Actionable Insights in Multivariate Time-series for Urban Analytics

Anika Tabassum
Department of Computer Science, Virginia Tech
anikat1@vt.edu

Supriya Chinthavali
Oak Ridge National Laboratory
chinthavalis@ornl.gov

Varisara Tansakul
Oak Ridge National Laboratory
tansakulv@ornl.gov

B. Aditya Prakash
College of Computing, Georgia Tech
badityap@cc.gatech.edu

ABSTRACT

Multivariate time-series are gaining popularity in various urban applications regarding emergency management. Segmentation algorithms mostly focus on identifying discrete events with changing phases in such data. For e.g., consider a power outage scenario during a hurricane. Each time-series may represent the number of power outages in a county for a time-period. Segments in such time-series are found in terms of different phases, such as, a hurricane starts, counties face severe-damage, and hurricane ends. Disaster management experts typically want to identify the most affected culprit counties during these phases. These can be effective for decision making regarding resource allocation to those regions to mitigate the damage. However, getting these actionable 'culprits' directly (either by visualizing or looking into the segmentation algorithm) is typically hard. Hence we introduce a novel problem RaTSS (Rationalization for time-series segmentation), that aims to find such actionable culprits (rationalizations) for the segmentation.

CCS CONCEPTS

• **Computing methodologies** → *Machine learning approaches.*

KEYWORDS

multivariate time-series, urban analytics, rationalization

ACM Reference Format:
Anika Tabassum, Supriya Chinthavali, Varisara Tansakul, and B. Aditya Prakash. 2021. Actionable Insights in Multivariate Time-series for Urban Analytics. In *MileTS '21: 7th KDD Workshop on Mining and Learning from Time Series, August 14th, 2021, Singapore.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

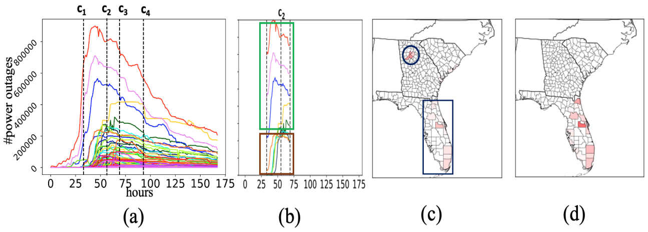


Figure 1: Disaster example (a): 2017 Hurricane Irma times-series (representing power failures of a county during the hurricane). The first and last cut point are based on the hurricane landfall (Sept. 10) and end time (Sept. 12). Cutpoints c_2 and c_3 are around the time when hurricane is changing trajectory. (b) shows a snippet of c_2. (c) a heatmap of top 10 most important culprit counties found by our algorithm across cutpoint c_2. (d) shows a simple magnitude-based culprits heatmap across c_2. Brighter colors indicate larger importance weights ((c),(d)).

Motivation: Multivariate time-series data, where each timestamp has readings from the observations of multiple entities or sensors, are prevalent in various urban scenarios, ranging from emergency management, public health, and so on. Urban domain experts (DEs) such as emergency management and health care authorities segment such data for identifying meaningful events [26, 27]. They want to understand which signals are the 'culprit' or more susceptible across each particular event [15, 19, 22, 28]. For e.g., in a hurricane disaster scenario, a set of multivariate time-series can represent power failures across a set of counties. Further DEs can identify the events which correspond to different phases of the hurricanes (e.g., based on severity of damage), by using multiple segmentation algorithms [9, 12, 14, 16, 18]. However, most time-series segmentation algorithms [11, 12, 17, 18] are usually complex, and while they give high quality segmentations, they do not give ready actionable insights to identify the culprits across events.

Finding 'culprits': We next discuss how finding culprits gives actionable insights which are useful for domain experts in the context of urban analytics. For the hurricane power failure data collected

This document has been authored by UT-Battelle, LLC, under contract DE-AC05-00OR22725 with the US Department of Energy (DOE). The US government retains and the publisher, by accepting the article for publication, acknowledges that the US government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this manuscript, or allow others to do so, for US government purposes. DOE will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

MileTS '21, August 14th, 2021, Singapore
© 2021 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-9999-9/18/06...$15.00
https://doi.org/10.1145/1122445.1122456

by DEs, such as emergency management authorities seek solutions to reduce power outages in different counties [10]. If DEs know which time-series/counties are the culprits (most important) with respect to an event, they can send personnel to fix damage and alert local authorities to reduce further loss [7]. For example, Fig. 1(a) shows the time-series data of Hurricane Irma with the segmentation, which roughly correspond to different phases. Fig. 1(b) shows the failure time-series snippet of top 10 'culprit' counties (i.e. time-series) across the second cutpoint (c_2) found by our algorithm. We see some counties have a very high change of power failures (green box) which are obviously useful to guide resource allocation as they denote widespread failures. At the same time there are some less obvious counties near Atlanta area (brown box in Fig 1(b)). These counties have relatively lower change but the change of failures is more sudden compared to the other counties. Fig. 1(c) shows the the geographic heatmap plot of our culprits (based on their importance weights as learnt by our algorithm). Clearly these counties (denoted by blue circle) are not obvious: they are located at the north-west corner, far away from where the hurricane is present (the blue rectangle, where the rest of the important culprits are). However they are still important and useful for situational awareness and resource allocation. Indeed, report [3] suggests that this is due to a separate tropical storm happening at this time. Note that figuring out these counties by just visualizing the time-series (Fig. 1(a)) across c_2 is hard (as they are buried). In fact, if we plot a heatmap of all the counties based on their change of magnitude across c_2 (Fig. 1(d)) we *cannot* recover these counties located at the middle (as their magnitudes are relatively lower).

How can we then find such 'culprits' for events? One plausible way to solve such problems might be to somehow interpret the internal change of state of the segmentation algorithm across any cut-point [20]. For instance, one might use a recent segmentation algorithm Autoplait [18] to segment, which learns Hidden Markov Model (HMM) internally. Hence one way to get the culprits will be to see which time-series cause the internal HMM to change across the cut-point. However it is easy to see these will only point to time-series which help explain *model behavior*, but not necessarily give actionable insights to the DE (as explained above such actionable insights are based more on the change in the time-series itself, and how similar/dissimilar these changes are to others). Additionally, DEs may end up using multiple segmentation algorithms (such as TICC [12] which uses multilayer MRF, [23] which uses temporal mixture model in contrast to Autoplait) for different datasets based on what constitute meaningful events for the data. Tracking the model behavior for each of these segmentation algorithms will give different time-series as culprits. Further this may become too complex to the DE, who will need additional technical help to understand the technical intricacies of the segmentation algorithm to get any actionable insights.

Our Contributions: Hence, we need a different way to identify actionable culprits for any segmentation algorithm. We propose a novel problem *Rationalization for time-series segmentation* (RaTSS) which aims to find human-friendly and actionable culprits (rationalizations) for the urban experts across the associated events in terms of *constituent time-series.* We also propose an efficient algorithm Find-RaTSS to solve RaTSS and finally quantitatively and qualitatively evaluate our performance on general and disaster data, i.e., Hurricane power failure and the recent COVID-19 pandemic. This paper is a collaboration between computer scientists and emergency management experts. To the best of our knowledge, no methods have been proposed so far for identifying culprits that can work for any black-box segmentation on multivariate time-series.

