

3.1 Measuring Performance

Given that machine learning is interested in predictive performance we need a way to measure the performance. There is a standard procedure used in machine learning; for simplicity, we first consider the case of iid data. For iid data, the evaluation procedure works as follows. Each data point is randomly assigned to be in either a training set of size N or a test set of size N' .¹ The models'

¹It is a common novice mistake to test a model on the same data set the model was trained on. This gives overly optimistic estimates of performance and does not tell us how well an algorithm will perform on new data.

parameters θ are set in a way specified by the model designer using the training set.² Then on each point in the test set the model makes a prediction about the data point x_i . If the model outputs a probability distribution $p(x_i)$ we convert it to an action a (or point estimate) by minimizing the expected loss

$$a_i = \operatorname{argmin}_a \mathbb{E}[L(x_i, a)] = \operatorname{argmin}_a \int L(x_i, a)p(x_i) dx_i. \quad (1)$$

This optimization is Bayes’ decision rule. Certain methods do not provide a predictive distribution p , and directly give an action a . If x is continuous ($x \in \mathbb{R}$) common loss functions are mean-square-error (MSE), $L(x_i, a) = (x_i - a)^2$, and mean-absolute-error (MAE), $L(x_i, a) = |x_i - a|$. The optimal actions for MSE and MAE are to supply the mean and median of p , respectively. For the log loss or intrinsic loss, $L(x_i, p) = -\log p(x_i)$, the optimal action is to supply p itself. The models’ performance or empirical loss \hat{L} is the average loss over each test point. See (Bishop, 2007, Ch. 1) for more details.

In the time series context, the model can be evaluated by its loss on the next step ahead prediction. In other words, predicting x_i given $x_{1:i-1}$. Theoretical perspectives on evaluating time series models is explored in Dawid (1986), most of which was inspired by evaluating probabilities in weather prediction.

3.2 Bayesian Machine Learning

A significant cohort of machine learning researchers borrow ideas from Bayesian statistics. Bayesians are willing to use probability to express uncertainty over any well defined quantity; sometimes these probabilities are interpreted as state of belief or knowledge. This is in contrast to the frequentist alternative where only samples from “repeatable” processes are treated as random and under the jurisdiction of probability theory.

This division in statistics has consequences in change point detection. Much of the older work on change point detection, dating back to Page (1955), was based on frequentist hypothesis testing ideas. Such setups are designed to signal an alert when a change point occurs; however, they only say the probability of an alert *given there is no change point* is less than α , usually 5% (this is known as a false positive rate). In Bayesian change point detection, we say the probability that there was a change point, *given the time series we observed*, is $x\%$. The substantive difference between the two statements is often puzzling to those not well versed in probability. The difference is made most clear by extreme case reasoning. There is no unique procedure in the frequentist change point detector, a change point detector that completely ignored the data and alerted randomly 5% of the time would satisfy the frequentist statement. There is no unique procedure in the Bayesian change point detector either; in the Bayesian change point detector the designer must first specify a prior probability distribution for what one would likely see given a change point. Once the prior assumptions are specified, there then exists a uniquely optimal procedure for finding the change points.

The division has a slightly different context in machine learning. Machine learning is concerned about test set performance, so we would like to specify algorithms that make the best predictions on new data. However, it is not possible to say before being given a data set what the best algorithm will be. The “no free lunch theorem” says that algorithms that perform better on one data set will lose some performance on others. Bayesian machine learning cares about achieving optimal performance on average under our prior assumptions. By contrast, frequentist analysis is typically concerned with asymptotic or minimax performance: how well does the algorithm perform in large training sets assuming nature is adversarial. In other words, how well does an algorithm perform in the worst-case; the true generating distribution has been selected to trick the algorithm.

²A parametric model will summarize what it learns from the data in the parameters. The standard example is linear regression: $y_i = ax_i + b$. The parameters are $\theta = \{a, b\}$. There are also nonparametric models where the training data cannot be completely summarized in a finite number of parameters. We will use the term parameters in both cases. There are also hyper-parameters, which are the parameters of probability distributions placed on parameters.

Bayesian methods always provide a predictive distribution p while it is common for non-Bayesian methods to just specify a single best prediction a . When a point estimate is required (1) must be used to convert the predictive distribution p to an action a .

