# Change-Point Detection for Monitoring Clinical Decision Support Systems with a Multi-Process Dynamic Linear Model

Siqi Liu*, Adam Wright†, Dean F. Sittig‡ and Milos Hauskrecht*
*Department of Computer Science
University of Pittsburgh, Pittsburgh, Pennsylvania
Email: {siqiliu,milos}@cs.pitt.edu
†Brigham and Women's Hospital and Harvard Medical School
Boston, Massachusetts
Email: awright@bwh.harvard.edu
‡School of Biomedical Informatics
University of Texas Health Science Center at Houston, Houston, Texas
Email: dean.f.sittig@uth.tmc.edu

***Abstract*—A clinical decision support system and its components may malfunction due to different reasons. The objective of this work is to develop computational methods that can help us to monitor the system and assure its proper operation by promptly detecting and analyzing changes in its behavior. We develop a new change-point detection method using the Multi-Process Dynamic Linear Model. The experiments on real and simulated data show that our method outperforms existing change-point detection methods, leading to higher accuracy and shorter delay in the detection.**

## I. Introduction

A clinical decision support system (CDSS) is a complex computerized system that assists a physician in managing patients. Ideally, the system would function optimally all the time. However, in reality the system may be prone to various kinds of malfunctions affecting its performance [1]. Developing tools that are able to detect early and reliably its malfunctions is critical to preserve its intended function and to eliminate potential patient-related risks. In this work we study the problem of detecting abnormal changes in the monitoring and alerting component from its alert rule statistics.

The monitoring and alerting component identifies and alerts on clinical and patient conditions in the regular clinical workflow. Typically it represents and executes expert-defined rules. If the condition of the rule is satisfied, it sends a reminder or alert to the physician. The proper execution of the rules depends on multiple factors. First, they rely on the information from patients' electronic health record (EHR). As a result, any change in how information is collected, stored, or coded in the EHR may affect the rule. Second, the rule activation and screening are typically triggered by other parts of the CDSS, so any updates or changes made to other components may affect the rule activation. Finally, the rules in the CDSS are maintained by humans, and any error (unintentionally) introduced in the rule logic may change the intended effect.

To detect changes of the monitoring and alerting component, it would be ideal to have measurements on different components of the CDSS. However, in reality, it is not feasible to have all these data. Our aim is to detect changes based solely on the firing counts of the rules in the system. The firing counts can be subjected to different sources of variations. Some of these sources may be identifiable. For example, the different days of the week (e.g., Sunday vs. Monday) may lead to different rule firing counts for the same rule due to the differences in the weekly clinical workflow. But many other possible sources are hard to identify. For example, changes in the population of patients screened by the rules over time may cause an increase or a decrease of the number of alerts. It is a challenge to develop a method that can distinguish real changes from the noise or a natural variation. Meanwhile, the method should work in real-time, i.e. the detection is done with a minimum delay after the change occurs.

To tackle these challenges, we study change-point detection methods [2]–[4]. Traditional change-point detection methods aim to detect changes by comparing the behavior or the statistics before and after the change. Unfortunately, these methods typically do not account for the different sources of variations, and they also assume the analysis is retrospective (instead of real-time), i.e. the changes are detected after all data have been collected, which means the delay of the detection is not a concern. Although nonparametric decomposition has been used to account for seasonal variations [5], it does not have a intuitive generative model for different behaviors of the data. Therefore, we develop a new method using the Multi-Process Dynamic Linear Model (MPDLM [6]) consisting of multiple models, which not only account for seasonal variations, but also represent different dynamics that may drive the observed data and lead to normal or abnormal behaviors. We devise a probabilistic score that can at any time step assess the chance the time series has changed, which works in real-time and

does not rely on any future observations.

## II. Method

The idea behind our approach is to represent the time series with an accurate probabilistic model, and then use multiple models to cover normal and abnormal behaviors. We start with a brief review of the Dynamic Linear Model (DLM), which is used to model the time series. Specifically, we introduce a special form of DLM that allows us to model seasonal (weekly) variations. Then, we show how to build DLMs reflecting normal and abnormal behaviors and combine them into the Multi-Process Dynamic Linear Model (MPDLM).