The rest of the paper is organized as usual. All the proofs and additional experiments are in the supplementary.

2 OUR APPROACH

Notations. Suppose, we have a multivariate time-series data matrix of m sequences $\mathbf{X} = \{x_1, x_2, ...x_m\}$, where each $x_u = x_u(t_1), ...x_u(t_t)$ has t observations. We are given a set of cutpoints $C = \{c_1, c_2, \ldots, c_k\}$, each $1 \leq c_j \leq t$.

Table 1: Notations.

Symbol	Description
$\mathbf{X} \in \mathbb{R}^{m \times T}$	Data matrix consisting of m time-series each having t timestamps
C	Segmentation with a set of cutpoints for $\mathbf{X}$
B	Any segmentation algorithm which outputs a set of cutpoints on $\mathbf{X}$.
$\mathbf{r}_j$	$m \times 1$ rationalization weight vector
$G(\mathcal{S}, \mathcal{E})$	A segment graph of $\mathcal{S}$ nodes (s_{ij}) and $\mathcal{E}$ edges (e_{ijk}) connected by weight $\mathbf{w}_{ijk}$
s_{ij}	A node (segment) of G consisting of the sub-sequence of m time-series within $i - j$ timestamps
e_{ijk}	An edge of G representing edge weight of the connected nodes s_{ij} & s_{jk}
$\mathbf{w}_{ijk}$	A $m \times 1$ weight vector for e_{ijk}, each cell is weight of x_i in e_{ijk}
K	Total number of paths in G
$F(s_{ij})$	A function to return a $m \times f$ matrix for each time-series in s_{ij} having f features
P_B	The path in G which maps to C
P_{rest}	All possible paths in G except the path π_B
$\boldsymbol{\pi}_B$, $\boldsymbol{\pi}_{\text{rest}}$	Cost of path P_B, P_{rest}
$\boldsymbol{\alpha}$	Global latent weight vector

The main challenges for formulating a rationalization problem are: **P1.** We require a general framework which works for any $\mathbf{X}$ and any segmentation C given by a black-box segmentation algorithm. **P2.** We need to associate constituent time-series across each cutpoint to actionable directives. Hence we propose an intermediary weighting scheme to measure importance of each time-series which can map them towards actionable directives.

Problem 2.1 (**Informal RaTSS.**). ***Given*** *a multivariate time-series* $\mathbf{X}$ *and a segmentation* C *for* $\mathbf{X}$*.* ***Find*** *the rationalization weight vector of size* $m \times 1$, $\mathbf{r}_j$ *across each cutpoint* c_j *in* C*, where each value* r_j^u *in* $\mathbf{r}_j$ *is a scalar and represents the importance weight for time-series* x_u *across* c_j*.*

What is r_j^u? One possible approach to measure importance is to assign a numerical weight r_j^u to the time-series x_u in terms of its change across a cutpoint c_j. However, we do not know anything about the segmentation algorithm B (**P1**), and even the way we should measure change (e.g., using time-series features) may be different for each cut-point. Hence feature-engineering to work for all B correctly is not possible. Since C is selected by B among all possible segmentations, it is intuitive to assume C by B is the best in some sense (as B outputs the segmentation C, choosing it over *all the other possible segmentations*). Thus our main idea is to choose $\mathbf{r}_j$s which can consider C the best compared to other possible segmentations.

Segment Graph. To efficiently represent all possible segmentations (as they will be exponential to the length of the sequence), we leverage a 'segment graph' data structure [9] and convert our rationalization problem in terms of a graph optimization. The segment graph $G(\mathcal{S}, \mathcal{E})$ is a *Directed Acyclic Weighted Graph* (DAWG) consisting of a set of nodes $\mathcal{S}$ and set of edges $\mathcal{E}$. The node set $\mathcal{S} = \{s_o, \{s_{ij}\}, t_o\}$ consists of all possible segments or sub-sequence in $\mathbf{X}$ and two dummy nodes s_o, t_o, to represent start and end of a path in G. The edge set $\mathcal{E} = \{e_{ijk}\}$ connects two adjacent segments s_{ij}, s_{jk}, where $i < j < k$. Each e_{ijk} is mapped to a possible cutpoint c_j and the edge weight vector $\mathbf{w}_{ijk}$ for the corresponding e_{ijk} represents the 'change' of every x_u between the adjacent segments s_{ij} and s_{jk}. P_B is the graph path in G, s.t. each edge $e_{ijk} \in P_B$ is mapped to the corresponding $c_j \in C$. P_{rest} is the set of all other possible paths in G other than P_B. Every path $p_v \in P_{\text{rest}}$ represents a possible segmentation of $\mathbf{X}$.

Thus, in terms of G our idea can be stated as, P_B should be the best path from s_o to t_a in G (compared to all paths in P_{rest}). $\mathbf{r}_j$ should be set in a way that helps make P_B the best path. The weight vectors $\mathbf{w}_{ijk}$ (how different adjoining segments are across each edge) also should play a role in quantifying the quality of each path. Clearly, we need a 'quality' metric (*Quality*) to compare the paths which depends on $\mathbf{r}_j$, P_B, P_{rest}, and $\mathbf{w}_{ijk}$. Then our required condition can be stated as:

Quality(P_B, $\mathbf{w}_{ijk}$ for each $e_{ijk} \in P_B, \mathbf{r}$) $>$ *Quality*(p_v,$\mathbf{w}_{ijk}$ for each $e_{ijk} \in p_v, \mathbf{r}$), $\forall p_v \in P_{\text{rest}}$. Creating 'rationalizations' $\mathbf{r}$ over all edges (to have affect on the quality of other paths) can lead to severe over-parameterization as the number of edges in G is $O(T^3)$. Hence we propose to instead have a *global latent* weight vector $\boldsymbol{\alpha} \in \mathbb{R}^{m \times 1}$ whose magnitude captures the global latent importance of each time-series, and then use $\boldsymbol{\alpha}$ over the edges in P_B to get $\mathbf{r}_j$ for each cutpoint $c_j \in C$, such that,

$$Quality(P_B, \alpha) > Quality(p_v, \alpha), \forall p_v \in P_{\text{rest}} \quad (1)$$

with $\mathbf{r}_j$ = someFunctionOf(α,$\mathbf{w}_{ijk}$). Using $\boldsymbol{\alpha}$ to calculate $\mathbf{r}_j$ can map constituent time-series be actionable. We plan to extend our problem in future for the scenarios where a *group* of time-series be actionable.

Problem 2.2 (**Informal Graph based RaTSS.**). ***Given, X**, C, and a segment graph G on **X**, so that its path P_B corresponds to C. **Find** $\boldsymbol{\alpha}$ to satisfy Eq. 1 and generate the rationalization weight vector $\mathbf{r}_j$ across each edge e_{ijk} in P_B using $\boldsymbol{\alpha}$.*

Formalizing Problem 2.2. To formalize this problem, several questions arise: **Q1.** *What is Quality?* We can assume *Quality* as *cost* of the path. The length of the path intuitively measures how different its adjacent segments are. The length $\boldsymbol{\pi}_v$ of a path p_v is an $m \times 1$ vector, each component represents the sum of all the edge weights $\mathbf{w}_{ijk}$ in p_v for time-series x_u. Hence, $\boldsymbol{\pi}_v = \sum_{e_{ijk} \in p_v} \mathbf{w}_{ijk}$. Further, segmentation algorithms try to find cutpoints so that each segment is homogeneous in some sense, and hence adjacent segments are expected to be very 'different' from one another. Thus in our framework, we want to learn $\boldsymbol{\alpha}$ which makes P_B the *longest* weighted path in G from s_o to t_a.