4 Change Point Models

The BOCPD algorithm introduced in Adams and MacKay (2007) efficiently finds a distribution on the time since the last change point, namely the *run length*. BOCPD requires one to specify an underlying predictive model (UPM) and a hazard function. The UPM is used to compute $p(x_t|x_{(t-\tau):t}, \theta_m)$ (known as a posterior predictive) for any $\tau \in [1, \dots, (t-1)]$, at time t . Intuitively, the UPM is a simple model of the time series x that changes at every change point. The hazard function $H(r|\theta_h)$ describes how likely a change point is given the current run length r . We define the change point vector $c_t \in \{0, 1\}$ to be true (1) if there is a change point at time t and false (0) otherwise. The parameters $\theta := \{\theta_h, \theta_m\}$ form the set of hyper-parameters for the model, and are assumed to be fixed and known in the setup of Adams and MacKay (2007). Learning the hyper-parameters and improving computational efficiency have been addressed in Turner et al. (2009).

4.1 Example BOCPD Model

A simple example of BOCPD would be to use a constant hazard function $H(r|\theta_h) := \theta_h$, meaning $p(r_t = 0|r_{t-1}, \theta_h)$ is independent of r_{t-1} and is constant, giving rise to geometric inter-arrival times for change points. A constant hazard implies the presence of a change point at time t is determined by a weighted coin flip independent of how long it has been since the last change point. The geometric inter-arrival distribution is the discrete time analog of the exponential distribution, which is often used to model the arrival of buses or radioactive decay where events are unrelated to one another. We can use a UPM with iid Gaussian observations with changing mean and variance; we apply a distribution known as the Normal-Inverse-Gamma on the mean and variance, which is computationally advantageous:

$$x_t \sim \mathcal{N}(\mu, \sigma^2), \quad (2)$$

$$\mu \sim \mathcal{N}(\mu_0, \sigma^2/\kappa), \quad \sigma^{-2} \sim \text{Gamma}(\alpha, \beta), \quad (3)$$

here the model hyper-parameters are $\theta_m := \{\mu_0, \kappa, \alpha, \beta\}$. A sample draw from this model is shown in Figure 1. Sampling data from a model can be a good way to visualize the assumptions it makes.

4.2 The BOCPD Algorithm

This section describes the core of the BOCPD method and thus assumes a level of comfort using the manipulations of probability theory. BOCPD calculates the run length predictive distribution at time t , i.e. $p(r_t|x_{1:t})$, in a recursive fashion. At every time step t we use the last run length distribution $p(r_{t-1}|x_{1:t-1})$ to efficiently calculate $p(r_t|x_{1:t})$. We can make predictions about the next data point x_t in a way that maintains our uncertainty in the true run length, through marginalization of the run length variable:

$$p(x_{t+1}|x_{1:t}) = \sum_{r_t} p(x_{t+1}|x_{1:t}, r_t)p(r_t|x_{1:t}) \quad (4)$$

$$= \sum_{r_t} p(x_{t+1}|x_t^{(r)})p(r_t|x_{1:t}), \quad (5)$$

where $x_t^{(r)}$ refers to the last r_t observations of x , and $p(x_{t+1}|x_t^{(r)})$ is computed using the UPM. The run length distribution can be found by normalizing the joint likelihood:

$$p(r_t|x_{1:t}) = \frac{p(r_t, x_{1:t})}{\sum_{r_t} p(r_t, x_{1:t})}. \quad (6)$$

