ABSTRACT
Modern world has witnessed a dramatic increase in our ability to collect, transmit and distribute real-time monitoring and surveillance data from large-scale information systems and cyber-physical systems. Detecting system anomalies thus attracts significant amount of interest in many fields such as security, fault management, and industrial optimization. Recently, invariant network has shown to be a powerful way in characterizing complex system behaviours. In the invariant network, a node represents a system component and an edge indicates a stable, significant interaction between two components. Structures and evolutions of the invariance network, in particular the vanishing correlations, can shed important light on locating causal anomalies and performing diagnosis. However, existing approaches to detect causal anomalies with the invariant network often use the percentage of vanishing correlations to rank possible causal components, which have several limitations: 1) fault propagation in the network is ignored; 2) the root casual anomalies may not always be the nodes with a high percentage of vanishing correlations; 3) temporal patterns of vanishing correlations are not exploited for robust detection. To address these limitations, in this paper we propose a network diffusion based framework to identify significant causal anomalies and rank them. Our approach can effectively model fault propagation over the entire invariant network, and can perform joint inference on both the structural and the time-evolving broken invariance patterns. As a result, it can locate high-confidence anomalies that are truly responsible for the vanishing correlations, and can compensate for unstructured measurement noise in the system. Extensive experiments on synthetic datasets, bank information system datasets, and coal plant cyber-physical system datasets demonstrate the effectiveness of our approach.

Keywords
causal anomalies ranking, label propagation, nonnegative matrix factorization

1. INTRODUCTION
With the rapid advances in networking, computers, and hardware, we are facing an explosive growth of complexity in networked applications and information services. These large-scale, often distributed, information systems usually consist of a great variety of components that work together in a highly complex and coordinated manner. One example is the Cyber-Physical System (CPS) which is typically equipped with a large number of networked sensors that keep recording the running status of the local components; another example is the large scale Information Systems such as the cloud computing facilities in Google, Yahoo! and Amazon, whose composition includes thousands of components that vary from operating systems, application softwares, servers, to storage, networking devices, etc.

A central task in running these large scale distributed systems is to automatically monitor the system status, detect anomalies, and diagnose system fault, so as to guarantee stable and high-quality services or outputs. Significant research efforts have been devoted to this topic in the literatures. For instance, Gertler et al. [9] proposed to detect anomalies by examining monitoring data of individual component with a thresholding scheme. However, it can be quite difficult to learn a universal and reliable threshold in practice, due to the dynamic and complex nature of information systems. More effective and recent approaches typically start with building system profiles, and then detect anomalies via analyzing patterns in these profiles [5, 13]. The system profile is usually extracted from historical time series data collected by monitoring different system components, such as the flow intensity of software log files, the system audit events and the network traffic statistics, and sometimes sensory measurements in physical systems.

The invariant model is a successful example [13, 14] for large-scale system management. It focuses on discovering stable, significant dependencies between pairs of system components that are monitored through time series recordings, so as to profile the system status and perform subsequent reasoning. A strong dependency between a pair of components is called invariant (correlation) relationship. By combining the invariants learned from all monitoring components, a global system dependency profile can be obtained. The significant practical value of such an invariant profile is that it provides important clues on abnormal system behaviors and in particular the source of anomalies, by checking whether existing invariants are broken. Figure 1 illustrates one example of the invariant network and two snapshots of broken invariants at time $t_1$ and $t_2$, respectively. Each
node in the figure represents the observation from a monitoring component. The green line signifies an invariant link between two components, and a red line denotes broken invariant (i.e., vanishing correlation). The network including all the broken invariants at given time point is referred to as the broken network.

Although the broken invariants provide valuable information of the system status, how to locate true, causal anomalies can still be a challenging task due to the following reasons. First, system faults are seldom isolated. Instead, starting from the root location/component, anomalous behavior will propagate to neighboring components [13], and different types of system faults can trigger diverse propagation patterns. Second, monitoring data often contains a lot of noises due to the fluctuation of complex operation environments.

Recently, several ranking algorithms were developed to diagnose the system failure based on the percentage of broken invariant edges associated with the nodes, such as the egonet based method proposed by Ge et al. [8], and the loopy belief propagation (LBP) based method proposed by Tao et al. [22]. Despite the success in practical applications, existing methods still have certain limitations. First, they do not take into account the global structure of the invariant network, neither how the root anomaly/fault propagates in such a network. Second, the ranking strategies rely heavily on the percentage of broken edges connected to a node. For example, the mRank algorithm [8] calculated the anomaly score of a given node using the ratio of broken edges within the egonet \(^1\) of the node. The LBP-based method [22] used the ratio of broken edges as the prior probability of abnormal state for each node. We argue that, the percentage of broken edges may not serve as a good evidence of the causal anomaly. This is because, although one broken edge can indicate that one (or both) of related nodes is abnormal, lack of a broken edge does not necessary indicate that related nodes are problem free. Instead, it is possible that the correlation is still there when two nodes become abnormal simultaneously [13]. Therefore the percentage of broken edges could give false evidences. For example, in Figure 1, the causal anomaly is node ①. The percentage of broken edges for node ① is 2/3, which is smaller than that of node ③ (which is equal to 1). Since there exists a clear evidence of fault propagation on node ①, an ideal algorithm should rank ① higher than ③. Third, existing methods usually consider static broken network instead of multiple broken networks at successive time points together. While we believe that, jointly analyzing temporal broken networks can help resolve ambiguity and achieve a denoising effect. This is because, the root casual anomalies usually remain unchanged within a short time period, even though the fault may keep propagating in the invariant network. As an example shown in Figure 1, it would be easier to detect the causal anomaly if we jointly consider the broken networks at two successive time points together.