Hence *Quality* function for any path p_v (i.e. the cost of p_v) is: $Quality(p_v, \boldsymbol{\alpha}) = \sum_{e_{ijk} \in p_v} \boldsymbol{\alpha}^T \mathbf{w}_{ijk} = \sum_{e_{ijk} \in p_v} \sum_{u=1}^{m} \boldsymbol{\alpha}_u \mathbf{w}_{ijk}^u$.

Q2. *How to generate $\mathbf{r}_j$ for each c_j using $\boldsymbol{\alpha}$?* Mentioned above, $\mathbf{r}_j$ = someFunctionOf(α,$\mathbf{w}_{ijk}$). $\mathbf{w}_{ijk}$ represents the change across an edge, while $\boldsymbol{\alpha}$ gives us the global latent weight for each time-series. $\mathbf{r}_j$ is intuitively the importance of each time-series over every cut-point $c_j \in C$. Hence a simple way to get $\mathbf{r}_j$ is, $\mathbf{r}_j = |\alpha \odot \mathbf{w}_{ijk}|$, where $\odot$ is the standard element-wise vector dot product (and the modulus is for negative values of $\boldsymbol{\alpha}$).

Q3. *How to set $\mathbf{w}_{ijk}$?* As discussed, $\mathbf{w}_{ijk}$ should reflect some change in each time-series across the edge. For **P2**, we represent each node s_{ij} as a $m \times f$ feature matrix, $F(s_{ij}) = [\mathbf{f}_{ij}^1, ... \mathbf{f}_{ij}^m]$, each $\mathbf{f}^u$ is a $f \times 1$ feature vector of time-series x_u in segment s_{ij}. Then $\mathbf{w}_{ijk} = ||F(s_{ij}) - F(s_{jk})||_{1,2}$.

This is not feature engineering. Our intuition is to capture basic changes across the segments - not what the segmentation algorithm regards as the change, we aim to choose standard statistical features $\mathbf{f}$ (we use mean, variance, minimum and maximum in this paper as for **P1**, we cannot know the features used by B). A feature ablation test in the supplementary shows all these features are necessary and useful. This is different than directly choosing features to set $\mathbf{r}_j$, as $\mathbf{r}_j$ is necessary to best explain *why* the cut-point is present.

Putting everything together. We next formalize our task as an optimization problem. Note that Eq. 1 will result in one inequality for each path in G, which are exponential in number (to T). Instead, we tackle a simplified version, by just adding all the inequalities to get a consolidated objective. Let $\boldsymbol{\pi}_B$ be the length of path P_B. Similarly, total length of P_{rest}, i.e., $\boldsymbol{\pi}_{\text{rest}}$ is an $m \times 1$ vector, each component represents the length of all other paths in P_{rest} other than P_B over each time-series. Hence, $\boldsymbol{\pi}_{\text{rest}} = \sum_{p_v \in P_{\text{rest}}} \boldsymbol{\pi}_v = \sum_{p_v \in P_{\text{rest}}} \sum_{\substack{e_{ijk} \in p_v \\ e_{ijk} \notin P_B}} \mathbf{w}_{ijk}$.

Note that we can also rewrite $\boldsymbol{\pi}_{\text{rest}}$ as $\sum_{e_{ijk} \in \mathcal{E} - P_B} p_{ijk} \mathbf{w}_{ijk}$ where p_{ijk} is the number of paths in G passing through e_{ijk}. Further, let $\Delta\boldsymbol{\pi} = \sum_{p_v \in P_{\text{rest}}} \boldsymbol{\pi}_B - \boldsymbol{\pi}_v = (K-1)\boldsymbol{\pi}_B - \boldsymbol{\pi}_{\text{rest}}$, where K is the total number of paths in G. Hence the set of inequalities in Eq. 1 will imply maximizing $\boldsymbol{\alpha}^T \Delta\boldsymbol{\pi}$.

Problem 2.3 (**Formal Graph based RaTSS.**). ***Given, X**, C, a segment graph G on **X**, so that its path P_B corresponds to C, and a Function F(.) to represent a node s_{ij} on G in $m \times f$ feature matrix.*

***Find** $\boldsymbol{\alpha}$, such that,* $\arg\max_{\boldsymbol{\alpha}} \boldsymbol{\alpha}^T (\Delta\boldsymbol{\pi}) - \lambda_1 ||\boldsymbol{\alpha}||_1$ *subject to* $\boldsymbol{\alpha} \neq 0, ||\boldsymbol{\alpha}||_2^2 = 1$ *satisfies* $\text{Quality}(P_B, \alpha) > \text{Quality}(p_v, \alpha), \forall p_v \in P_{\text{rest}}$. *Generate rationalization $\mathbf{r}_j$, for each edge e_{ijk} in P_B, so $\mathbf{r}_j = |\alpha \odot \mathbf{w}_{ijk}|$.*

We want $\boldsymbol{\alpha}$ to assign a higher weight for the time-series with high value in $\Delta\boldsymbol{\pi}$ (each value $\boldsymbol{\alpha}_u$ in $\boldsymbol{\alpha}$ represents a weight for time-series x_u). The term $\lambda_1 ||\boldsymbol{\alpha}||_1$ is to encourage sparsity for simple explanations. The constraint $||\boldsymbol{\alpha}||_2^2 = 1$ is to ensure that $\boldsymbol{\alpha}_u$s are bounded and comparable. Fig. 2 shows an overview of RaTSS. Table 1 shows an overview of the notations.

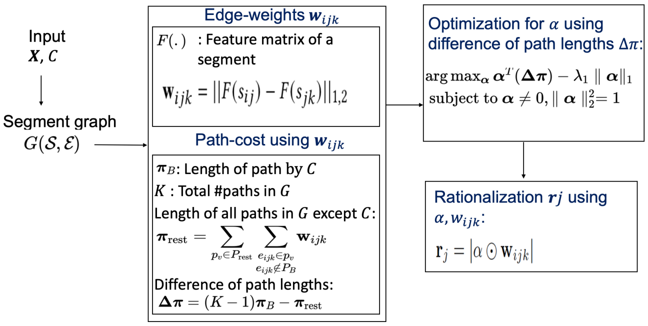

Input
$\boldsymbol{X}, C$
Segment graph
$G(\mathcal{S}, \mathcal{E})$
Edge-weights $\boldsymbol{w}_{ijk}$
$F(.)$: Feature matrix of a segment
$\mathrm{w}_{ijk} = ||F(s_{ij}) - F(s_{jk})||_{1,2}$
Path-cost using $\boldsymbol{w}_{ijk}$
$\boldsymbol{\pi}_B$: Length of path by C
K : Total #paths in G
Length of all paths in G except C:
$\boldsymbol{\pi}_{\text{rest}} = \sum_{p_v \in P_{\text{rest}}} \sum_{\substack{e_{ijk} \in p_v \\ e_{ijk} \notin P_B}} \mathbf{w}_{ijk}$
Difference of path lengths:
$\boldsymbol{\Delta\pi} = (K-1)\boldsymbol{\pi}_B - \boldsymbol{\pi}_{\text{rest}}$
Optimization for α using difference of path lengths $\Delta\pi$:
$\arg\max_{\boldsymbol{\alpha}} \boldsymbol{\alpha}^T(\boldsymbol{\Delta\pi}) - \lambda_1 \parallel \boldsymbol{\alpha} \|_1$
subject to $\boldsymbol{\alpha} \neq 0, \parallel \boldsymbol{\alpha} \|_2^2 = 1$
Rationalization $\boldsymbol{rj}$ using α, w_{ijk}:
$\mathrm{r}_j = |\alpha \odot \mathrm{w}_{ijk}|$

Figure 2: Overview of RaTSS

Our Algorithm. We design an efficient algorithm Find-RaTSS for Problem 2.3 with the following steps.

