Leveraging Cloud Data to Mitigate User Experience from 'Breaking Bad'

Nicholas A. James† Arun Kejariwal‡ David S. Matteson†
†Cornell University ‡Twitter Inc.

ABSTRACT

Low latency and high availability of an app or a web service are key, amongst other factors, to the overall user experience (which in turn directly impacts the bottomline). Exogenic and/or endogenic factors often give rise to breakouts in cloud data which makes maintaining high availability and delivering high performance very challenging. Although there exists a large body of prior research in breakout detection, existing techniques are not suitable for detecting breakouts in cloud data owing to being not robust in the presence of anomalies.

To this end, we developed a novel statistical technique to automatically detect breakouts in cloud data. In particular, the technique employs *Energy Statistics* to detect breakouts in both application as well as system metrics. Further, the technique uses robust statistical metrics, viz., median, and estimates the statistical significance of a breakout through a permutation test. To the best of our knowledge, this is the first work which addresses breakout detection in the presence of anomalies.

We demonstrate the efficacy of the proposed technique using production data and report Precision, Recall and F-measure measure. The proposed technique is 3.5× faster than a state-of-the-art technique for breakout detection and is being currently used on a daily basis at Twitter.

1. INTRODUCTION

In a recent report, Mary Meeker from KPCB mentioned that mobile usage continues to rise reapidly (14% Y/Y) and mobile usage now accounts for 25% of the total web usage [1]. In a similar vein, Strategy Analytics reported that mobile data traffic is expected to to rise by 300% by 2017 to a peak of 21 Exabytes, from 5 Exabytes in 2012 [2]. Growing traffic and user engagement directly impacts the performance and availability of an app/website. To this end, KISSmetrics reported the following [3]:

- 73% of mobile internet users say that they have encountered a website that was too slow to load.
- 38% of mobile internet users say that they have encountered a website that was not available.
- A 1 second delay in page response can result in a 7% reduction in conversions.

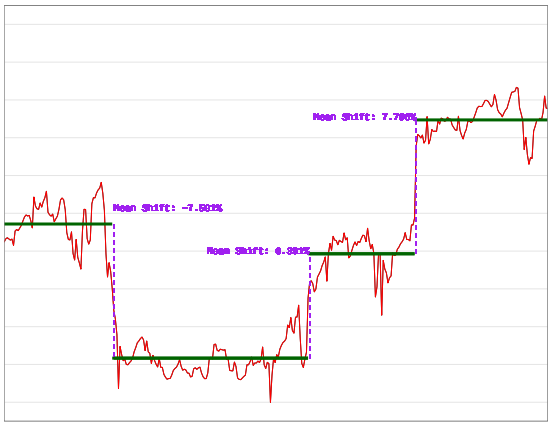


Figure 1: Example *breakouts* observed in production data at Twitter

Likewise, in [4], it was reported that performance has a direct impact on business KPIs (Key Performance Indicators). In [5], Shunra (now acquired by Hewlett-Packard) reported: *If your mobile app fails, 48% of users are less likely to ever use the app again. 34% of users will just switch to a competitors app, and 31% of users will tell friends about their poor experience, which eliminates those friends as potential customers.*

Amongst a large multitude of factors, *breakouts* – characterized by either a mean shift or a rampup from one steady state to another in a given time series (exemplified in Figure 1) – in system and/or application metrics can potentially impact performance and availability, thereby adversely impacting the end user experience. A wide variety of factors, some are enumerated below, can induce breakouts[1] in system and/or application metrics.

(a) Continuous code deployment
(b) A/B testing [6, 7, 8]
(c) Launch of new products or new product features
(d) Partial failure of a cluster: Höelzle and Barroso point out that hardware failure in the cloud is more of a norm than exception [9] (also see [10, 11]).

Breakouts can potentially impact latency and availability experienced by the end user. In light of this, it is critical to detect breakouts early (robust breakout detection would also facilitate assessing the efficacy of an A/B test). Although there exists a large body of prior research in breakout detection, existing techniques are not suitable for detecting

[1] *Breakout*, a term commonly used in finance, is referred to as *changepoint* in statistics.

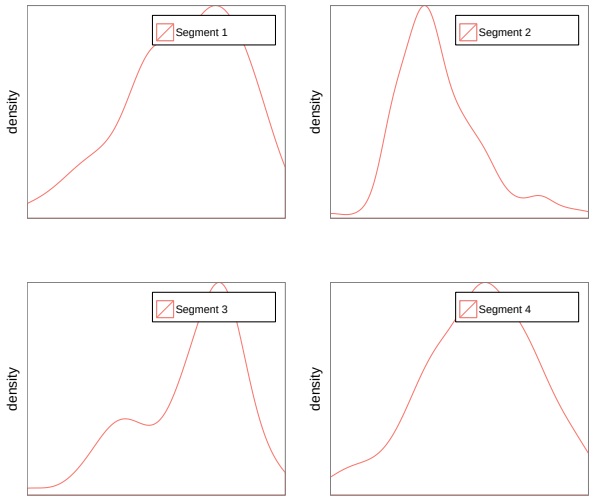


Figure 2: Data distribution of the four segments shown in Figure 1

breakouts in cloud data owing to not being robust in the presence of anomalies. To this end, we developed a novel technique to automatically detect breakouts in cloud data (which comprises of millions of time series at Twitter [12]). The main contributions of the paper are as follows:

- First, we propose a novel statistical technique, called **E-Divisive with Medians (EDM)**, to automatically detect breakouts in cloud data. Unlike the existing techniques for breakout detection, **EDM** is robust against the presence of anomalies.[2] The salient features of **EDM** are the following:
  - **EDM** employs E-statistics [14] to detect divergence in mean. Note that, in general, **EDM** can also be used detect change in distribution in a given time series (discussed further in Section 3).
  - **EDM** uses robust statistical metrics, viz., median [15, 16] , and estimates the statistical significance of a breakout through a permutation test.
  - **EDM** is non-parametric. This is of paramount importance as the cloud data does not follow the commonly assumed normal distribution, as illustrated by Figure 2 or any other widely accepted model. From the figure we note that none of the four segments in Figure 1 follow a common distribution.

  To the best of our knowledge, this is the first work which addresses breakout detection in the presence of anomalies.
- Second, we present a detailed evaluation of **EDM** using production data.
  - Using production data, we demonstrate that techniques such as `PELT` [17] do not fare well when applied to cloud data owing to a non-normal distribution of cloud data.
  - We also report Precision, Recall and F-measure to assess the efficacy of **EDM**.

  The proposed technique is 3.5× faster than a state-of-the-art technique for breakout detection and is being currently used on a daily basis at Twitter.

[2]Note that the presence of anomalies in production cloud is not uncommon [13].

The remainder of the paper is organized as follows: Section 2 presents a brief background. Section 3 details the proposed technique for detecting breakouts in cloud data with anomalies. Section 4 presents an evaluation of the proposed technique. Lastly, conclusions and future work are presented in Section 6.

2. BACKGROUND

In this section we present a brief background of the concepts used by **EDM** for detecting breakouts.

2.1 Divergence Measure

To detect breakouts, we employ a metric based on the weighted L^2-distance between the characteristic functions of random variables. Let X and Y be independent random variables, X' be an i.i.d copy of X and Y' be an i.i.d. copy of Y. Let the cumulative distribution function of X and Y be denoted by F and G respectively.

DEFINITION **1.** *The energy distance between X and Y is defined as follows [14]:*

$$\mathcal{E}(X,Y) = 2E|X-Y| - E|X-X'| - E|Y-Y'| \quad (1)$$

In [18], Rizzo and Székely showed that the L^2-distance between F and G satisfies the following:

$$2\int_{-\infty}^{\infty} (F(x)-G(x))^2 dx = \mathcal{E}(X,Y) \quad (2)$$

For a random variable X, its characteristic function $\phi_x(t)$ is defined by $\phi_x(t) = E(\exp\{iXt\})$. Using this notation, Székely and Rizzo [14] show that the energy distance between X and Y can also be represented in terms of their characteristic functions:

$$\mathcal{E}(X,Y) = \int_{-\infty}^{\infty} \frac{|\phi_x(t)-\phi_y(t)|^2}{\pi t^2} dt. \quad (3)$$

Since the characteristic function, like the cumulative distribution function, uniquely defines a random variable, we define a class of distance measures based on them. Let

$$\mathcal{D}(X,Y;\alpha) = \int_{-\infty}^{\infty} |\phi_x(t)-\phi_y(t)|^2 \omega(t;\alpha) dt \quad (4)$$

where $\omega(t;\alpha)$ is a weight function, parameterized by α, such that $D(X,Y;\alpha) < \infty$. The indexing parameter α is used to scale the distance between distributions. For instance, the metric used in Equation 3 is obtained by using $\omega(t;\alpha) = \frac{1}{\pi t^2}$. In [19], Székely and Rizzo suggested the following for ω:

$$\omega(t;\alpha) = \left(\frac{2\pi^{\frac{1}{2}}\Gamma(1-\alpha/2)}{\alpha 2^{\alpha}\Gamma[(1+\alpha)/2]} |t|^{\alpha} \right)^{-1}$$

where $\Gamma(\cdot)$ is the complete gamma function. Using this weight function allows us to obtain a metric that generalizes

the one in Equation 1. For $\alpha \in (0, 2]$, the generalized energy distance between X and Y is given by:

$$\mathcal{E}(X, Y; \alpha) = 2E|X - Y|^\alpha - E|X - X'|^\alpha - E|Y - Y'|^\alpha$$

Székely and Rizzo [18] also show that with this weight function, and $\alpha \in (0, 2]$, we have

$$\mathcal{D}(X, Y; \alpha) = \mathcal{E}(X, Y; \alpha).$$

For detecting divergence in mean, α is set to 2; on the other hand, for detecting arbitrary change in distribution, $0 < \alpha < 2$ may be a better choice [19]. This property is exemplified through the following Lemma.