### A. Dynamic Linear Model

The DLM models a sequence of real-valued observations $\{y_t : t = 1, 2, \cdots\}$ using a sequence of real-valued hidden state vectors $\{x_t : t = 1, 2, \cdots\}$ of dimension $d$. The dynamics of the model is captured by:

$$\begin{aligned} y_t &= Fx_t + v, \quad v \sim N(0, V), \\ x_t &= Gx_{t-1} + w, \quad w \sim N(0, W). \end{aligned} \tag{1}$$

where $G$ is a transition matrix that models the change in the hidden state over time, and $F$ is an emission matrix that reflects the expression of observations $y_t$ given the current $x_t$. Both transition and emission are stochastic and corrupted by a zero-mean Gaussian noise ($w$ and $v$) with covariance $W$ and $V$. At the beginning ($t = 0$), we assume the hidden state $x_0 \sim N(m_0, C_0)$, where $m_0$ and $C_0$ is the mean and covariance matrix of $x_0$ respectively.

The DLM is very flexible in that it can define many different behaviors of the time series including seasonality. We use this property to represent weekly variations present in our rule firing count time series. The best way to understand the seasonal DLM is to break the hidden state ($x_t$) into multiple components: a baseline ($u_t$) defining the mean, a slope ($l_t$) defining the trend of the mean, and a seasonal component ($s_t$) defining the change in the mean for each phase (a day in a week) of a seasonal cycle (a week). Let $p$ denote the period of the cycle. Then we can define a function of time $[t]_p = (t + p - 1) \bmod p + 1$ that maps the time to its corresponding phase. We construct $x_t$ as a vector consists of the components:

$$x_t = (u_t, l_t, s^{([t]_p)}, s^{([t-1]_p)}, \ldots, s^{([t-p+2]_p)})^T. \tag{2}$$

and correspondingly the transition and emission matrices:

$$F = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & \ldots & 0 \end{bmatrix}_{1 \times d},$$

$$G = \begin{bmatrix} 1 & 1 & 0 & 0 & \ldots & 0 & 0 \\ 0 & 1 & 0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & -1 & -1 & \ldots & -1 & -1 \\ 0 & 0 & 1 & 0 & \ldots & 0 & 0 \\ 0 & 0 & 0 & 1 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \ldots & 1 & 0 \end{bmatrix}_{d \times d}. \tag{3}$$

### B. Multi-Process Dynamic Linear Model

The seasonal DLM reflecting the weekly variation can be used to model the rule firing counts (after transformation). However, our main objective is to use the model for detecting changes in the time series. We address the problem by defining multiple DLMs representing normal and different abnormal behaviors.

One way to define a collection of related DLMs is to use the MPDLM [6]. Suppose we have a collection of DLMs. Each DLM $i$ has its own parameters $(F^{(i)}, G^{(i)}, V^{(i)}, W^{(i)})$. In the MPDLM, the individual DLMs may switch in time depending on which model currently drives the time series. Let $M_t^{(i)}$ be a random variable indicating whether model $i$ is driving the time series at time $t$ and generating $y_t$, and $M_t$ be a vector composed of $M_t^{(i)}$ for all $i$.

In this work, we use a combination of three DLMs: MS (Model Stable), MAO (Model Additive Outlier), and MLS (Model Level Shift). MS is a model for normal time series behavior. MAO represents a spike behavior in which the most recent observation is very different from previous normal observations, and this difference is limited to just one point. MLS represents a baseline-shift behavior in which the new observations are very different from the previous observations, and the changes persist for a longer time. We design these models by varying their covariance matrices $V$ and $W$. Take MS as the reference model. MAO has a much higher value for the covariance $V$, because a spike is treated as a temporary noise in the observation and has no influence on the hidden state of the time series. On the other hand, MLS has a much higher value for the covariance $W$.

Now let $Y_t$ denote the time-series observations up to time $t$, i.e. $Y_t = \{y_u : u = 1, 2, \ldots, t\}$. Given the MPDLM and its parameters, for each DLM $i$, we can calculate the prior probability of $i$ driving the time series right before observing $y_t$, $p(M_t^{(i)} = 1|Y_{t-1})$. and the posterior probability after observing $y_t$, $p(M_t^{(i)} = 1|Y_t)$. The probabilities enable us to infer if there was a switch in the current model and hence a change in the time series. To keep the inference tractable, we use *Kalman filter* with *collapsing* [6], [7].