(A) Computing $\boldsymbol{\Delta\pi}$. We follow three steps to compute $\boldsymbol{\Delta\pi}$ from G: (i) Computing the length of P_B, i.e. $\boldsymbol{\pi}_B$, which is trivial. (ii) Calculating the total length of all the possible paths, except $\boldsymbol{\pi}_B$ in G, i.e., $\boldsymbol{\pi}_{\text{rest}}$. (iii) Calculating the total number of all possible paths in G, i.e., K. Our main challenge is to efficiently compute (ii) and (iii) since the number of paths in G is exponential. For computing $\boldsymbol{\pi}_{\text{rest}}$, we design an efficient technique to calculate the number of paths p_{ijk} passing through an edge e_{ijk} in constant time, by exploiting the properties of G. We find p_{ijk} using Lemma 2.1 (proof in the supplementary).

Lemma 2.1 (Number of paths through an edge e_{ijk} is p_{ijk}). *Given, i and k be the begin and end timestamp of the adjacent segments s_{ij} and s_{jk} of e_{ijk}. For total timestamp T, the number of paths through e_{ijk} is, $p_{ijk} = 1$, for $i = 1, k = T$; $p_{ijk} = 2^{(i-2)}$ for $k = T$; $p_{ijk} = 2^{(T-k-1)}$ for $i = 1$, and $p_{ijk} = 2^{(i-2)+(T-k-1)}$ for other cases.*

Using Lemma 2.1, we further compute the total number of all possible paths in G, K, intuitively, by setting $i = 1$, and $k = 2 \dots T-1$, i.e., $K = 1 + \sum_{\substack{k=2 \\ i=1}}^{T-1} p_{ijk} = 2^{T-2}$.

(B) Optimizing $\mathbf{r}_j$. We solve Problem 2.3 with gradient descent learning using Lagrange multiplier λ_2 for the constraint and the gradient $-\boldsymbol{\Delta\pi} + \lambda_1 sign(\boldsymbol{\alpha}) + \lambda_2 \boldsymbol{\alpha}$. To select the hyper-parameters, we adopt gridsearch and choose $\boldsymbol{\alpha}$ such that $||\boldsymbol{\alpha}||_2^2$ is closer to 1. We select the top k time-series (larger weight in $\mathbf{r}_j$), when cumulative fraction of the total rationalization weight >= 0.95. Algorithm 1 shows a detail pseudo-code of Find-RaTSS which is linear in the number of time-series m and cubic in T (detail on complexity is in the supplementary). Although in practice, we were able to run the parallel version in quadratic time.

Implementation Details of Find-RaTSS: For handling large data for Algorithm 1, we adopt several techniques.

(i) `Parallelization`: For efficient and fast computation of $\boldsymbol{\pi}_{\text{rest}}$, we parallelize Algorithm 1. We divide the task of calculating number of paths for edges among a set of processors n. Also we use a shared memory from Python Multiprocessing library to compute $\boldsymbol{\pi}_{\text{rest}}$.

(ii) `Floating point precision`: To efficiently store large value of p_{ijk} and K (for $T > 900$), we rearrange Eq. of Problem 2.3 as $\frac{\boldsymbol{\Delta\pi}}{K} = \boldsymbol{\pi}_B - \frac{\boldsymbol{\pi}_{\text{rest}}}{K}$. And we ignore $\frac{\boldsymbol{\pi}_{\text{rest}}}{K}$ for a very small value.

(iii) `Normalization of` $\boldsymbol{\Delta\pi}$: For efficient optimization and better $\boldsymbol{\alpha}$ convergence, we normalize on $\frac{\boldsymbol{\Delta\pi}}{K}$ by their max value and rearrange as $\boldsymbol{\Delta\pi} = \frac{\boldsymbol{\Delta\pi}}{KM}$, $M = \max(\frac{\boldsymbol{\Delta\pi}}{K})$.

```
Input: X: a set of time-series, $C$: a segmentation
Result: $\mathbf{r}_j = \{r^1, ..., r^m\}$, rationalization weight of X for
        every $c_j$ in $C$
Consider $F$: a function for characterizing time-series. Given
  a segment, it returns a feature matrix $m \times f$
1. Construct nodes in segment Graph $G$
2. Construct edges in $G$
3. foreach $e_{ijk} \in \mathcal{E}$ do
    Calculate total number of paths $p_{ijk}$ in $e_{ijk}$ using
      Lemma 2.1
    Calculate edge weight $\mathbf{w}_{ijk} = ||F(s_{ij}) - F(s_{jk})||_{1,2}$
    Edge cost: $\boldsymbol{\pi}_{\text{rest}}^{ijk} = p_{ijk}\mathbf{w}_{ijk}$
end
4. $\boldsymbol{\pi}_{\text{rest}} = \sum_{e_{ijk} \in \mathcal{E}} \boldsymbol{\pi}_{\text{rest}}^{ijk}$
5. Compute $\boldsymbol{\pi}_B$
6. $K = 2^{T-2}$
7. Solve $\boldsymbol{\alpha}$ using Problem 2.3, hyper-parameters $\lambda_1, \lambda_2$
8. foreach $c_j \in C$ do
    Compute $\mathbf{r}_j$ using Problem 2.3
end
```

Algorithm 1: Algorithm Find-RaTSS.

3 EXPERIMENTS

We implement RaTSS in Python and Matlab[1]. Our experiments were conducted on a 4 Xeon E7-4850 CPU with 512 GB of 1066Mhz main memory. We collect both general data and domain specific data, to quantitatively and qualitatively evaluate the performance of Find-RaTSS. For each data, we use the segmentation (C) which provides ground-truth (general data) or meaningful events (domain specific data). The description of the datasets and the segmentation algorithm used for each dataset are in the supplementary.