LEMMA **1.** *For any pair of independent random variables X and Y, and for any $\alpha \in (0, 2)$, if $E(|X|^\alpha + |Y|^\alpha) < \infty$, then $\mathcal{E}(X, Y; \alpha) \in [0, \infty)$, and $\mathcal{E}(X, Y; \alpha) = 0$ if and only if X and Y are identically distributed. Furthermore, if $\alpha = 2$, we have that $\mathcal{E}(X, Y; 2) = 0$ if and only if $EX = EY$.*

PROOF. A proof is given in [19]. □

The metric $\mathcal{E}$ allows for a simple and intuitive approximation to $\mathcal{D}$ and doesn't require any integration. Let $\boldsymbol{X}_n = \{X_i : i = 1, \ldots, n\}$ and $\boldsymbol{Y}_m = \{Y_j : j = 1, \ldots, m\}$ be independent iid samples from the distribution of $X, Y \in \mathbb{R}^d$, respectively, such that $E|X|^\alpha, E|Y|^\alpha < \infty$ for some $\alpha \in (0, 2)$. We can then approximate $\mathcal{E}$ by $\widehat{\mathcal{E}}$ as follows:

$$\begin{aligned}\widehat{\mathcal{E}}(\boldsymbol{X}_n, \boldsymbol{Y}_m; \alpha) &= \frac{2}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}|x_i - y_j|^\alpha \\ &\quad - \binom{n}{2}^{-1}\sum_{i<j}|x_i - x_j|^\alpha \\ &\quad - \binom{m}{2}^{-1}\sum_{i<j}|y_i - y_j|^\alpha \end{aligned} \tag{5}$$

The first term on the right hand side of Equation 6 correspond to the *between* distance between $\boldsymbol{X}_n$ and $\boldsymbol{Y}_m$. The second and third terms on the right side of Equation 6 correspond to the *within* distance of $\boldsymbol{X}_n$ and $\boldsymbol{Y}_m$ respectively [19].

By the strong law of large numbers for U-statistics [20], $\widehat{\mathcal{E}} \to \mathcal{E}$ as $\min(n, m) \to \infty$. Furthermore, Székely and Rizzo [18] show that under the null hypothesis of equal distributions, i.e., $\mathcal{E}(X, Y; \alpha) = 0$,

$$\frac{nm}{n+m}\widehat{\mathcal{E}}(\boldsymbol{X}_n, \boldsymbol{Y}_m; \alpha) \Rightarrow \mathcal{A}$$

as $\min(n, m) \to \infty$, where $\mathcal{A}$ is a non-degenerate random variable and $M \Rightarrow N$ means that M converges in distribution to N. However, under the alternative hypothesis, $\frac{nm}{n+m}\widehat{\mathcal{E}} \to \infty$ as $\min(m, n) \to \infty$. For notational simplicity, we will use the following in the remainder of the paper:

$$\widehat{\mathcal{Q}}(\boldsymbol{X}_n, \boldsymbol{Y}_m; \alpha) = \frac{nm}{n+m}\widehat{\mathcal{E}}(\boldsymbol{X}_n, \boldsymbol{Y}_m; \alpha) \tag{6}$$

2.2 Permutation Test

The convergence of the statistic presented in Equation 6 allows us to determine the statistical significance of a proposed breakout. Let the observations of a time series be given by $Z_1, Z_2, \ldots, Z_n$ and $1 \le \tau < \kappa \le n$ be constants. We define the following sets $A_\tau = \{Z_1, Z_2, \ldots, Z_\tau\}$ and $B_\tau(\kappa) = \{Z_{\tau+1}, \ldots, Z_\kappa\}$. A breakout location $\hat{\tau}$ is then estimated as the value that maximizes

$$\widehat{\mathcal{Q}}(A_\tau, B_\tau(\kappa); \alpha)$$

for $1 \le \tau < \kappa \le n$. Along with the estimated breakout location we also have an associated test statistic

$$\hat{q} = \widehat{\mathcal{Q}}(A_{\hat{\tau}}, B_{\hat{\tau}}(\hat{\kappa}); \alpha).$$

Given $\alpha = 2$, large values of $\hat{q}$ correspond to a significant change in mean (and a distribution in general). However, calculating a precise critical value requires a knowledge of the underlying distributions, which are generally unknown. Therefore, we propose a permutation test to determine the significance of $\hat{q}$.

Under the null hypothesis that there does not exist a breakout, we conduct a permutation test as follows. First, the observations are permuted to construct a new time series. Then, we re-apply the estimation procedure to the permuted observations. This process is repeated and after the rth permutation of the observations we record the value of the test statistic $\hat{q}^{(r)}$.

This permutation test will result in an exact p-value if we consider all possible permutations. However, this is not computationally tractable in general. Therefore, we obtain an approximate p-value by performing a sequence of R *random* permutations. The approximate p-value is computer as follows:

$$\#\{r : \hat{q}^{(r)} \ge \hat{q}\}/(R + 1)$$

The re-sampling risk, the probability of a different decision than the one based on the theoretical p-value, can be uniformly bounded by an arbitrarily small constant using the approach proposed by Gandy [21]. In our analysis we test at the 5% significance level and use $R = 199$ permutations.

2.3 Metrics

In order to minimize user impact, it is imperative to detect breakout(s) at the earliest. We qualify the timeliness of breakout detection via **EDM** using the metric **TTD** defined below:

DEFINITION **2.** *We define* **TTD (Time to Detect)** *as the number of time series observations between the occurrence of a breakout and the breakout estimate reported by a breakout detection algorithm.*

Precision is the ratio of true positives (tp) over the sum of true positives (tp) and false positives (fp). *Recall* is the ratio of true positives (tp) over the sum of true positives (tp) and false negatives (fn). *F-measure* is defined as follows (refer to [22] for a detailed discussion):

$$F = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{7}$$

3. E-DIVISIVE WITH MEDIANS

Suppose that we are given the following time series, $Z_1, Z_2, \ldots, Z_n$ consisting of independent observations. A breakout is characterized by a value $\gamma \in (0, 1)$ such that observations

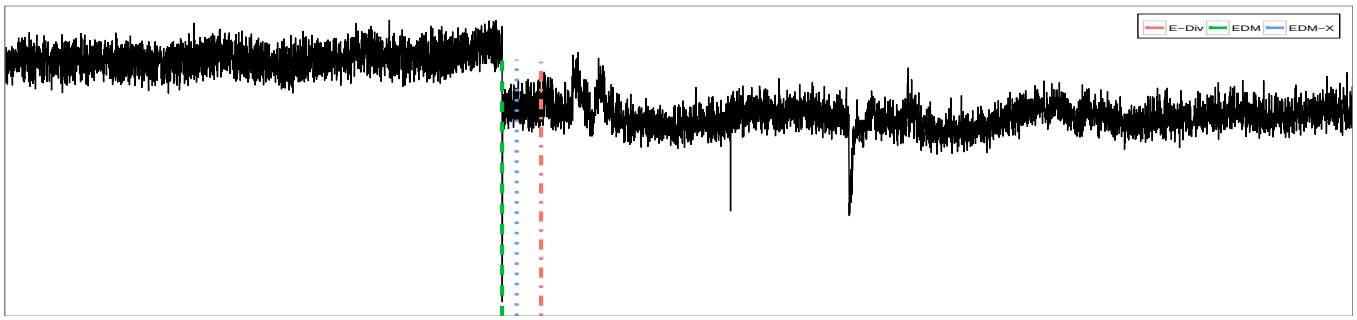


Figure 3: Example (using production data) highlighting the impact of presence of anomalies on breakout detection

$\{Z_1, Z_2, \ldots, Z_{\lfloor \gamma N \rfloor}\}$ have distribution function F, and observations $\{Z_{\lfloor \gamma N \rfloor + 1}, Z_{\lfloor \gamma N \rfloor + 2}, \ldots, Z_n\}$ have distribution function G. Furthermore, it is assumed that $F \neq G$. In order to determine if the observations in the provided time series are identically distributed we perform the following hypothesis test:

$$\mathrm{H}_0 : \gamma = 1$$

$$\mathrm{H}_A : 0 < \gamma < 1$$

If the null hypothesis of no breakout is rejected, we must then also return an estimate for the breakout location. Prior work in breakout detection assumes that the time series under consideration is free of anomalies. However, this is not the case in production cloud data. Figure 3 illustrates the impact of the presence of anomalies on the location of a breakout detected. From the figure we note that there are multiple global anomalies, both positive and negative. The breakout locations obtained using `E-Div` (of the `ecp` R package [23]) and the algorithms – **EDM** and **EDM-X** – presented later in this section are marked with vertical lines. From a TTD perspective, we note that using the proposed algorithms we obtain estimates of the location of breakouts than other non-parametric procedures. This is due to the fact that **EDM** and **EDM-X** are anomaly "aware".