Notice that if we observe an abnormal $y_t$ for the first time, there is no way we can tell whether MAO or MLS has generated it, because it could be explained by either the noise in the observation or in the hidden state. Therefore, we wait for the next observation $y_{t+1}$ and calculate $p(M_t^{(i)} = 1|Y_{t+1})$. Specifically, $p(M_t^{(MLS)} = 1|Y_{t+1})$ is the change-point score. Figure 1 shows an example of applying the method to a real time series. The top graph shows the observed rule firing counts after a transformation (see Section III). The remaining three graphs show the posterior probabilities of the three models as indicated by the labels. Notice that there is a one-time-unit delay in the probability outputs as described above. Briefly most of the time, the time-series behavior is stable and explained by the normal model (MS). This is captured by high posterior probabilities recorded for MS. At the time, where the observations deviate from the normal model, the

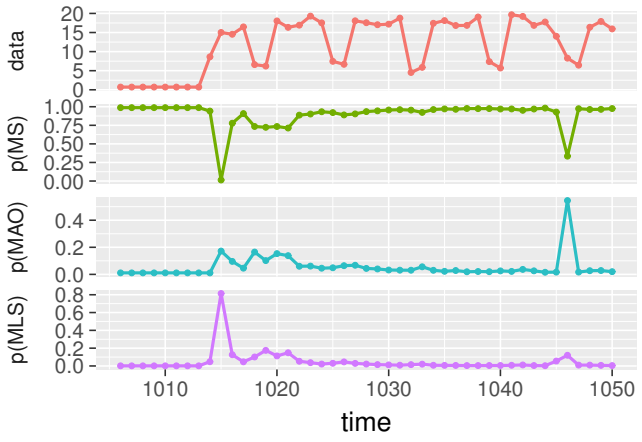


**Fig. 1:** Applying the MPDLM method to a time series. The top graph shows the observations. The remaining graphs show the posterior probabilities of the three models (MS, MAO, MLS). There is a one-time-unit delay for the probability outputs.

posterior probabilities of the other models (MAO or MLS) go up. The posterior probabilities can be used to infer which model is most likely to explain the observed deviation from the normal model. MLS covers the change-point behavior we are interested to detect.

### C. Parameter Setting

The prior distribution of the models is a multinomial distribution. Unless we have information about the frequency of different types of behaviors, we can just set a noninformative prior, i.e. an equal probability for each model.

In each DLM, we only need to set $V$ and $W$. However, it is tricky to estimate them from the data, because 1) we do not have labels for the data, and since the data could be a mix of normal and abnormal data, the estimates could be biased; 2) even if we have labels, we may not have enough data to estimate the variance for MAO and MLS. Therefore, we derive some heuristics to set the parameters.

We use three tuning parameters to control the ratios of the variance parameters: $\kappa > 1, \delta > 1, \gamma \in [0, 1]$. Let $\hat{V}$ be an estimate of the variance for normal data (which we found is not that important compared with the ratios). For MS, we set $V = \hat{V}$ and $W = 0$. For MAO, we set $V = \kappa\hat{V}$ and $W = 0$. For MLS, we set $V = \hat{V}$, $w^{(u)} = \gamma\delta\hat{V}$, $w^{(l)} = 0$, and $w^{(i)} = (1-\gamma)\delta\hat{V}, \forall i$, where $w^{(u)}$, $w^{(l)}$ and $w^{(i)}$ indicate the components on the diagonal of $W$ corresponding to $u$, $l$, and $s^{(i)}$ in $x$ (see Equation (2)). Intuitively, $\kappa$ represents the ratio of the size of a spike to a normal point. Without prior information, setting $\delta = \kappa - 1$ is recommended, because MAO and MLS would have the same variance for $y$. $\gamma$ further breaks down the variance in MLS into variance in the baseline and the seasonal levels.

## III. Experiments

We use real data of rule firing counts from a large teaching hospital collected over a period of approximately five years [1]. There are 14 rules containing at least one known change-point with a total of 22 change-points, which are used in the *real data* experiment. To further evaluate the relation between the performance of the method and the properties of the change, we create *simulated data* by first picking 4 rules that do not contain any change-point, then randomly sampling 10 segments with length 240 per rule, and finally simulating a change in the middle. We simulate the change at time $c$ in time series $y$ by changing the values as $y_i = \lambda y_i, i \geq c$. In different experiments, we set $\lambda$ to 2/1, 3/2, 6/5, 1/2, 2/3, and 5/6 respectively, to cover both increasing and decreasing changes in different sizes. The final values $y_i$ are rounded, so they are still nonnegative integers. We use multiplicative instead of additive changes, because the data are counts and have heteroscedasticity (variance changes for different means).