Table 2: F1-scores of Find-RaTSS and baselines on the datasets with ground-truth

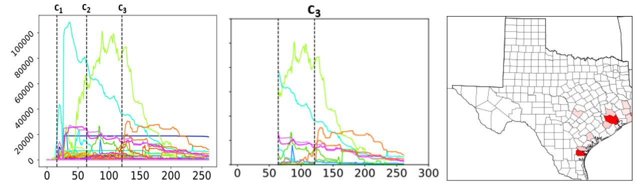
Dataset	Time stamps	Time series	Find-RaTSS	Feature	Magnitude	Fore cast
Gaussian	350	8	**1.0**	0.27	0.42	0.17
Insect [11]	5000	4	**0.83**	0.33	**0.83**	0.5
Chicken Dance [18]	322	4	**0.86**	0.81	0.71	0.5
Great Barbet [11]	2200	2	**1.0**	**1.0**	0.5	0.5
Sudden Cardiac [11]	7000	2	**1.0**	0.67	0.67	0

[1] https://github.com/AdityaLab/RaTSS

Baselines: We compare Find-RaTSS against plausible approaches, to find out the effect without principally considering any optimization. (i) *Feature-based*: calculate the change across every c_j using basic features, i.e., w_{ijk} and select x_u which have high w^u_{ijk}. (ii) *Magnitute-based*: calculate change of time-series values across c_j for a window (5% of the timestamp) and pick the time-series that are easily figured out by visualizing **X**. (iii) *Forecast-based*: For every time-series x_u, we train an LSTM forecast model based the segment before c_j and test the model for the segment after c_j. We select the time-series whose forecasting error is high. Since high forecasting error denote high measure of unpredictability on the time-series magnitude after the cutpoint. In other words change of time-series is high across the cutpoint.

Quantitative evaluation. We compare Find-RaTSS with the baselines on the data with ground-truth (GT). For rationalizations, we select top k same as the number of GT in the segmentation. Find-RaTSS consistently outperforms all the baselines (upto 41%) (see F1-scores in Table 2, high score yields better result).

Case-Studies in domain-specific data. We already explained how our rationalizations are actionable for Hurricane Irma (See Fig. 1 in Sec. 1 for detail). Next, we show culprits by Find-RaTSS in other disasters and public health data.

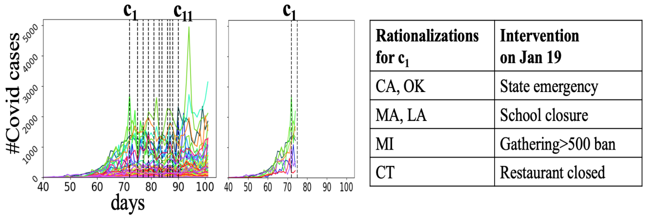


(a) (b) (c)

Figure 3: (a) 2017 Hurricane Harvey time-series. (b) A snippet of our rationalizations r_j for cutpoint c_3 and (c) Heatmap plot of r_j for c_3. Find-RaTSS finds non-obvious rationalizations separate from Hurricane trajectory (see detail in the text).

Hurricane *Harvey*: The segmentation of hurricane Harvey is given during hurricane landfall (around Aug. 26), change of hurricane trajectory (around Aug. 28), and end time of the hurricane (around Aug. 30). Fig. 3(b) shows our rationalizations for cutpoint c_3. Note that this is the cutpoint when the hurricane is ending. Along with the decrease of power failures of other counties, Find-RaTSS correctly highlights the sudden increase of power failures of Orange, Jefferson, Hardin (Fig. 3(c) South-east corner). However, the main reason for this increase is, rising water of the Neches river, which caused cities to lose service from major pump stations [6]. Note that, finding these non-obvious counties by visualizing the time-series (Fig. 3(a)) across c_3 when power failures of all other counties are decreasing is hard. These culprits can help DEs prioritize resource allocation for quick recovery.

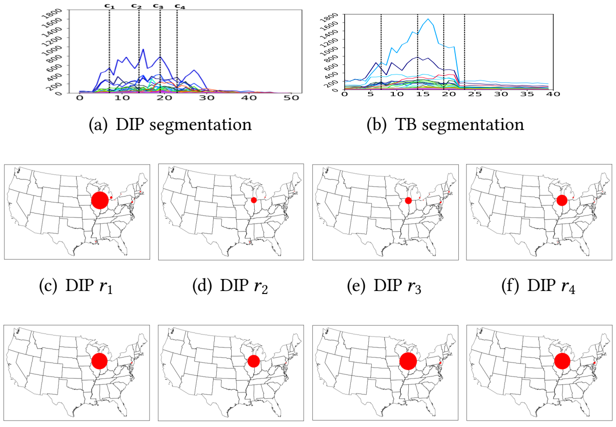
COVID-19 Interventions: For the current COVID-19 pandemic, our goal is to extract which states had interventions (like school closures, etc.) using Find-RaTSS and the disease trajectories. We collect (https://covidvis.github.io/) daily COVID-19 incidences from Jan 2020-May 2020 and consider cutpoints as the state emergencies after 2 weeks (mean incubation period of COVID-19 is 2-14 days). For c_1 (Fig. 4(b)), Find-RaTSS infers all the states which had some intervention around 2 weeks back (Jan 19). Fig. 4(c) shows an example of different interventions happened for the rationalizations across c_1 before 2 weeks. Overall, across all cutpoints, Find-RaTSS infers 87% states with interventions >= 2 weeks. For most rationalized states (60%), the interventions happened exactly 2 weeks back. This is very useful as in real-time it is very hard to have a complete knowledge of interventions, e.g., indeed, there is no centralized database of such acts. Hence public health DEs need to use indirect methods (knowing which interventions are in place is crucial for modeling the disease spread) [8, 13, 21]. Find-RaTSS can potentially give the DEs direct attention to such states.



Rationalizations for c_1	Intervention on Jan 19
CA, OK	State emergency
MA, LA	School closure
MI	Gathering>500 ban
CT	Restaurant closed

(a) (b) (c)

Figure 4: COVID-19 pandemic: (a) Time-series and segmentation (interventions). (b) Majority of the states for the first cutpoint c_1 inferred by Find-RaTSS had interventions in the past 2 weeks. (c) Types of interventions for c_1 that happened before 2 weeks.



(a) DIP segmentation (b) TB segmentation

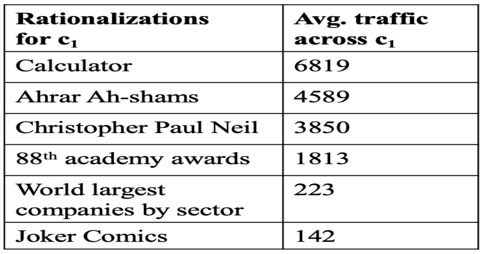
(c) DIP r_1 (d) DIP r_2 (e) DIP r_3 (f) DIP r_4

(g) TB r_1 (h) TB r_2 (i) TB r_3 (j) TB r_4

Figure 5: Find-RaTSS finds rationalizations in Diptheria and TB. Fig.(c)-(f) are our rationalizations (larger size by higher weight in r$_j$) in US map found by Find-RaTSS for the second and third cutpoint. Similar results for TB are also shown in Fig.(g)-(j).

Diptheria (DIP): For Diphtheria and TB segmentation, we consider the time period from 1900 to 1950. This segmentation includes some historical significant events, i.e., campaign (c_1), outbreak (c_3), and invention of vaccination (c_4)[2]. Across the third cutpoint (outbreak), Find-RaTSS captures three major cities Chicago, Philadelphia, and Boston, where Chicago has the high mortality rates[4]The other cities have low but sudden increase of Diptheria cases across c_3 due to epidemic and then reduce drastically before c_4 with the discovery of vaccine[5]. Further, if we compare rationalization weights from Fig 5(c)-5(f) with Fig 5(g)-5(j) we observe,r_j^u weights of the affected cities are positively correlated for both Diptheria and TB across the cutpoints. The same report (mentioned above) suggests this association is mainly due to presence of an iron-repressor gene. Clearly, by providing these insights, rationalizations can help DE understand the correlation between the diseases and design vaccination policies[24, 25].