3.1 Robustness against anomalies

The approximation, $\widehat{\mathcal{E}}$ given in Equation 6 is susceptible to anomalies since one single anomaly can greatly change its value. This is due to the fact that $\widehat{\mathcal{E}}$ is based upon a linear combination of sample means. To alleviate this issue we instead use a robust location estimator, the median. We thus define the robust between sample distance:

$$m^{\alpha}_{XY} = \text{median}\left\{|x_i - y_j|^{\alpha} : 1 \leq i \leq n, 1 \leq j \leq m\right\}$$

and similarly define m_{XX} and m_{YY} as the median of the within sample distances. We then obtain a robust version of $\widehat{\mathcal{E}}$ as follows:

$$\widetilde{\mathcal{E}}(\boldsymbol{X}_n, \boldsymbol{Y}_m; \alpha) = 2m^{\alpha}_{XY} - m^{\alpha}_{XX} - m^{\alpha}_{YY} \tag{8}$$

For any two given sets $\boldsymbol{X}_n$ and $\boldsymbol{Y}_m$ the time necessary to compute Equation 8 a single time is linearly proportional to the number of distance terms present. Therefore, if we assume that $n \geq m$, both $\widehat{\mathcal{E}}$ and $\widetilde{\mathcal{E}}$ require $O(n^2)$ calculations to evaluate.

However, if a single observation is added/removed from either $\boldsymbol{X}_n$ or $\boldsymbol{Y}_m$, the value of $\widehat{\mathcal{E}}$ can be updated in $O(n)$ time, but $\widetilde{\mathcal{E}}$ will require $O(n^2)$. However, if we use a tree data structure, we can update $\widetilde{\mathcal{E}}$ in $O(n \log(n))$ time; but, this comes at the expense of needing $O(n^2 \log(n))$ time to calculate the initial value of our statistic. Since such updates may be done a large number of times, we consider this trade-off to be acceptable.

Although we can now quickly perform updates we will have to keep track of all $O(n^2)$ distances. Even for moderately sized time series this may become intractable, even with 24GB of memory. For this reason we make use of interval trees (see the Appendix for further details) in order to obtain an approximate median. Through experimentation we learned that even the $O(n \log(n))$ update is too slow, and thus we use the following approximation.

Let $\delta > 1$. We approximate the within distance for the set $\boldsymbol{X}_n$ as follows:

$$m^{\alpha,\delta}_{XX} = \text{median}\left\{|x_i - x_j|^{\alpha} : 1 \leq i < j \leq \delta \text{ or } i + 1 = j\right\}$$

We similarly define $m^{\alpha,\delta}_{YY}$. The between distance is approximated by using only δ observations from each set. Figure 4 shows two possible ways of selecting the δ observations.

Head Figure 4 (A) chooses to take the δ observations that are at head of both sets $\mathbf{X}$ and $\mathbf{Y}$.

Tail Figure 4 (B) chooses to take the δ observations at the tail of set $\mathbf{X}$ and the head of $\mathbf{Y}$.

Based on our experiments using production data we learned that using the **Tail** (as illustrated in Figure 4 (B)) yields better breakout estimates and hence, we use:

$$m^{\alpha,\delta}_{XY} = \text{median}\left\{|x_i - y_j|^{\alpha} : n - \delta + 1 \leq i \leq n, 1 \leq j \leq \delta\right\}$$

In light of the aforementioned approximation, Equation 6 can be written as:

$$\widetilde{\mathcal{Q}}(\boldsymbol{X}_n, \boldsymbol{Y}_m; \alpha, \delta) = \frac{nm}{n+m}\widetilde{\mathcal{E}}(\boldsymbol{X}_n, \boldsymbol{Y}_m; \alpha, \delta) \tag{9}$$

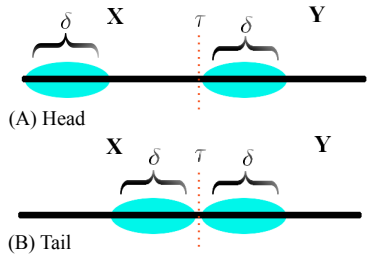


Figure 4: This figure depicts two different ways of selecting which δ observations to use for approximating the between distance.

Typically δ is chosen such that it is much smaller than $\sqrt{n}$. Therefore, with these approximations we can create the statistic $\widetilde{\mathcal{E}}(\boldsymbol{X}_n, \boldsymbol{Y}_n; \alpha, \delta)$, which can be calculated in $O(n\log(n))$ time and updated in $O(\log(n))$ time when using the interval tree approximation.

3.2 Algorithm

The **EDM** algorithm makes use of the $\widetilde{\mathcal{E}}(\cdot,\cdot;\alpha,\delta)$ statistic presented in the previous section. Let a time series be given by $Z_1, Z_2, \ldots, Z_n$ and $1 < \delta \leq \tau$ and $\tau + \delta \leq \kappa \leq n$. We define the following sets: $A_\tau = \{Z_1, Z_2, \ldots, Z_\tau\}$ and $B_\tau(\kappa) = \{Z_{\tau+1}, Z_{\tau+2}, \ldots, Z_\kappa\}$. Thus, both A_τ and $B_\tau(\kappa)$ have at least δ observations. Using Equation 9 we obtain the breakout estimate, $\hat{\tau}$, as follows:

$$(\hat{\tau}, \hat{\kappa}) = \underset{\tau,\kappa}{\operatorname{argmax}} \ \widetilde{\mathcal{Q}}(A_\tau, B_\tau(\kappa); \alpha, \delta) \tag{10}$$

By solving the maximization problem given in Equation 10 we not only obtain an estimate $\hat{\tau}$, but also its associated test statistic value $\hat{q}$. Given this and a predetermined significance level, we perform a permutation test (detailed in subsection 2.2) to determine whether the reported breakout is statistically significant.

Algorithm 1 is used to determine $\hat{\tau}$ and $\hat{\kappa}$. We set $D = 10$ in our implementation. However, we suggest selecting D such that $2^D \approx n$. Then, the algorithm makes use of two key procedures, `ForwardUpdate` and `BackwardUpdate`.

Parameters: Z, δ, and D
Let T_A, T_B, and T_{AB} be interval trees with 2^D leaf nodes
`// Initialize within distance trees`
for $1 \leq i \leq \delta$ **do**
  for $i+1 \leq j \leq \delta$ **do**
    Insert $|Z_i - Z_j|$ to T_A
    Insert $|Z_{i+\delta} - Z_{j+\delta}|$ to T_B
  end
end
`// Initialize between distance tree`
for $1 \leq i \leq \delta$ **do**
  for $1 \leq j \leq \delta$ **do**
    Insert $|Z_i - Z_{j+\delta}|$ to T_{AB}
  end
end
$\langle m1, m2, m3\rangle$ = approx. median $\langle T_{AB}, T_A, T_B\rangle$
$bestStat = \frac{\tau(\kappa-\tau)}{\kappa}(2m1 - m2 - m3)$
$bestLoc = \delta$
$\tau = \delta$
$forwardMove = 0$
`// Update trees`
while $\tau \leq n - \delta$ **do**
  if *forwardMove = 1* **then**
    Perform ForwardUpdate
  end
  else
    Perform BackwardUpdate
  end
  forwardMove = 1 - *forwardMove*
end
return *bestLoc*

Algorithm 1: EDM

These procedures allow us to efficiently update $\widetilde{\mathcal{Q}}$ by making use of the current states of the interval trees.

- ForwardUpdate iterates κ from $\tau + \delta + 1$ to n and updates the value of $\widetilde{\mathcal{Q}}$ after each iteration. Each iteration corresponds to adding values to $B_\tau(\kappa)$.
- BackwardUpdate iterates κ from $n - 1$ to $\tau + \delta + 1$ and updates the value of $\widetilde{\mathcal{Q}}$ after each iteration. Each iteration corresponds to removing values from $B_\tau(\kappa)$.
- For both procedures ForwardUpdate and BackwardUpdate, all the parameters are passed by reference. Additionally, both procedures obtain the an approximate medians in $\mathcal{O}(D)$ (refer to the Appendix for details). In both cases, all the interval trees are updated. Hence, the statistic value can be computed in logarithmic time.