We compare our method (denoted as DLM) with the following change-point detection methods:

- RND: a baseline that gives uniformly sampled scores.
- SCP: a method detecting change-points for data with Gaussian distributions [4], [8].
- MW: a method based on Mann-Whitney nonparametric statistics [3].
- Pois: a method based on the likelihood ratio test [9] assuming the data follow Poisson distributions.

A sliding window of size 14 is used for all these methods to take care of the nonstationarity of the data and control the computational cost in real-time detection. Since the data are counts and have heteroscedasticity, we use a square-root transformation ($\sqrt{y + 0.5}$) to stabilize the variance except for MW and Pois [10]. We set the prior distribution of $x_0$ to $N(0, 10^6 I)$, where $I$ is the identity matrix, $\kappa = 100, \delta = \kappa - 1, \gamma = 0.99$, and $\hat{V} = 1$ for our model. We also tried to estimate $\hat{V}$, but the performance was slightly worse in these experiments. The reason, we think, is that the *ratios* between the variances in different models matters more than the *values* of the variances. Meanwhile, accurately estimating the variances from data of so many variations is quite challenging.

We use AMOC curves [11] to evaluate the performance. Usually, a delay in time is allowed in the change-point detection. We set the maximum acceptable delay to 13, according to the sliding window size 14. On the other hand, some normal data points might be falsely classified as change-points, i.e. false positives, and the false positive rate (FPR) is the proportion of the false positives out of the total of the negatives. When varying the threshold for the change-point score, we get a series of delay-vs-FPR pairs. When we plot them in a smooth curve, the result is an AMOC curve. If a change is not detected at all, a penalty is used as the delay, which is set to 14 in our experiments. The scores for the first 140 points for each time series are ignored, since it is not possible to give (accurate) scores at the beginning.

Figure 2 shows the average AMOC curves over all the change-points on real data comparing all the methods. Notice that our method dominates all the other methods almost everywhere. With an FPR of 0.01, our method achieves a mean delay of 3.32. With an FPR of 0.05, the mean delay is