Web Traffic of Wikipedia: We provide a case-study showing that our algorithm can successfully identify culprits also in a general dataset even in a high dimensional time-series.



Rationalizations for c_1	Avg. traffic across c_1
Calculator	6819
Ahrar Ah-shams	4589
Christopher Paul Neil	3850
88th academy awards	1813
World largest companies by sector	223
Joker Comics	142

Figure 6: A snippet of the name of the articles found by Find-RaTSS across the first cutpoint from *Wikipedia* data.

To understand the importance of content and improve advertisement strategies, we consider the web traffic of 3000 Wikipedia articles from 2015 July-2017 October. The segmentation mark a different season of a year (around the end of 2015, middle of 2016, and beginning of 2017). We find overall, the majority of articles/rationalizations (64%) by Find-RaTSS are eventful. Fig. 6 shows an example of the rationalizations by Find-RaTSS across the first cutpoint c_1. We observe Find-RaTSS considers some low-traffic articles, e.g., 'World's largest companies by sector' within top 10 along with the high traffic ones. The reason is, 'Forbes Global 2000' for worlds largest companies was published around June 2016 after c_1[3]. This is not easy to find such low traffic easily by visualizing the time-series or using any other baselines from Table. 2 (as buried with other high traffic articles).

4 CONCLUSION

In this paper, we introduce a novel problem Rationalizing time-series segmentation in terms of constituent time-series (RaTSS), to identify culprits for urban domain experts in a set of events found by time-series segmentation algorithms. We propose an algorithm Find-RaTSS to solve RaTSS in terms of a novel graph optimization problem using a segment graph data structure. Find-RaTSS successfully finds culprits in several domains such as emergency management and public health. In addition we compare its performance with non-trivial baselines using synthetic and real-life general datasets with inferred ground-truth. As future work, we plan to explore other formulations for rationalizations like the average longest path instead of the longest path on the segment graph. We also intend to explore more complex culprits (like scoring *groups* of time-series instead of individual ones as we do) which may be more suitable for some other applications.

ACKNOWLEDGMENTS

This paper is based on work partially supported by the NSF (Expeditions CCF-1918770, CAREER IIS-2028586, RAPID IIS-2027862, Medium IIS-1955883, NRT DGE-1545362), and ORNL.

REFERENCES

[1] [n. d.]. National, Regional, and State Level Outpatient Illness and Viral Surveillance. https://gis.cdc.gov/grasp/fluview/fluportaldashboard.html.

[2] [n. d.]. Web Traffic Time Series Forecasting. https://www.kaggle.com/c/web-traffic-time-series-forecasting/data.

[3] 2017. Georgia Power Working to Restore Service to 161,000 in DeKalb. https://www.dekalbcountyga.gov/news/georgia-power-working-restore-service-161000-dekalb.

[4] 2017. Project Tycho. https://www.tycho.pitt.edu/featured-works/featured-work/25/.

[5] 2019. Death in Chicago. http://digitalchicagohistory.org/exhibits/show/death-in-chicago/death-in-chicago-bibliography#diphtheria.

[6] 2019. Flooding Hits Texas Towns Devastated by Harvey. https://www.nytimes.com/2019/09/19/us/houston-beaumont-flooding-imelda.html.

[7] Alan M Barker, Eva B Freer, Olufemi A Omitaomu, Steven J Fernandez, Supriya Chinthavali, and Jeffrey B Kodysh. 2013. Automating natural disaster impact analysis: An open resource to visually estimate a hurricane's impact on the electric grid. In *2013 Proceedings of IEEE Southeastcon*. IEEE, 1–3.

[8] CDC. 2020. Coronavirus Disease 2019 (COVID-19). https://www.cdc.gov/coronavirus/2019-ncov/cases-updates/hospitalizations-forecasts.html

[9] Liangzhe Chen, Sorour E Amiri, and B Aditya Prakash. 2018. Automatic Segmentation of Data Sequences. AAAI.

[10] Eagle. 2012. Eagle-I. https://eagle-i.doe.gov/.

[11] Shaghayegh Gharghabi, Yifei Ding, Chin-Chia Michael Yeh, Kaveh Kamgar, Liudmila Ulanova, and Eamonn Keogh. 2017. Matrix Profile VIII: Domain Agnostic Online Semantic Segmentation at Superhuman Performance Levels. In *ICDM*. IEEE, 117–126.

[12] David Hallac, Sagar Vare, Stephen Boyd, and Jure Leskovec. 2017. Toeplitz inverse covariance-based clustering of multivariate time series data. In *KDD*. ACM, 215–223.

[13] David M. Hartley and Eli N. Perencevich. 2020. Public Health Interventions for COVID-19: Emerging Evidence and Implications for an Evolving Public Health Crisis. *JAMA* 323 (May 2020), 1908–1909. https://jamanetwork.com/journals/jama/fullarticle/2764656

[14] Eamonn Keogh, Selina Chu, David Hart, and Michael Pazzani. 2004. Segmenting time series: A survey and novel approach. In *Data mining in time series databases*. World Scientific, 1–21.

[15] Shengjie Lai, Nick W Ruktanonchai, Liangcai Zhou, Olivia Prosper, Wei Luo, Jessica R Floyd, Amy Wesolowski, Mauricio Santillana, Chi Zhang, Xiangjun Du, et al. 2020. Effect of non-pharmaceutical interventions to contain COVID-19 in China. (2020).

[16] Wei-Han Lee, Jorge Ortiz, Bongjun Ko, and Ruby Lee. 2018. Time series segmentation through automatic feature learning. *arXiv preprint arXiv:1801.05394* (2018).

[17] Lei Li, James McCann, Nancy S Pollard, and Christos Faloutsos. 2009. Dynammo: Mining and summarization of coevolving sequences with missing values. In *KDD*. ACM, 507–516.

[18] Yasuko Matsubara, Yasushi Sakurai, and Christos Faloutsos. 2014. Autoplait: Automatic mining of co-evolving time sequences. In *SIGMOD*. ACM, 193–204.

[19] Robert Moss, Elham Naghizade, Martin Tomko, and Nicholas Geard. 2019. What can urban mobility data reveal about the spatial distribution of infection in a single city? *BMC public health* 19, 1 (2019), 1–16.

[20] Nikhil Muralidhar, Anika Tabassum, Liangzhe Chen, Supriya Chinthavali, Naren Ramakrishnan, and B Aditya Prakash. 2020. Cut-n-Reveal: Time Series Segmentations with Explanations. *ACM Transactions on Intelligent Systems and Technology*

[2] https://timelines.issarice.com/wiki/Timeline_of_diphtheria

[3] https://www.forbes.com/sites/forbespr/2016/05/25/forbes-14th-annual-global-2000-the-worlds-biggest-public-companies-2/#21ca87643c44

(TIST) 11, 5 (2020), 1–26.

[21] An Pan, Li Liu, Chaolong Wang, Huan Guo, Xingjie Hao, Qi Wang, Jiao Huang, Na He, Hongjie Yu, Xihong Lin, et al. 2020. Association of public health interventions with the epidemiology of the COVID-19 outbreak in Wuhan, China. *Jama* 323, 19 (2020), 1915–1923.