3.2.1 Special Case: $\alpha = 2$

It should be noted that when $\alpha = 2$, it is possible to obtain a much more efficient algorithm. In this case, $\mathcal{E}(X, Y; 2) = 2(EX - EY)^2$; hence, changes in mean can be detected. As mentioned before, a robust location can be estimate by considering the sample median instead of the sample mean. In this case, we define $\widetilde{\mathcal{E}}$ as follows:

$$\widetilde{\mathcal{E}}(A_\tau, B_\tau(\kappa); 2, \delta) = 2[\text{median}(A_\tau) - \text{median}(B_\tau(\kappa))]^2$$

Parameters: Z, δ, T_A, T_B, T_{AB}, τ, *bestStas*, *bestLoc*
$n = Z.size()$
$\tau \leftarrow \tau + 1$
Update counts in T_A, T_B, and T_{AB} resulting from new τ value
for $\tau + \delta \leq \kappa \leq n$ **do**
  Insert $-Z_\kappa - Z_{\kappa-1}|$ to tree T_B
  $\langle m1, m2, m3\rangle$ = approx. median $\langle T_{AB}, T_A, T_B\rangle$
  $stat = \frac{\tau(\kappa-\tau)}{\kappa}(2m1 - m2 - m3)$
  if $stat > bestStat$ **then**
    $bestStat = stat$
    $bestLoc = \tau$
  end
end

Procedure ForwardUpdate

Parameters: Z, δ, T_A, T_B, T_{AB}, τ, *bestStat*, *bestLoc*
$n = Z.size()$
$\tau \leftarrow \tau + 1$
Update counts in T_A, T_B and T_{AB} resulting from new τ value
$\kappa = n$
while $\kappa \geq \tau + \delta$ **do**
  Insert $|Z_\kappa - Z_{\kappa-1}|$ to tree T_B
  $\langle m1, m2, m3\rangle$ = approx. median of $\langle T_{AB}, T_A, T_B\rangle$
  stat = $\frac{\tau(\kappa-\tau)}{\kappa}(2m1 - m2 - m3)$
  if $stat > bestStat$ **then**
    $bestStat = stat$
    $bestLoc = \tau$
  end
  $\kappa \leftarrow \kappa - 1$
end

Procedure BackwardUpdate

Parameters: Z and δ

max-heaps $LMax$ and $RMax$
min-heaps $LMin$ and $RMin$
$bestStat = -\infty$
$bestLoc = -1$
for $1 \leq i < \delta$ **do**
  addToHeaps($LMax$, $LMin$, Z_i)
end
for $\delta \leq i \leq n - \delta$ **do**
  addToHeaps($LMax$, $LMin$, Z_i)
  mL = getMedian($LMax$, $LMin$)
  empty $RMax$ and $RMin$
  for $i \leq j < i + \delta$ **do**
    addToHeaps($RMax$, $RMin$, Z_j)
  end
  for $i + \delta \leq j \leq n$ **do**
    addToHeaps($RMax$, $RMin$, Z_j)
    mR = getMedian($RMax$, $RMin$)
    $stat = \frac{i(j-i)}{j}(mL - mR)^2$
    if *stat ¿ bestStat* **then**
      $bestStat = stat$
      $bestLoc = i$
    end
  end
end
return *bestLoc*

Algorithm 2: EDM-X

This algorithm only considers the median of the actual observations and not the median of their distances. Unlike the case where $0 < \alpha < 2$, only $O(n)$ additional memory is required and updates can be performed in $O(\log(n))$ time. This simplification enables the use of exact medians instead of approximations. Because of this feature, we call this algorithm **E-Divisive with Exact Medians (EDM-X)** – see Algorithm 2. The algorithm is able to keep track of the exact medians by using pairs of heaps; a max-heap stores the $\frac{n}{2}$ smallest observations, while a min-heap stores the $\frac{n}{2}$ largest observations. If n is odd, then one of these heaps will have an additional element. Procedure `addToHeaps` is used by Algorithm 2 to maintain these properties for a given pair of heaps. And since the heaps are passed by reference no extra space or time is required to make copies.

4. EVALUATION

In this section we detail the evaluation methodology and present results demonstrating the efficacy, measured in terms of TTD (refer to subsection 2.3, of the algorithms presented in the previous section. Our experiments show that the presence of anomalies can significantly skew the TTD of breakout algorithm.

4.1 Methodology

The efficacy of **EDM** and **EDM-X** was evaluated using a wide corpus of time series data obtained from production. The time series corresponded to both *system* and *application* metrics. For example, but not limited to, the following metrics were used:

- System Metrics
  - CPU utilization, Heap usage, Disk writes
  - Time spent in garbage collection
- Application Metrics
  - Request rate
  - Latency

Parameters: M, m, and x
if *m is empty* **then**
  Add x to M
end
if *m isn't empty and* $x \leq m.top()$ **then**
  Add x to M
end
else
  Add x to m
end
if $M.size() > m.size + 1$ **then**
  Move $M.top()$ to m
end
if $m.size() > M.size() + 1$ **then**
  Move $m.top()$ to M
end

Procedure addToHeaps

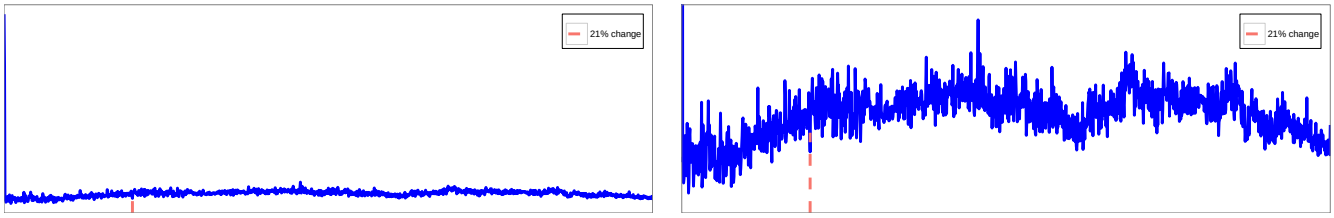
Parameters: $M$and m
Output : The current median
if $m.size() = M.size() + 1$ **then**
  return $m.top()$
end
if $M.size() = m.size() + 1$ **then**
  return $M.top()$
end
if $M.size() = m.size()$ **then**
  return $(M.top() + m.top())/2$
end

Procedure getMedian

In addition to the time series of the metrics mentioned above, we also used minutely time series of the stock price of a publicly traded company. Overall, more than 20 data sets were used for evaluation. Given the velocity, volume, and real-time nature of cloud infrastructure data, it is not practical to obtain time series data with "true" breakouts labeled. However, to determine TTD, location of a "true" breakout is needed. To this end, for the data sets (obtained from production) we used for evaluation, we determined the "true" breakouts manually and then computed the TTD.

4.2 PELT and E-Divisive

Visual analysis serves as the starting point for deriving insights from Big Data [24, 25, 26]. With the increase in volume in Big Data, there has been increasing impetus being given to extreme scale visual analytics [27]. The May 2013 edition of IEEE Computer covered the challenges in the realm of Big Data visual analytics [28, 29]. However, as mentioned earlier, due to the velocity and volume of cloud data, visual detection of breakouts is not practical. Furthermore, sometimes a breakout isn't always obvious due to the range of the observed values. This is exemplified by Figure 5. From Figure 5a we note that there is an anomaly on the left hand side due to which even a 21% change in mean is



(a) (b)

Figure 5: An example highlighting the limitations of visual detection of breakout(s)

cannot be detected via visual inspection. However, on zooming in (in other words, limiting the range of the y-axis), see Figure 5b, we observe the aforementioned breakout.

To this end, we first evaluated the `PELT` (Pruned Exact Linear Time) method by Killick and Haynes [30]. This is a parametric method that can be used to detect single as well as multiple breakout analysis. In the current context, we focus only on its properties for estimating a single breakout. This method is usually applied by using a log-likelihood function to measure fit, but as shown in [17] the underlying concepts can be extended to a number of different measure of fit. One the major benefits of this algorithm is its speed, which has been shown to have an expected linear running time.

We also evaluated the `E-Divisive` method [31]. This is a non-parametric breakout detection algorithm that is based upon the statistic presented in Equation 6. Akin to `PELT`, this method can also be used to estimate multiple breakouts, but we will once again only examine its performance at identifying a single breakout. However, unlike `PELT`, `E-Divisive` is a non-parametric algorithm and makes weak distributional assumptions. Hence, `E-Divisive` can be applied in a wider range of settings, such as those where one is not certain that `PELT`'s assumptions necessarily hold. On the other hand, `E-Divisive` has a quadratic running time, which is much slower than that of `PELT`.