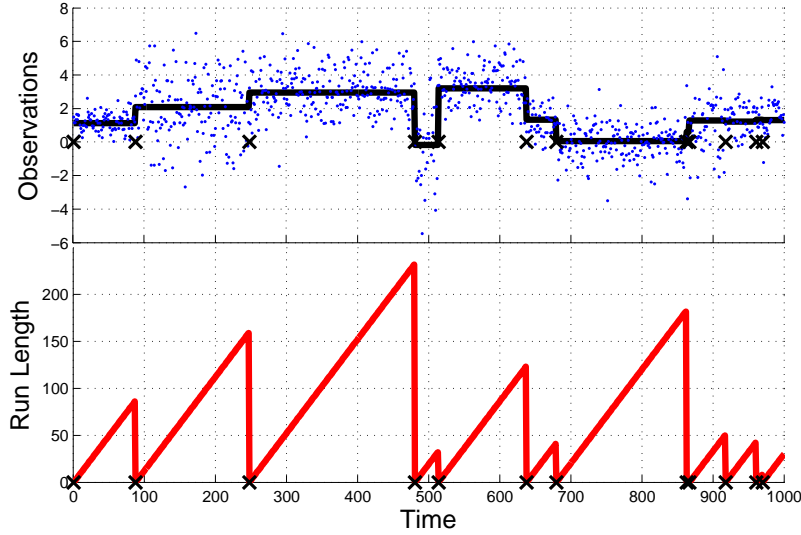


Figure 1: A sample draw from BOCPD with a Gaussian UPM and constant hazard. **Top panel:** The blue dots are the data points x_t ; at each time point t , x_t is sampled from (2). The solid black line represents the mean in each regime. Note that the mean and variance is changing in each regime; at each change point the mean and variance is a new sample from (3). **Bottom panel:** The black crosses are the change points and the solid red line is the run length.

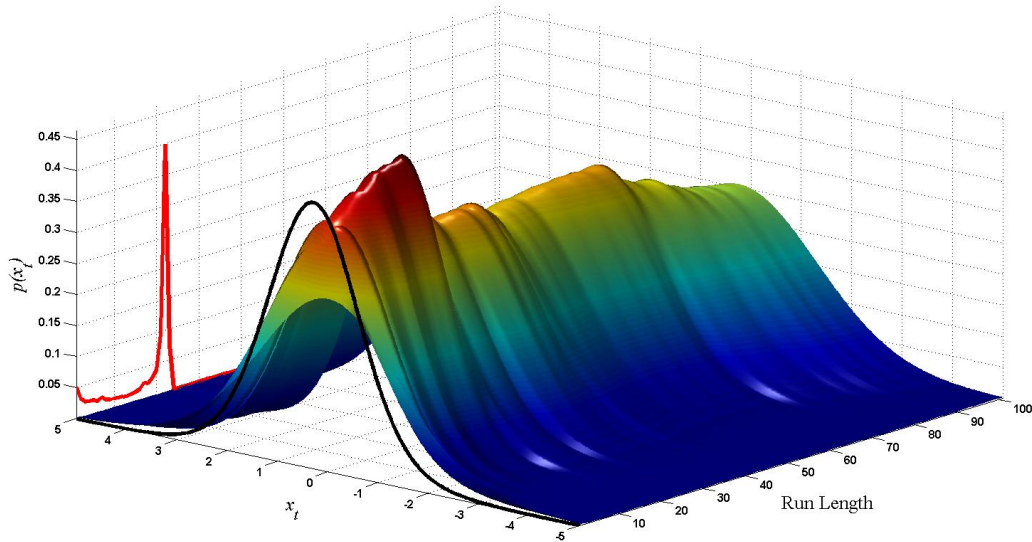


Figure 2: Demonstration of BOCPD on a sample draw with a Gaussian UPM and constant hazard. The surface plot is the predictive distribution $p(x_t|r_{t-1}, x_{1:t-1})$ conditional on the run length r_{t-1} at $t = 100$ using the UPM as in (9). The solid red line is the run length distribution $p(r_{t-1}|x_{1:t-1})$ from (6), while the solid black line is the marginal distribution $p(x_t|x_{1:t-1})$ using (5).

The joint likelihood can be updated online recursively:

$$\gamma_t := p(r_t, x_{1:t}) = \sum_{r_{t-1}} p(r_t, r_{t-1}, x_{1:t}) \quad (7)$$

$$= \sum_{r_{t-1}} p(r_t, x_t | r_{t-1}, x_{1:t-1}) p(r_{t-1}, x_{1:t-1}) \quad (8)$$

$$= \sum_{r_{t-1}} \underbrace{p(r_t | r_{t-1})}_{\text{hazard}} \underbrace{p(x_t | r_{t-1}, x_t^{(r)})}_{\text{UPM}} \underbrace{p(r_{t-1}, x_{1:t-1})}_{\gamma_{t-1}}. \quad (9)$$

This defines a forward *message passing* (Pearl, 1982) scheme to recursively calculate γ_t from γ_{t-1} . The conditional can be restated in terms of messages as $p(r_t | x_{1:t}) \propto \gamma_t$. The calculations in this section are illustrated visually in Figure 2. All the distributions mentioned so far are implicitly conditioned on the set of hyper-parameters θ .