[22] Nick Warren Ruktanonchai, JR Floyd, Shengjie Lai, Corrine Warren Ruktanonchai, Adam Sadilek, Pedro Rente-Lourenco, Xue Ben, Alessandra Carioli, Joshua Gwinn, JE Steele, et al. 2020. Assessing the impact of coordinated COVID-19 exit strategies across Europe. *Science* (2020).

[23] Allou Samé and Gérard Govaert. 2012. Online Time Series Segmentation Using Temporal Mixture Models and Bayesian Model Selection. *ICMLA* 1 (2012), 602–605.

[24] Swapna Thorve, Mandy L Wilson, Bryan L Lewis, Samarth Swarup, Anil Kumar S Vullikanti, and Madhav V Marathe. 2018. EpiViewer: an epidemiological application for exploring time series data. *BMC bioinformatics* 19, 1 (2018), 449.

[25] Srinivasan Venkatramanan, Jiangzhuo Chen, Sandeep Gupta, Bryan Lewis, Madhav Marathe, Henning Mortveit, and Anil Vullikanti. 2017. Spatio-temporal optimization of seasonal vaccination using a metapopulation model of influenza. In *2017 IEEE International Conference on Healthcare Informatics (ICHI)*. IEEE, 134–143.

[26] Scott L Zeger, Rafael Irizarry, and Roger D Peng. 2006. On time series analysis of public health and biomedical data. *Annu. Rev. Public Health* 27 (2006), 57–79.

[27] Yu Zheng, Licia Capra, Ouri Wolfson, and Hai Yang. 2014. Urban computing: concepts, methodologies, and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)* 5, 3 (2014), 1–55.

[28] Fan Zuo, Jingxing Wang, Jingqin Gao, Kaan Ozbay, Xuegang Jeff Ban, Yubin Shen, Hong Yang, and Shri Iyer. 2020. An Interactive Data Visualization and Analytics Tool to Evaluate Mobility and Sociability Trends During COVID-19. *arXiv preprint arXiv:2006.14882* (2020).

A SUPPLEMENTARY

A.1 Additional Detail on Algorithm Find-RaTSS

A.1.1 *Proof of Lemma 2.1.*

Proof. Following is the proof for different i, j, and k.

- **Case** $i = 1, k = T$: Trivial. Since, there is only on segment from 1 to T and only one path.
- **Case** $i = 1, k < T$: (By induction). Suppose, $T = k + 1$, then the number of paths is $2^{k+1-k-1} = 2^0 = 1$. This is trivial, because from s_{jk} there is only one possible edge $s_{jk} - s_{k(k+1)}$ and thus there is only one possible path.
  Suppose, with $T = k + q$ where $q > k$, the number of paths is 2^{q-1}. We need to show, with $T = k + q + 1$, the number of paths become 2^q. Since $q + 1 > q$, all the paths that can be possible to reach from s_{jk} to $s_{j(k+q)}$ can also be possible to reach from s_{jk} to $s_{j(k+q+1)}$ using the edge $s_{j(k+q)} - s_{j(k+q+1)}$. That is, 2^{q-1} paths possible from s_{jk} to $s_{j(k+q+1)}$ using the edge $s_{j(k+q)} - s_{j(k+q+1)}$.
  Again, by the edge construction property of G, a path ends when end timestamp of a node is $T = k + q + 1$. If we remove the timestamp $k + q$ and consider only $k, k + 1, \ldots, k + q - 1, k + q + 1$, then the number of possible paths become 2^{q-1} to reach $k + q + 1$ (by induction). Because, all edges that used nodes ended with $k + q$ now can use nodes ended with $k + q + 1$.
  Thus, the total number of possible paths with and without using the edge $s_{j(k+q)} - s_{j(k+q+1)}$ becomes $2^{q-1} + 2^{q-1} = 2^{q-1+1} = 2^q$ (proved).
- **Case** $i > 1, k = T$**:** Following the above if we consider $T = i$ and the $k = 1$. The number of paths is $2^{T-k-1} = 2^{i-1-1} = 2^{i-2}$.
- **Case** $otherwise$**:** Hence, if there are 2^{i-2} paths possible to reach a node s_{ij} and 2^{T-k-1} possible paths to reach timestamp T from a node s_{jk}. The total number of paths possible through the edge $s_{ij} - s_{jk}$ or e_{ijk} is $p_{ijk} = 2^{(i-2)+(T-k-1)}$.

□

A.1.2 *Total number of paths K.*

Corollary A.1 (**Total number of possible paths in segment graph** G **is** K). *Given total timestamp is T, and number of paths through an edge e_{ijk} is p_{ijk}. If we consider all the edges in edgeset G as $\mathcal{E}$, the total number of possible paths in G is- $K = 1 + \sum_{k=2,\, i=1}^{T-1} p_{ijk} = 2^{T-2}$*

Proof. According to the path property of G, every path starts from $i = 1$ and ends at timestamp T. For $i = 1, k = T$ there is only one path possible (the whole segmentation). For $1 < j < T - 1$, the first node of all other paths in G has to start from some s_{1j}. Consider $j = k$, using Lemma 2.1 we find p_{1kk}. Thus, $K = 1 + \sum_{k=2}^{T-1} p_{1kk} = 1 + 2^{T-2} - 1 = 2^{T-2}$. □

A.1.3 *Complexity of Find-RaTSS.*

Lemma A.2 (**Time and Space Complexity**). *The worst-case time complexity of our algorithm is as follows:*

- *Case serial: $O(T^3mf) + O(I) + O(Cmf)$, I= number of iterations in the gradient descent phase.*
- *Case parallel: $O(\frac{T^3}{n}mf) + O(I) + O(Cmf)$, n = number of processors.*

The space complexity of Find-RaTSS is $O(m + mn)$.

Proof. We discuss separate proof for time and space complexity.
Time complexity: (i) *Case serial:* The first term is to calculate π_{rest} for each e_{ijk}. Total e_{ijk} in G is $O(T^3)$. The second term is to solve α using Gradient Descent learning, and the third term is to solve $\mathbf{r}_j$ for every cutpoint c_j.

(ii) *Case Parallel:* Parallelization can be applied in Find-RaTSS for π_{rest} computation, then the total time complexity will be distributed among n processors and hence this is $O(\frac{T^3}{n}mf)$. However, time complexity of gradient descent and $\mathbf{r}_j$ computation is similar as in serial cases.
Space complexity: (i) Case serial: The first term $O(m)$ is to store π_{rest} and π_B. Since we need $O(m)$ π_{rest} and $O(m)$ π_B for m time-series. Hence $O(m) + O(m) = O(2m) \approx O(m)$.

(ii) Case parallel: To store final computation of π_{rest} and π_B it takes $O(m)$. Whereas the second term $O(mn)$ is to store temporary $\boldsymbol{\pi}_{\text{rest}}$ computation for each processor among n processors, $O(mn)$ is the number of possible edges in a segment graph G. Hence temporary space for n processor and final computation of $\boldsymbol{\pi}_B$ and $\boldsymbol{\pi}_{\text{rest}}$ is $O(mn) + O(m) = O(m + mn)$. □

A.2 Additional Detail on Experiments

Table 3: Datasets used.