4.2.1 *Data Without Anomalies*

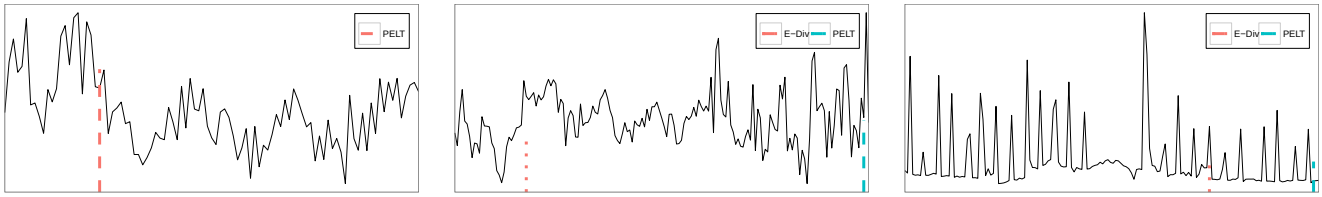
First, we applied the `PELT` procedure to the datasets mentioned earlier in this section. Figure 6a exemplifies a case wherein the `PELT` method is efficient in detecting a breakout. This is further supported by the TTD values in column 3 of Table 1. However, since `PELT` makes distributional assumptions through its use of likelihood functions, `PELT`'s performance suffer – large TTD value – when these assumptions do not hold. This is illustrated by Figure 6b and column 3 of Table 1.

To address this problem, we used `E-Divisive` to compute breakout location. Figure 6b and column 2 of Table 1 show

	Raw Data TTD		Rolling Median TTD		Anomalies Removed TTD	
	E-Div	PELT	E-Div	PELT	E-Div	PELT
Dataset 1	0	0	0	0	71	74
Dataset 2	0	0	0	1	18	42
Dataset 3	2	1	1	1	1	1
Dataset 4	0	6	0	38	0	0
Dataset 5	0	65	1	65	0	65
Dataset 6	2	5	2	5	2	5
Dataset 7	6	7	4	7	1	7
Dataset 8	3	4	2	4	3	4
Dataset 9	9	8	6	8	8	15
Dataset 10	14113	15	14114	16	14113	15
Dataset 11	0	1	0	4	-	-
Dataset 12	0	1	1	1	1	1
Dataset 13	45	2	45	2	-	-
Dataset 14	0	1	1	2	0	1
Dataset 15	1	1590	0	1590	-	-
Dataset 16	0	1	0	1	-	-
Dataset 17	2	263	1	263	1681	1733
Dataset 18	1	0	2	0	2	0
Dataset 19	2	61	1	61	105	108
Dataset 20	4479	5607	4476	5607	4479	5607
Dataset 21	27	349	41	13	-	-
Dataset 22	0	0	3	19	4	4
Dataset 23	4	1	15	15	-	-
Dataset 24	32	44	17	0	1	1
Dataset 25	0	6	18	89	5	4

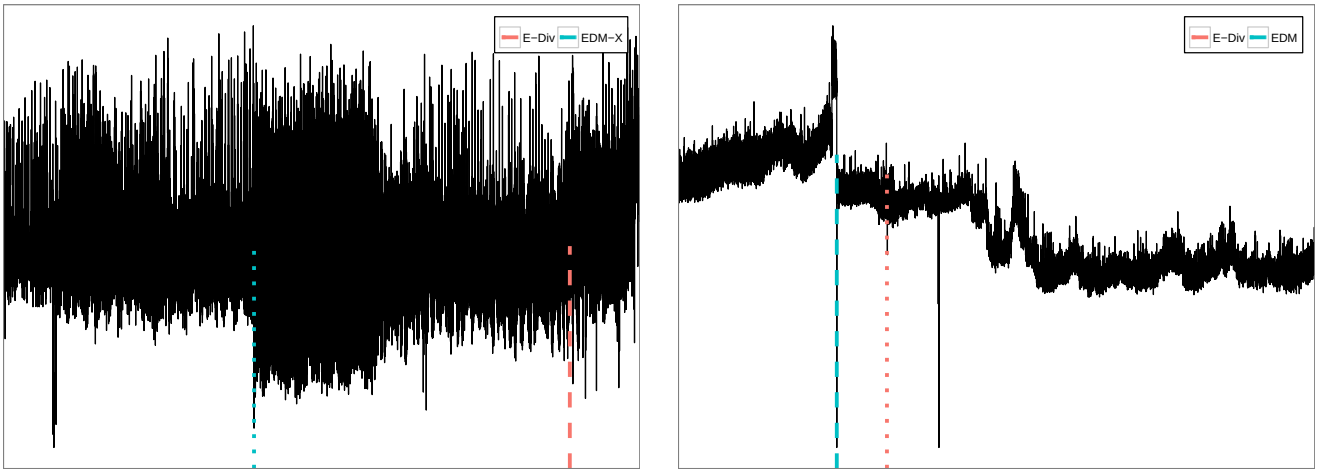
Table 1: TTD for the `E-Divisive` and `PELT` methods when applied to raw and rolling median time series

that in almost all cases `E-Divisive` results in a smaller TTD. Furthermore, since `E-Divisive` is a non-parametric method it can be applied to a wider array of settings, especially those where `PELT`'s assumptions are not guaranteed to hold. However, although `E-Divisive` is significantly slower than `PELT` we find this an acceptable trade off because of the decreased TTD and greater range of applications.



(a) (b) (c)

Figure 6: Efficacy of `PELT` and `E-Divisive`



(a) `E-Divisive` and **EDM-X** (b) `E-Divisive` and **EDM**

Figure 7: Illustration of efficacy of **EDM-X** and **EDM**

4.2.2 *Data with anomalies*

In the previous section we showed that when a dataset doesn't contain any anomalies that both `PELT` and `E-Divisive` can be used to compute robust estimates locations of a breakout. However, this is not the case in the presence of anomalies[3], as illustrated by Figure 6c. A common approach to mitigate the effect of anomalies is local smoothing. The smoothers we considered were the rolling mean and rolling median. For these smoothers, each observation is replaced by either the mean or median of its neighboring values. As anomalies can still effect the smoothed values when calculating the rolling mean, we used the rolling median. Although these methods can reduce the impact of anomalies, it can result in an increased TTD as seen from columns 4 and 5 of Table 1. Another drawback to this approach is that one must choose the size of the neighborhood to use to calculate the smoothed values. A neighborhood that is too small will limit the mitigation of the effect of an anomaly; on the other hand, a neighborhood one that is too big can potentially smooth the mean changes (a breakout) in a time series.

Another approach is to remove anomalies before performing breakout analysis. To this end, we used the S-H-ESD algorithm [13] to automatically detect anomalies. Subsequently, the anomalies were removed and breakout was detected using both `PELT` and `E-Divisive` – see columns 6 and 7 of Table 1. However, we do not consider this an ideal approach as anomaly and breakout detection are tightly intertwined. This stems from the fact that breakouts can cause normal observations to appear as anomalies, whereas anomalies can cause the data to appear to have a different mean. Unlike the local smoothing approach preemptive anomaly removal effects both `E-Divisive` and `PELT`. Both algorithms become less able to identify a change, as is expected because of the relationship between breakout and anomaly detection.

4.3 EDM

We next evaluated the efficacy of **EDM**. The TTD values for `E-Divisive`, **EDM-X** and **EDM** are reported in Table 2. Recall that **EDM** is designed to detect breakouts in an anomaly "aware" fashion. From the table we note that in most cases that TTD values are in the same ballpark as in the case of `E-Divisive`. In a couple of cases – Datasets 10 and 20 – both **EDM-X** and **EDM** outperform `E-Divisive` significantly, see Figures 7a and 7b. From Figure 7a we note that, unlike `E-Divisive`, **EDM-X** was able to detect the true location of the change in mean. This is due to fact that **EDM-X** was not susceptible to the anomalies at the left hand side of the time series. Likewise, from Figure 7b we note that **EDM** is robust against the anomalies on the right hand side of the true location of mean change; hence, **EDM** returned a very accurate estimate of the breakout.

Amongst **EDM-Head** and **EDM-Tail**, the latter seem to perform better in most cases. This is desirable from a recency perspective. Only in the case of Dataset 13 **EDM-Tail** performs significantly worse than `E-Divisive`.

The *Precision, Recall* and *F-measure* for both **EDM-X** and **EDM** is reported in Table 3. From the table we note **EDM-X** has a higher *F-measure* than **EDM-Head** and **EDM-Tail** for the data sets we used. The approximate p-values obtained using the permutation test (detailed in subsection 2.2) for each run are tabulated in Table 4. From the table we see that in some cases the p-value is higher than our threshold of 5%.