The example in Section 4.1 used a UPM that assumed the data was iid in each regime. Gaussian processes (GPs) (Rasmussen and Williams, 2006) are probability distributions over generic functions. BOCPD has been combined with Gaussian process time series models in Saatci et al. (2010). While Turner et al. (2009) extended BOCPD to learn the hyper-parameters θ using the time series $x_{1:T}$ only (unsupervised), Turner and Rasmussen (2010) incorporates labeled data for when the change points occur $c_{1:T}$ to better learn θ .

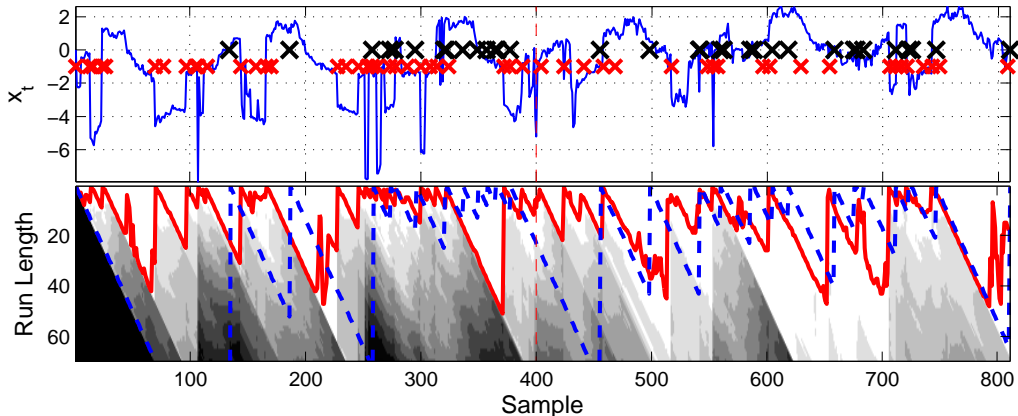


Figure 3: **Modem Data**: the top panel shows the modem transmission strength in blue. The labeled change points, the power supply alarm, are shown as black crosses. The red crosses represent alerts, which are placed when the probability of a change point since the last alert exceeds 95%. The bottom panel shows the log run length distribution $p(r_t | x_{1:t})$ (lighter means higher probability). The red line indicates the median of the distribution while the dashed blue line is the labeled run length. Everything before the vertical dashed line is used to train the hyper-parameters θ .

5 Satellite Data

In this section we look at fault data from a satellite base station provided by a major satellite communications corporation. We use faults or alarms as change points c and sensor measurements as x when training BOCPD. In this context, the run length is the time since the last fault, which might be useful for faults that are not always easily detectable. We can also provide a distribution on time until next change points as in a prediction on time until the next fault using the hazard function learned in training. We consider both unsupervised, ignores the known change points in training (Turner et al., 2009), and supervised, uses the change points in training (Turner and Rasmussen, 2010), to train the hyper-parameters θ .