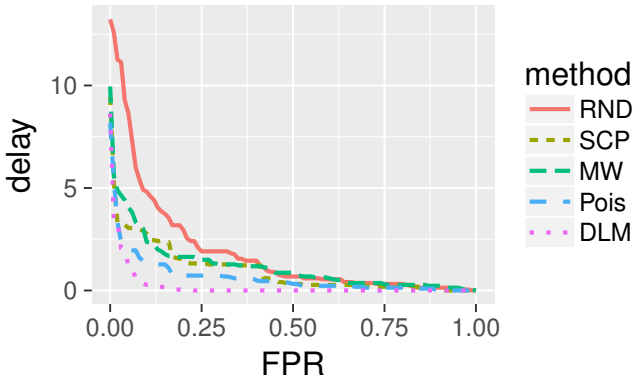


**Fig. 2:** AMOC curves on real data.

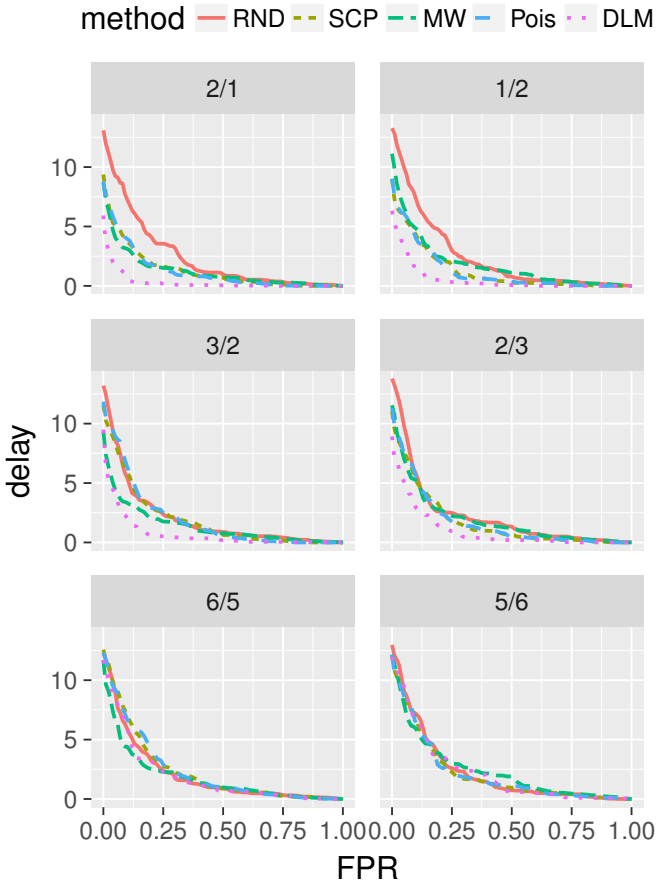


**Fig. 3:** AMOC curves on simulated data. The label on top of each subgraph indicates the fold of the changes.

1.18. The means of the areas under the AMOC curves (AUC-AMOC) are in the first row of Table I. The AUCs summarize the AMOC curves by averaging over the FPR. Wilcoxon tests show that our method is significantly better than the other methods ($^{**}p \leq 0.01$).

Figure 3 shows the average AMOC curves on simulated data. They are grouped by the experiment settings, that is the fold of the simulated changes (the value of $\lambda$), which is in the label on top of each subgraph. A general trend is that as the change gets smaller, all the curves get closer to the random baseline (RND). This reflects that the smaller the change, the harder to detect it (in time). However, except when the change is at the smallest setting, our method dominates all the other methods almost everywhere by a noticeable margin. Table I (row 2 to 7) shows the mean AUC-AMOC for different folds of changes (first column). Wilcoxon tests show that our method performs better than all the other methods significantly ($^{**}p \leq 0.01$, $^{***}p \leq 0.001$) in all cases except $\lambda = 6/5, 5/6$. Even in those cases, it is close to the best performers, and the difference is not significant ($p > 0.1$), while overall the performance of all methods is close to random.

**TABLE I:** The Mean AUC-AMOC on Real and Simulated Data.

| data | RND | SCP | MW | Pois | DLM |
|---|---|---|---|---|---|
| real | 1.88 | 0.98 | 1.16 | 0.62 | **0.19** ** |
| 2/1 | 2.37 | 1.26 | 1.21 | 1.19 | **0.28** *** |
| 3/2 | 1.97 | 1.86 | 1.36 | 1.88 | **0.68** ** |
| 6/5 | 2.01 | 2.24 | **1.74** | 2.26 | 1.88 |
| 1/2 | 2.36 | 1.22 | 1.74 | 1.19 | **0.50** *** |
| 2/3 | 2.16 | 1.67 | 1.86 | 1.66 | **0.94** ** |
| 5/6 | 2.19 | 2.06 | 2.33 | **2.05** | 2.17 |

*Acknowledgment:* This research was supported by grants R01-LM011966 and R01-GM088224 from the NIH. The content of this paper is solely the responsibility of the authors and does not necessarily represent the official views of the NIH.

## REFERENCES

[1] A. Wright, T.-T. T. Hickman, D. McEvoy, S. Aaron, A. Ai, J. M. Andersen, S. Hussain, R. Ramoni, J. Fiskio, D. F. Sittig, and D. W. Bates, "Analysis of clinical decision support system malfunctions: a case series and survey," *Journal of the American Medical Informatics Association: JAMIA*, Mar. 2016.

[2] A. Sen and M. S. Srivastava, "On Tests for Detecting Change in Mean," *The Annals of Statistics*, vol. 3, no. 1, pp. 98–108, 1975.

[3] A. N. Pettitt, "A Non-Parametric Approach to the Change-Point Problem," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, no. 2, pp. 126–135, 1979.

[4] J. Chen and A. K. Gupta, *Parametric Statistical Change Point Analysis*. Boston: Birkhäuser Boston, 2012.

[5] S. Liu, A. Wright, and M. Hauskrecht, "Change-Point Detection Method for Clinical Decision Support System Rule Monitoring," *16th Conference on Artificial Intelligence in Medicine*, vol. 10259, pp. 126–135, 2017.

[6] P. J. Harrison and C. F. Stevens, "Bayesian Forecasting," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 38, no. 3, pp. 205–247, 1976.

[7] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASME-Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.

[8] R. Killick and I. Eckley, "changepoint: An R Package for changepoint analysis," *Lancaster University*, pp. 1–15, 2013.

[9] G. Casella and R. Berger, *Statistical Inference*, ser. Duxbury advanced series in statistics and decision sciences. Thomson Learning, 2002.

[10] M. S. Bartlett, "The Use of Transformations," *Biometrics*, vol. 3, no. 1, pp. 39–52, 1947.

[11] T. Fawcett and F. Provost, "Activity monitoring: Noticing interesting changes in behavior," in *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, vol. 1, 1999, pp. 53–62.