Based on our experimental results, we argue for the use of **EDM** when it is suspected that anomalies might be present in a given time series. In addition, the run time of **EDM-X** and **EDM** is much smaller to that of `E-Divisive`, see

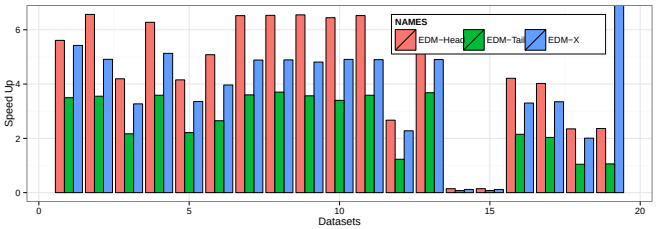


Figure 8: Speedup of **EDM** and **EDM-X** relative to `E-Divisive`

[3]Note that the presence of anomalies in production cloud is not uncommon [13].

	E-Div	**EDM-X**	**EDM-Head**	**EDM-Tail**
Dataset 1	0	0	4	6
Dataset 2	0	64	84	0
Dataset 3	2	12	2	0
Dataset 4	0	0	3	2
Dataset 5	0	0	5	1
Dataset 6	2	1	0	0
Dataset 7	6	6	26	6
Dataset 8	3	1	69	11
Dataset 9	9	3	8	0
Dataset 10	14113	8	1489	43
Dataset 11	0	66	5	1
Dataset 12	0	0	47	2
Dataset 13	45	215	246	1332
Dataset 14	0	78	5	0
Dataset 15	1	3	9	4
Dataset 16	0	268	89	95
Dataset 17	2	1	122	1
Dataset 18	1	26	0	0
Dataset 19	2	27	4	1
Dataset 20	4479	183	55	3
Dataset 21	27	70	204	78
Dataset 22	0	0	34	4
Dataset 23	4	19	0	4
Dataset 24	32	143	47	3
Dataset 25	0	11	6	2

Table 2: TTD for various nonparametric breakout procedures. **EDM-Head** and **EDM-Tail** refer to the **EDM** algorithm when the δ between distance observations is chosen according to the Figures 4(a) and 4(b) respectively

Figure 8. In our analysis, when performing the permutation test for **EDM** and **EDM-X**, the maximum number of permutations were always performed. However, the implementation of `E-Divisive` in the **ecp** package allows for early termination of the permutation test. Inspite of this, Figure 8 shows that **EDM** and **EDM-X** are at least 2× as fast as `E-Divisive` in almost all cases, and sometimes 6× faster.

Even though the **EDM** and **EDM-X** algorithms have been shown to be competitive with `E-Divisive` in the absence of anomalies, and better in the presence of anomalies, these methods do have their own limitations. For instance, see Figure 9. From the figure we note that **EDM** reports an inaccurate breakout estimate. This is attributed to the large number of anomalies as well as the fact that the anomalies are closely intertwined with the normal observations.

Another limitation of **EDM** and **EDM-X** is that they are both only able to detect a single breakout. Thus, if more than one breakout exists, it is unclear which (if any) will be found by **EDM-X** and **EDM**. Furthermore, depending on the size and nature of the breakouts, it is possible for performance to degrade, i.e., TTD may increase. This results from the fact that both **EDM** and **EDM-X** attempt to partition the time series into two homogeneous segments.

	EDM-X	**EDM-Head**	**EDM-Tail**
Precision	0.8400	0.9048	0.9048
Recall	1	0.8261	0.8261
F-Measure	0.9130	0.8636	0.8636

Table 3: Precision, recall, and F-Measure for **EDM-X** and **EDM**

	EDM-X	**EDM-Head**	**EDM-Tail**
Dataset 1	0.005	0.130	0.115
Dataset 2	0.005	0.005	0.005
Dataset 3	0.005	0.005	0.005
Dataset 4	0.005	0.005	0.005
Dataset 5	0.005	0.100	0.050
Dataset 6	0.005	0.005	0.005
Dataset 7	0.005	0.005	0.005
Dataset 8	0.005	0.035	0.120
Dataset 9	0.005	0.005	0.015
Dataset 10	0.005	0.005	0.005
Dataset 11	0.005	0.005	0.005
Dataset 12	0.005	0.015	0.010
Dataset 13	0.005	0.010	0.010
Dataset 14	0.005	0.005	0.005
Dataset 15	0.005	0.005	0.005
Dataset 16	0.005	0.005	0.005
Dataset 17	0.005	0.005	0.005
Dataset 18	0.005	0.005	0.005
Dataset 19	0.005	0.085	0.020
Dataset 20	0.005	0.005	0.005
Dataset 21	0.005	0.925	0.990
Dataset 22	0.005	0.020	0.985
Dataset 23	0.005	0.005	0.005
Dataset 24	0.005	0.025	0.030
Dataset 25	0.005	0.005	0.005

Table 4: Approximate p-values obtained from permutation test (detailed in subsection 2.2)

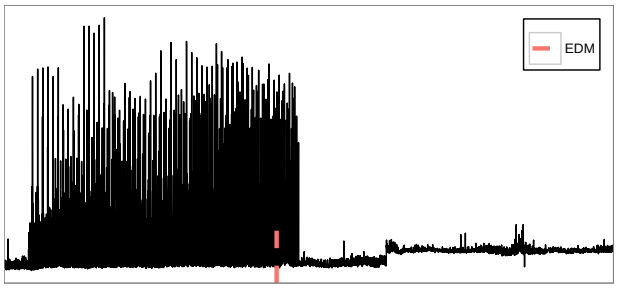


Figure 9: An example illustrating limitations of **EDM**

5. RELATED WORK

Breakout detection has been research in a wide variety of fields owing to the different applications. In this section we present a brief overview of prior work in breakout detection in statistics, finance, medicine and signal processing.

As mentioned earlier, *breakout* is referred to as a *changepoint* in statistics. Changepoint detection has been researched in statistics for over five decades [32, 33, 34, 35]. These come in two flavors: parametric and non-parametric. Many of the existing parametric methods assume that the underlying distribution belongs to the exponential family [35]. There has

been recent research in detecting changes with heavy tailed distributions [36]. Many of these approaches make use of limiting distributions obtained from Extreme Value Theory [37]. In cases where it is difficult or impossible to prove that the data adheres to parametric assumptions non-parametric approaches provide an alternative solution. These methods place less restrictive assumptions on the data and can thus be used more widely in general; however, due to the weaker assumptions, these methods are less powerful than their parametric counterparts [38]. Although most of the prior researched centered around detecting changes in mean, detecting changes in variance (with known/unknown mean value) has garnered some attention [39, 40, 41].

Tsay [42] presents an approach to detect changes in mean of an ARMA model in the presence of anomalies. Unlike **EDM**, the approach employs a two staged process that first removes the anomalies and then carry out breakout analysis. Another approach to handle anomalies during breakout detection is to assume that the data follows a heavy tailed distribution [43] and thus large values become less uncommon [44].

5.0.1 Parametric Analysis

The parametric algorithms used to perform breakout analysis assume that the observed distributions belong to a family of distributions $\mathcal{F}_\Theta = \{F_\theta : \theta \in \Theta\}$, such that each member of the family can be uniquely identified by its parameter value. Once the class of distributions has been specified, parametric methods attempt to detect changes in the parameter value. Specifically, these approaches usually attempt to maximize a likelihood. For example, Carlin et al., [45], Lavielle and Teyssière [46] employ this approach. These papers however, assume a Gaussian distribution. An extension of this to select methods of the exponential family [47] is supported in the `changepoint` R package [30].

5.0.2 Non-parametric Analysis

A very common approach is to perform density estimation [48]. Although density estimation seems like a natural approach, other ideas have been shown to yield satisfactory results. For example, Lung-Yut-Fong et al. [49] perform analysis by working with rank statistics; Matteson and James [31] present an approach based upon Euclidean distances.

5.1 Finance

One of the more popular application areas of breakout detection is finance [50, 51, 52]. In this regard, models are regularly used to analyze return and stock price data. It is often assumed that the model parameters remain constant over the observed period. However, if the parameters are mostly time varying, the obtained results are likely to become out-of-date and consequently may not be robust [51]. Explicit examples of trading strategies that make use of breakout detection can be found in [51] which rely on historical analysis, charts and familiarity with the market.

The ARCH model of Engle [53] and its various generalizations are very often used to model the returns for a number of financial instruments. Franses and Ghijsels [54] present a method for fitting GARCH models to financial data that may have additive outliers. In a similar vein, in [55], Matteson and James presented an approach that only requires a few mild statistical conditions to hold and doesn't rely on any back testing. Regardless of the strategy, both works show that acknowledging the existence of breakouts can increase profits, or better yet, change would be losses into gains.

5.2 Medical Applications

Breakout detection also has applications in medicine. For example, Grigg et al. describe the use of the cumulative sum (CUSUM) chart, RSPRT (resetting sequential probability ratio test), and FIR (fast initial response) CUSUM to detect improvements in a process as well as detecting deterioration in a medical setting. In genetics, array comparative genomic hybridization is used to record DNA copy numbers. Changes in the DNA copy number can indicate a portion of a gene that may be effected by cancer or some other abnormal feature. Thus, detecting breakout in this setting [56, 57] can provide insights about future medical research. Breakout analysis also finds application in segmentation of electroencephalogram (EEG). An EEG is a measure of the brain's electrical activity which is recoded by electrodes on the subject's scalp. EEGs can be used in the process of diagnosing disorders such as epilepsy and insomnia, since such disorders cause clear changes in the EEG readings. Breakout procedures have been suggested as a way to remove the human bias in the analysis of such data [58, 59]. Other application areas include studying breast cancer survival rates [60], analysis of fMRI data [61], and many more [35].