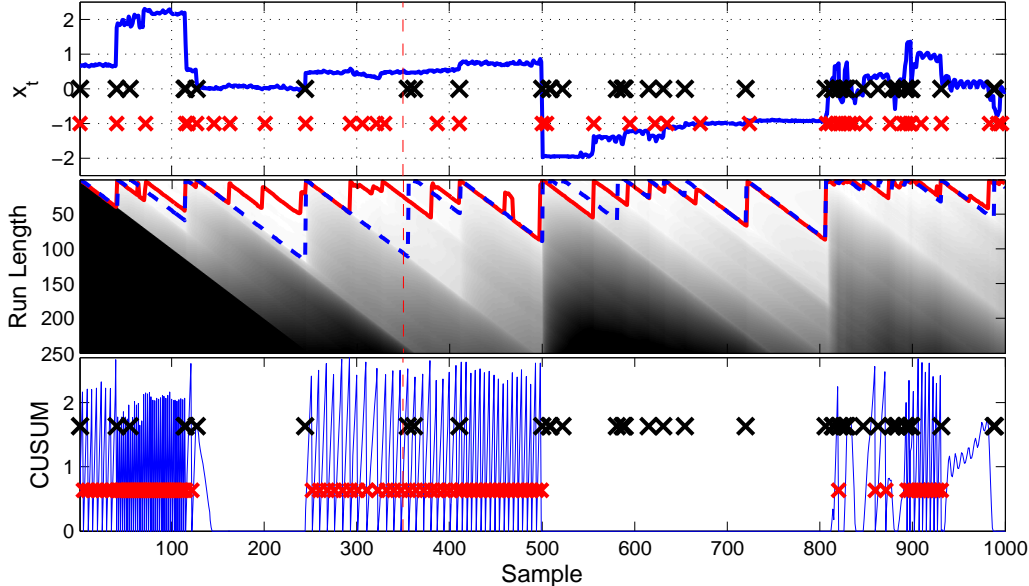


Figure 4: **Amplifier Data**: the top panel shows the amplifier power in blue. The labeled change points are shown as black crosses. The graphical conventions are the same as those in Figure 3. The bottom panel shows the output of CUSUM. The red crosses are the CUSUM alert and the blue line is the CUSUM test statistic S_t , which was reset to zero at each alert.

In the first example, we use alarms from a power supply as the labeled change points c and a modem transmission strength as the time series x . The variables share only a tenuous link and the two are not obviously related upon visual inspection. However, supervised BOCPD still manages to find a signal. The time series was differenced and standardized prior to training. We show the run length distribution of BOCPD and the time series in Figure 3.

We trained on the first 350 time points and tested on the remaining 460 points in Table 1; in all of the experiments we compare our results to the time independent model (TIM) as a reference. In data space prediction the TIM model corresponds to modeling the data as iid Gaussian. In the run length space the TIM model corresponds to assuming a constant hazard and an uninformative UPM $p(x_t) \propto 1$, which gives a geometric run length distribution in large t . We also compared to the CUSUM method (Page, 1955), which is a classic frequentist change point detection method. The CUSUM maintains a test statistic S_t , which one uses to alert when it crosses a threshold. We set

Table 1: The results are provided with 95% error bars and the p-value testing the null hypothesis that methods are equivalent to the best performing method, according to NLL, using a one sided t-test. The first three columns represent the loss in predicting the run length (using $p(r_t|x_{1:t})$) while the final column shows the loss in data space of predicting the next data point (using $p(x_{t+1}|x_{1:t})$). The units of NLL is nats/observation. For continuous variables (data space prediction) we shift the NLL so the best method has NLL zero. The MAE and MSE have the units of t and t^2 , respectively, in run length prediction. Variants of BOCPD are marked with \star .

| Modem Data (350 Training Points, 460 Test Points) | | | | | | | | |
|---|------------------------------------|---------|------------------------------------|---------|-----------------------------------|---------|---------------------------------|---------|
| Method | NLL $\times 10^1$ | p-value | MAE $\times 10^0$ | p-value | MSE $\times 10^{-2}$ | p-value | NLL $\times 10^1$ | p-value |
| CUSUM | 46.45 \pm 0.24 | <0.0001 | 28.8 \pm 1.3 | <0.0001 | 11.85 \pm 0.78 | <0.0001 | N/A | N/A |
| TIM | 41.30 \pm 0.49 | 0.0690 | 15.50 \pm 0.92 | 0.1417 | 5.30 \pm 0.37 | <0.0001 | 7.24 \pm 0.87 | <0.0001 |
| Unsupervised \star | 133 \pm 15 | <0.0001 | 18.3 \pm 1.7 | 0.0002 | 6.22 \pm 0.97 | 0.0002 | 0.1 \pm 1.3 | 0.4617 |
| Supervised \star | 40.74 \pm 0.55 | N/A | 14.77 \pm 0.97 | N/A | 4.33 \pm 0.35 | N/A | 0.0 \pm 1.4 | N/A |
| Amplifier Data (350 Training Points, 650 Test Points) | | | | | | | | |
| CUSUM | 44.28 \pm 0.26 | <0.0001 | 22.57 \pm 0.95 | <0.0001 | 8.48 \pm 0.54 | <0.0001 | N/A | N/A |
| TIM | 44.28 \pm 0.37 | <0.0001 | 21.79 \pm 0.94 | <0.0001 | 11.02 \pm 0.65 | <0.0001 | 37.7 \pm 1.4 | <0.0001 |
| Unsupervised \star | 103 \pm 16 | <0.0001 | 11.4 \pm 1.4 | 0.0183 | 3.97 \pm 0.77 | 0.0032 | 0.0 \pm 1.4 | 0.9530 |
| Supervised \star | 37.0 \pm 1.1 | N/A | 9.6 \pm 1.1 | N/A | 2.66 \pm 0.56 | N/A | 1.5 \pm 1.2 | N/A |