SI #	Dataset	Time stamps	Time series	Cut points	Segmentation
(1)	*Gaussian*	350	8	3	[12]
(2)	*Insect*	5000	4	2	[11]
(3)	*Chicken Dance*	322	4	7	[18]
(4)	*Great Barbet*	2200	2	2	[11]
(5)	*Sudden Cardiac*	7000	2	3	[11]
(6)	*Wikipedia*	803	3000	3	[12]
(7)	*Hurricane Harvey/Irma*	264/169	250/271	3/4	[20]
(8)	*COVID-19*	53	104	11	date of emergency[4]
(9)	*Diptheria/TB*	52/41	90/60	4/4	[12]
(10)	*Flu*	52	50	2	[12]

Description of General Datasets: We use a variety of datasets (both synthetic and real) where we infer the ground-truth culprits (Table 3 SI 1-5). Additionally we use a case-study data of Wikipedia articles (Table 3 SI 6) to show that our algorithm can also capture meaningful culprits even in a general dataset.

(1) *Gaussian*: We generate a synthetic data, where each time-series is a univariate Gaussian. We select three cutpoints and

change the parameter of the Gaussian for specific $2-3$ time-series at those cutpoints. Ground-truth (GT) rationalizations are the time-series whose Gaussian parameters change at a cutpoint.

(2) *Insect* [11]: EPG recording of insect vector feeding, where each time-series represents an insect. Each cutpoint is an event when feeding state of the insect changes and GT is the insect whose feeding state changes at that event.

(3) *ChickenDance* [18]: Motion capture sequences of a set of sensors (left-right hands/legs) in a chicken dance originally collected by CMU[5].An events occurs, when is a change of dance state, e.g., wings, tail feather, etc. GT are the set of sensors which have high change of motion while changing a dance state.

(4) *Great Barbet* [11]: Voice recordings of same species Barbet birds in MFCC format. Each recording is a mixture of two different birds with approximately half-a-minute snippet of their song. The cutpoints are set of events when snippet of one bird ends and other starts. GT are the set of birds, whose call ends or starts at an event.

(5) *Sudden Cardiac* [11]: ECG channel reading of hearts of patients. Events occur when heart state changes, e.g., normal heart to sudden heart failure or contraction. GT are the patient whose heart activity changes at an event.

(6) *Wikipedia*: Web traffic count of various Wikipedia articles collected daily from the July 2015- October 2017 [2].

Description of Domain specific Datasets: Next, we describe our domain specific datasets from urban analytics. Here as there is no ground truth, we discuss the qualitative performance our algorithm.

(7) *Hurricane Outage:* Oak Ridge National Laboratory (ORNL) has developed several grid situational awarenss tool such as *VERDE*, *EAGLE-I* mainly for the stakeholders in emergency management [10]. The National Outage Map using *EAGLE-I* collects power outage distribution of all the customers from utility websites every 15 minutes. Each time-series is the power outage distribution of different counties for Hurricanes *Irma* and *Harvey* in the hurricane-affected areas. Rationalizations on such data can help DE with retrospective analysis for emergency management planning.

(8) COVID-19 : COVID cases of different states are collected daily from Jan-May by New York Times[6]

(9) *Diptheria* and *TB*: Diptheria and Tuberculosis (TB) cases collected bi-weekly from the year 1900-2014 by Project Tycho[7]. Time-series are different cities of US representing Diptheria (TB) cases over time period (years). We run Dynammo [17] to replace the missing values of original data.

(10) *FLU*: Center for Disease Control (CDC) weekly influenza-like-illness count (wILI), where *FLU* represents all US state during the year 2017-18 of all HHS regions of the US[8]. [1].

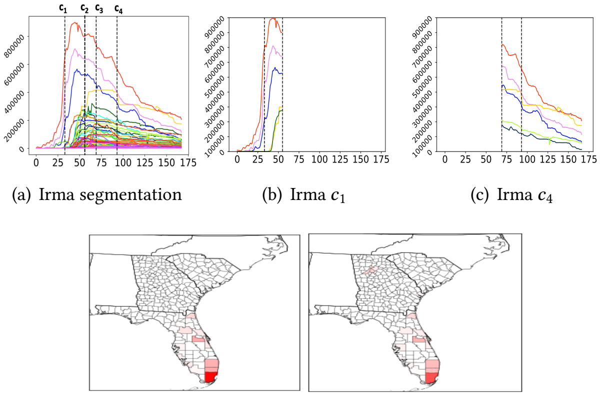
Rationalizations on *Diptheria* and *FLU* data can help DE planning for vaccine allocation and co-occurrence analysis during an epidemic.

Table 4: Feature Ablation test in terms of F1-score for Ground-truth datasets.f_1: mean, f_2: variance, f_3: min, f_4: max

Dataset	Features removed			
	f_1	f_2	f_3	f_4
Gaussian	0.52	0.44	0.49	0.64
ChickenDance	0.73	0.86	0.401	0.643
Great Barbet	0.5	0.5	0	1.0

A.2.1 Additional experiments: Feature ablation test. For showing robustness of the statistical features we use for Find-RaTSS, Table. 4 shows how rationalization affected on the *Gaussian*, *ChickenDance* and *Great Barbet* by removing each feature used in RaTSS.

A.3 Additional Results for Case-Studies: Hurricane Irma



(a) Irma segmentation (b) Irma c_1 (c) Irma c_4

(d) Rationalization for c_1 (e) Rationalization for c_4

Figure 7: The rationalizations $\mathbf{r}_j$ found by Find-RaTSS for c_1 and c_4 in a heat map (higher r_j^u corresponds to brighter red).

Find-RaTSS correctly captures Miami, Broward, and counties at south west during hurricane landfall (first cutpoint c_1) when failures starts to increase. From Fig. 7(c), across c_4 (when hurricane is ending), Find-RaTSS finds the same culprit counties, however this time due to the significant decrease of their failures. These Insights can help DE understand how long these counties take to restore their power failures.

A.4 Additional Results for Case-studies: *FLU*

Fig. 8 shows our rationalizations (HHS regions) where influenza cases co-occur during the start and end of influenza season 2017-18. These rationalizations can help DEs provide actionable insights on decision-making regarding vaccine allocation.

[5] http://mocap.cs.cmu.edu
[6] https://github.com/nytimes/covid-19-data
[7] https://www.tycho.pitt.edu/
[8] https://gis.cdc.gov/grasp/fluview/fluportaldashboard.html

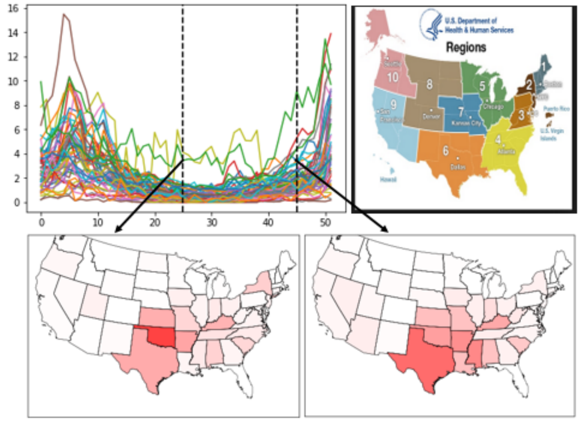


Figure 8: CDC weighted influenza incidence (wILI) on all the HHS regions in US during the season 2017-18. The segmentation obtained by TICC [12] try to give cutpoints when there starts high epidemic, and during low epidemic. Find-RaTSS identifies the HHS regions 4, 6, 7 which undergo similar influenza season during year 2017-18.