5.3 Signal Processing

Breakouts detection has been researched in the field of signal processing (and others such as, but not limited to, computer vision, image processing) but is usually referred to *edge detection* or *jump detection* [62, 63, 64, 65]. In [66], Basseville presented a survey of techniques to detect changes in signals and systems; Ziou and Tabbone present an overview of edge detection techniques in [67]. In the context of dynamic systems, Tugnait presented techniques to detect changes in [68].

In [69], Jackson et al. presented an algorithm for optimal partitioning of data on an interval. The algorithm was subsequently enhanced by Killick et al. [17] to detect breakouts with an expected linear running time.

6. CONCLUSIONS

In this paper, we proposed a novel statistical technique, called **E-Divisive with Medians (EDM)**, to automatically detect breakouts in cloud data. Unlike the existing techniques for breakout detection, **EDM** is robust against the presence of anomalies. **EDM** employs E-statistics [14] to detect divergence in mean. Note that, in general, **EDM** can also be used detect change in distribution in a given time series. Further, **EDM** uses robust statistical metrics, viz., median, and estimates the statistical significance of a breakout through a permutation test. We used production data and to evaluate the efficacy of **EDM** and reported Precision, Recall and F-measure to demonstrate the same. **EDM** is 3.5× faster than the state-of-the-art technique for breakout detection and is being currently used on a daily basis at Twitter.

As future work, we intend to extend **EDM** to support breakout detection in the presence of seasonality. Further, we plan to explore data transformation techniques to address the limitations mentioned in Section 4.

7. REFERENCES

[1] M. Meeker. Internet Trends 2014 - Code Conference. http://www.kpcb.com/file/kpcb-internet-trends-2014, May 2014.

[2] Handset Data Traffic (2001-2017). http://www.strategyanalytics.com/default.aspx?mod=reportabstractviewer&a0=8623.

[3] How Loading Time Affects Your Bottom Line. http://blog.kissmetrics.com/loading-time/.

[4] Impact of web latency on conversion rates. http://www.slideshare.net/bitcurrent/impact-of-web-latency-on-conversion-rates.

[5] Mobile Application Performance Testing: Aluminum foil, elevators and other mobile testing myths debunked. http://media.shunra.com/whitepapers/MobilePerfTesting_27913.pdf.

[6] B. Chatham, B. D. Temkin, and M. Amato. A primer on a/b testing. In *Forrester Research*, 2004.

[7] R. Kohavi, R. Longbotham, D. Sommerfield, and R. M. Henne. Controlled experiments on the web: survey and practical guide. *Data Mining and Knowledge Discovery*, 18(1):140–181, 2009. http://link.springer.com/content/pdf/10.1007%2Fs10618-008-0114-1.pdf.

[8] Dan Siroker and Pete Koomen. *A/B Testing: The Most Powerful Way to Turn Clicks Into Customers*. Wiley Publishing, 2013.

[9] U. Hölzle and L. A. Barroso. *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines*. Morgan and Claypool Publishers, 2009.

[10] Y. S. Dai, B. Yang, J. Dongarra, and G. Zhang. Cloud service reliability: Modeling and analysis. http://www.netlib.org/utk/people/JackDongarra/PAPERS/Cloud-Shaun-Jack.pdf.

[11] K. V. Vishwanath and N. Nagappan. Characterizing cloud computing hardware reliability. In *Proceedings of 1st ACM Symposium on Cloud computing*, pages 193–204, 2010.

[12] B. Lorica. How twitter monitors millions of time-series. http://strata.oreilly.com/2013/09/how-twitter-monitors-millions-of-time-series.html, 2013.

[13] Owen Vallis, Jordan Hochenbaum, and Arun Kejariwal. A novel technique for long-term anomaly detection in the cloud. In *6th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 14)*, June 2014.

[14] Gábor J Székely and Maria L Rizzo. Energy statistics: A class of statistics based on distances. *Journal of Statistical Planning and Inference*, 143(8):1249–1272, 2013.

[15] Peter J Huber and Elvezio Ronchetti. *Robust statistics*. Wiley, Hoboken, N.J., 1981.

[16] Frank R Hampel, Elvezio Ronchetti, Peter J. Rousseeuw, and Werner A. Stahel. *Robust statistics: the approach based on influence functions*. Wiley, New York, 1986.

[17] Rebecca Killick, Paul Fearnhead, and IA Eckley. Optimal detection of changepoints with a linear computational cost. *Journal of the American Statistical Association*, 107(500):1590–1598, 2012.

[18] M. L. Rizzo and G. J. Székely. DISCO analysis: A nonparametric extension of analysis of variance. *The Annals of Applied Statistics*, 4(2):1034–1055, 2010.

[19] G. J.Székely and M. L. Rizzo. Hierarchical clustering via joint between-within distances: Extending ward's minimum variance method. *Journal of classification*, 22(2):151–183, 2005.

[20] Wassily Hoeffding. The strong law of large numbers for u-statistics. *Institute of Statistics mimeo series*, 302, 1961.

[21] A. Gandy. Sequential implementation of monte carlo tests with uniformly bounded resampling risk. *Journal of the American Statistical Association*, 488(104):1504–1511, 2009.

[22] Tan Pang-Ning, Michael Steinbach, Vipin Kumar, et al. Introduction to data mining. In *Library of Congress*, 2006.

[23] Nicholas A. James and David S. Matteson. ecp: An R package for nonparametric multiple change point analysis of multivariate data. Technical report, Cornell University, 2013.

[24] D. Keim, G. Andrienkoa, J.-D. Fekete, C. Görg, J. Kohlhammer, and G. Melancon. Visual analytics: Definition, process, and challenges. In A. Kerren, J. T. Stasko, J.-D. Fekete, and C. North, editors, *Information Visualization*, pages 154–175. Springer-Verlag, 2008.

[25] Pak Chung Wong and Jim Thomas. Visual analytics. *IEEE Computer Graphics and Applications*, 24(5):20–21, 2004.

[26] J. Pitt, A. Bourazeri, A. Nowak, M. Roszczynska-Kurasinska, A. Rychwalska, I. R. Santiago, M. L. Sanchez, M. Florea, and M. Sanduleac. Transforming big data into collective awareness. *IEEE Computer*, 46(6):40–45, 2013.

[27] Pak Chung Wong, Han-Wei Shen, and Valerio Pascucci. Extreme-scale visual analytics. *IEEE Computer Graphics and Applications*, 32(4):23–25, 2012.

[28] H. Childs, B. Geveci, W. Schroeder, J. Meredith, K. Moreland, C. Sewell, T. Kuhlen, and E. W.Bethel. Research challenges for visualization software. *Computer*, 46(5):34–42, 2013.

[29] R. Kosara and J. Mackinlay. Storytelling: The next step for visualization. *Computer*, 46(5):44–50, 2013.

[30] Rebecca Killick and Kaylea Haynes. *changepoint: An R package for changepoint analysis*, 2014. R package version 1.1.2.

[31] David S. Matteson and Nicholas A. James. A nonparametric approach for multiple change point analysis of multivariate data. *Journal of the American Statistical Association*, 109(505):334–345, 2013.

[32] ES Page. A test for a change in a parameter occurring at an unknown point. *Biometrika*, 42(3-4):523–527, 1955.

[33] E Brodsky and Boris S Darkhovsky. *Nonparametric methods in change point problems*. Number 243. Springer, 1993.

[34] Miklós Csörgö and Lajos Horváth. *Limit theorems in change-point analysis*. Wiley New York, 1997.

[35] Jie Chen and Arjun K Gupta. *Parametric Statistical Change Point Analysis: With Applications to Genetics, Medicine, and Finance*. Springer, 2011.

[36] Marc Raimondo and Nader Tajvidi. A peaks over threshold model for change-point detection by wavelets. *Statistica Sinica*, 14(2):395–412, 2004.

[37] Emil Julius Gumbel. *Statistics of Extremes*. Courier Dover Publications, 2012.

[38] Changliang Zou, Guosheng Yin, Long Feng, Zhaojun Wang, et al. Nonparametric maximum likelihood approach to multiple change-point problems. *The Annals of Statistics*, 42(3):970–1002, 2014.

[39] Jie Chen and AK Gupta. Testing and locating variance changepoints with application to stock prices. *Journal of the American Statistical Association*, 92(438):739–747, 1997.

[40] O Sola Adegboye and AK Gupta. On testing against restricted alternatives for the variances of gaussian models. *Australian Journal of Statistics*, 31(3):409–415, 1989.