the parameters of the CUSUM as recommended by Grigg and Spiegelhalter (2008). The threshold is chosen to control the false positive rate. However, this does not provide predictions needed for evaluation. Consequently, the output of CUSUM in training was fit to the labeled run length using a linear model.

We also consider a time series of amplifier power. The correspondence between the change points and the time series is much closer in this time series than in the modem transmission strength. We see in Figure 4 that the median of the run length distribution almost perfectly matches the true run length. We also show the output of CUSUM on the same time series. It alerts far too often, it seems to have trouble with the recurrent nature of the change points in this problem. Note that the CUSUM is guaranteed not to alert more than 5% of the time if there is no change point. However, it does not say that it will find where the change points are in the event of recurring change points. We see that this is a problem for CUSUM in Figure 4. Quantitative results are shown in Table 1.

6 Conclusions

Change point detection is a widely applicable machine learning technique and we have shown it has many uses when applied to monitoring electronic systems. BOCPD presents a general method that allows for many other simpler models to be used as UPs. It is therefore a very important part of time series analysis and machine learning. We trained and evaluated BOCPD from the machine learning perspective, we set the hyper-parameters according to training data and checked its predictive performance on a separate test set. We have shown that just controlling a false positive rate is insufficient for good predictive performance in complex change point problems, such as satellite fault logs. There is much room for extensions within the framework provided by BOCPD.

References

- Adams, R. P. and MacKay, D. J. C. (2007). Bayesian online changepoint detection. Technical report, University of Cambridge, Cambridge, UK.
- Barry, D. and Hartigan, J. A. (1993). A Bayesian analysis of change point problems. *Journal of the American Statistical Association*, 88:309–319.
- Bishop, C. M. (2007). *Pattern Recognition and Machine Learning*. Springer, 1 edition.
- Dawid, A. P. (1986). Probability forecasting. *Encyclopedia of Statistical Sciences*, 7:210–218.
- Fearnhead, P. and Liu, Z. (2007). Online inference for multiple changepoint problems. *Journal of the Royal Statistical Society*, 69:589–605.
- Grigg, O. A. and Spiegelhalter, D. J. (2008). An empirical approximation to the null unbounded steady-state distribution of the cumulative sum statistic. *Technometrics*, 4:501–511.
- Page, E. S. (1955). A test for a change in a parameter occurring at an unknown point. *Biometrika*, 523–527:42.
- Pearl, J. (1982). Reverend Bayes on inference engines: A distributed hierarchical approach. In *AAAI*, pages 133–136, Pittsburgh, PA, USA.
- Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. The MIT Press.
- Saatci, Y., Turner, R., and Rasmussen, C. E. (2010). Gaussian process change point models. In *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel.
- Turner, R. and Rasmussen, C. E. (2010). Supervised Bayesian online change point detection. In Press.
- Turner, R., Saatci, Y., and Rasmussen, C. E. (2009). Adaptive sequential Bayesian change point detection. In Harchaoui, Z., editor, *Temporal Segmentation Workshop at NIPS 2009*, Whistler, BC, Canada.
- Xuan, X. and Murphy, K. (2007). Modeling changing dependency structure in multivariate time series. In *Proceedings of the 24th International Conference on Machine Learning*, pages 1055–1062, Corvallis, OR, USA.