[41] Brandon Whitcher, Simon D Byers, Peter Guttorp, and Donald B Percival. Testing for homogeneity of variance in time series: Long memory, wavelets, and the nile river. *Water Resources Research*, 38(5):12–1, 2002.

[42] Ruey S Tsay. Outliers, level shifts, and variancechanges in time series. *Journal of Forecasting*, 7(1):1–20, 1988.

[43] Mico Loretan and Peter CB Phillips. Testing the covariance stationarity of heavy-tailed time series: An overview of the theory with applications to several financial datasets. *Journal of empirical finance*, 1(2):211–248, 1994.

[44] Paul Embrechts, Sidney I Resnick, and Gennady Samorodnitsky. Extreme value theory as a risk management tool. *North American Actuarial Journal*, 3(2):30 – 41, 1999.

[45] Bradley P Carlin, Alan E Gelfand, and Adrian FM Smith. Hierarchical bayesian analysis of changepoint problems. *Applied statistics*, pages 389–405, 1992.

[46] Marc Lavielle and Gilles Teyssiere. Detection of multiple change-points in multivariate time series. *Lithuanian Mathematical Journal*, 46(3):287–306, 2006.

[47] Erling Bernhard Andersen. Sufficiency and exponential families for discrete sample spaces. *Journal of the American Statistical Association*, 65(331):1248–1255, 1970.

[48] Yoshinobu Kawahara and Masashi Sugiyama. Sequential change-point detection based on direct density-ratio estimation. *Statistical Analysis and Data Mining*, 5(2):114–127, 2012.

[49] Alexandre Lung-Yut-Fong, Céline Lévy-Leduc, and Olivier Cappé. Homogeneity and change-point detection rests for multivariate data using rank statistics. *arXiv preprint arXiv:1107.1971*, 2011.

[50] Charles D Kirkpatrick II and Julie Dahlquist. *Technical analysis: the complete resource for financial market technicians*. FT press, 2010.

[51] Adam Grimes. *The Art & Science of Technical Analysis: Market Structure, Price Action & Trading Strategies*, volume 548. John Wiley & Sons, 2012.

[52] Robert D Edwards, John Magee, and WHC Bassetti. *Technical analysis of stock trends*. CRC Press, 2012.

[53] Robert F Engle. Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. *Econometrica: Journal of the Econometric Society*, pages 987–1007, 1982.

[54] Philip Hans Franses and Hendrik Ghijsels. Additive outliers, GARCH and forecasting volatility. *International Journal of Forecasting*, 15(1):1–9, 1999.

[55] David S Matteson, Nicholas A James, William B Nicholson, and Louis C Segalini. Locally stationary vector processes and adaptive multivariate modeling. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 8722–8726. IEEE, 2013.

[56] Adam B Olshen, ES Venkatraman, Robert Lucito, and Michael Wigler. Circular binary segmentation for the analysis of array-based dna copy number data. *Biostatistics*, 5(4):557–572, 2004.

[57] Kevin Bleakley and Jean-Philippe Vert. The group fused lasso for multiple change-point detection. *arXiv preprint arXiv:1106.4199*, 2011.

[58] JS Barlow, OD Creutzfeldt, D Michael, J Houchin, and H Epelbaum. Automatic adaptive segmentation of clinical eegs. *Electroencephalography and Clinical Neurophysiology*, 51(5):512–525, 1981.

[59] A.Ya. Kaplan and S.L. Shishkin. Application of the change-point analysis to the investigation of the brains electrical activity. In *Non-Parametric Statistical Diagnosis*, volume 509 of *Mathematics and Its Applications*, pages 333–388. Springer Netherlands, 2000.

[60] Cćile Contal and John O'Quigley. An application of changepoint methods in studying the effect of age on survival in breast cancer. *Computational Statistics & Data Analysis*, 30(3):253 – 270, 1999.

[61] Lucy F. Robinson, Tor D. Wager, and Martin A. Lindquist. Change point estimation in multi-subject fmri studies. *NeuroImage*, 49(2):1581 – 1592, 2010.

[62] A.S. Willsky and H.L. Jones. A generalized likelihood ratio approach to the detection and estimation of jumps in linear systems. *Automatic Control, IEEE Transactions on*, 21(1):108–112, Feb 1976.

[63] J. Segen and A.C. Sanderson. Detecting change in a time-series. *Information Theory, IEEE Transactions on*, 26(2):249–254, Mar 1980.

[64] M. Basseville, B. Espiau, and J. Gasnier. Edge detection using sequential methods for change in level–part i: A sequential edge detection algorithm. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 29(1):24–31, Feb 1981.

[65] Albert Benveniste, Michle Basseville, and George Moustakides. The asymptotic local approach to change detection and model validation. Research Report 564, September 1986.

[66] Michèle Basseville. Detecting changes in signals and systemsa survey. *Automatica*, 24(3):309–326, 1988.

[67] Djemel Ziou, Salvatore Tabbone, et al. Edge detection techniques-an overview. *Pattern Recognition And Image Analysis C/C Of Raspoznavaniye Obrazov I Analiz Izobrazhenii*, 8:537–559, 1998.

[68] Jitendra K Tugnait. Detection and estimation for abruptly changing systems. *Automatica*, 18(5):607–615, 1982.

[69] Brad Jackson, Jeffrey D Scargle, David Barnes, Sundararajan Arabhi, Alina Alt, Peter Gioumousis, Elyus Gwin, Paungkaew Sangtrakulcharoen, Linda Tan, and Tun Tao Tsai. An algorithm for optimal partitioning of data on an interval. *Signal Processing Letters, IEEE*, 12(2):105–108, 2005.

8. APPENDIX

In this appendix we present an in-depth description of the details necessary to implement both **EDM** and **EDM-X**, as well as the interval tree used to calculate approximate medians. All of these algorithms assume that our time series values lie within the interval $[0, 1]$. Thus if $M = \max\{Z_i : 1 \leq i \leq n\}$ an $m = \min\{Z_i : 1 \leq i \leq n\}$ we transform our observations according to the following linear function

$$f(x) = \frac{x - m}{M - m}.$$

It should be noted that this transformation only scales the value of our approximate (or true) median by a value of $\frac{1}{M-m}$.

8.1 Interval Trees

In this subsection we detail how interval trees are used by **EDM** and **EDM-X**. Our interval tree is a complete binary tree with 2^D leaf nodes, where D is the user specified depth. The ith leaf node represents the interval $\left[\frac{i-1}{2^D}, \frac{i}{2^D}\right)$, except for the 2^Dth interval which is a closed interval, instead of

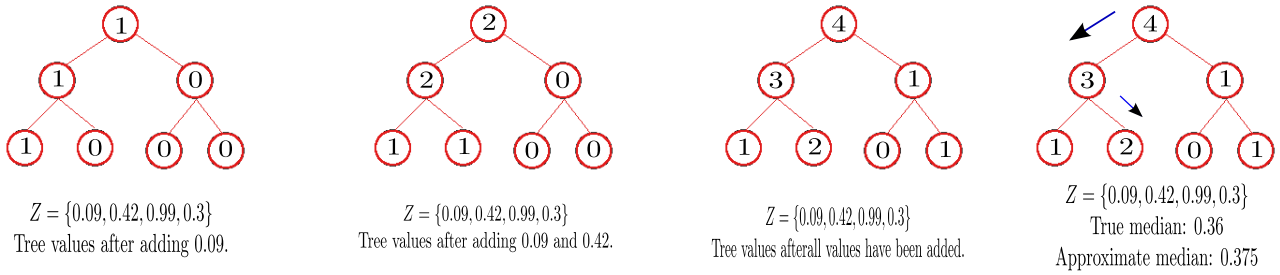


Figure 10: Illustration of the use of interval trees to determine approximate median

a half-open interval. Each internal node corresponds to the union of the intervals of its children. Thus, the root represents the interval $[0, 1]$. In this data structure each node will contain a count of the number of observations that lie within its interval.

Owing to the nature of the tree, one can find an approximate median in $\mathcal{O}(D)$ time. One can find a value m, such that $K = \lceil \frac{n}{2} \rceil$ of our observations are less that or equal to m in the following manner: Starting at the root node compare the value of its left child with K. If its value is larger than K, move to that node. On the other hand, if K is larger, subtract the value of the left node from K and move to the right child. This procedure is continued until a leaf node is reached; then, the midpoint of the leaf's corresponding interval is returned. However, if at some point an internal node is reached who's value is equal to K, the following is carried out: Let a and b be the values of the left and right children respectively, and x, y the midpoints of their corresponding intervals. The following is returned:

$$\frac{1}{a+b}\left(a \times x + b \times y\right)$$

The major benefit of using an interval tree to obtain an approximate median instead of finding the true median is that the data structure can be updated efficiently and does not require sorting. Furthermore, from our experiments we have found that the relative difference between the true median and the approximation to be below 10%. Figure 10 illustrates how to update the tree as well as how to an approximate median.