To address the limitations of existing methods, we propose several network diffusion based algorithms for ranking causal anomalies. Our contributions are summarized as follows.

1. We employ the network diffusion process to model propagation of causal anomalies and use propagated anomaly scores to reconstruct the vanishing correlations. By minimizing the reconstruction error, the proposed methods simultaneously consider the whole invariant network structure and the potential fault propagation. We also provide rigid theoretical analysis on the properties of the proposed methods.

2. We further develop efficient algorithms which reduce the time complexity from \(O(n^3)\) to \(O(n^2)\), where \(n\) is the number of nodes in the invariant network. This makes it feasible to quickly localize root cause anomalies in large-scale systems.

3. We employ effective normalization strategy on the ranking scores, which can reduce the influence of extreme values or outliers without having to explicitly remove them from the data.

4. We develop a smoothing algorithm that enables users to jointly consider dynamic and time-evolving broken network, and thus obtain better ranking results.

5. We evaluate the proposed methods on both synthetic datasets and two real datasets, including the bank information system and the coal plant cyber-physical system datasets. Experimental results demonstrate the effectiveness of our methods.

2. BACKGROUND AND PROBLEM DEFINITION

In this section, we first introduce the technique of the invariant model [13] and then define our problem.

2.1 System Invariant and Vanishing Correlations

The \textit{invariant} model is used to uncover significant pairwise relations among massive set of time series. It is based on the AutoRegressive eXogenous (ARX) model [10] with time delay. Let \(x(t)\) and \(y(t)\) be a pair of time series under consideration, where \(t\) is the time index, and let \(n\) and \(m\) be the degrees of the ARX model, with a delay factor \(k\). Let \(\hat{y}(t; \theta)\) be the prediction of \(y(t)\) using the ARX model parametrized by \(\theta\), which can then be written as

\[
\hat{y}(t; \theta) = a_1 y(t - 1) + \cdots + a_n y(t - n) + b_0 x(t - k) + \cdots + b_m x(t - k - m) + d
\]

\[
= \varphi(t)^\top \theta, \tag{1}
\]

\(\text{An egonet is the induced 1-step subgraph for each node.}

\(\text{Figure 1: Invariant network and vanishing correlations (red edges).}

\(\text{a) } t_1 \quad \text{b) } t_2\)
the invariant network
where \( \theta \) is the sparse factor. The invariant network is referred to as the model training. The invariant network is referred to as the model training. The invariant network is referred to as the model training.

### 2.2 Problem Definition
Let \( G_t \) be the invariant network with \( n \) nodes. Let \( \hat{G}_t \) be the broken network for \( G_t \). We use two symmetric matrices \( A \in \mathbb{R}^{n \times n} \) and \( P \in \mathbb{R}^{n \times n} \) to denote the adjacency matrix of network \( G_t \) and \( \hat{G}_t \), respectively. These two matrices can be obtained as discussed in Section 2.1. The two matrices can be binary or continuous. For binary case of \( A \), \( 1 \) is used to denote that the correlation exists between two time series, and \( 0 \) denotes the lack of correlation; while for \( P \), \( 1 \) is used to denote that the correlation is broken (vanishing), and \( 0 \) otherwise. For the continuous case, the fitness score \( F(\theta) \) (3) and the residual \( R(t) \) (4) can be used to fill the two matrices, respectively.

Our main goal is to detect the abnormal nodes in \( G_t \) that are most responsible for causing the broken edges in \( \hat{G}_t \). In this sense, we call such nodes “causal anomalies”. Accurate detection of causal anomalous nodes will be extremely useful for examination, debugging and repair of system failures.

### 3. RANKING CAUSAL ANOMALIES
In this section, we present the algorithm of Ranking Causal Anomalies (RCA), which takes into account both the fault-propagation and fitting of broken invariants simultaneously.

#### 3.1 Fault Propagation
We consider a very practical scenario of fault propagation, namely anomalous system status can always be traced back to a set of root cause anomaly nodes, or causal anomalies, as initial seeds. As the time passes, these root cause anomalies will then propagate along the invariant network, most probably towards their neighbors via paths identified by the invariant links in \( G_t \). To explicitly model this spreading process on the network, we have employed the label propagation technique [16, 24, 26]. Suppose that the (unknown) root cause anomalies are denoted by the indicator vector \( e \), whose entries \( e_i \)'s \( (1 \leq i \leq n) \) indicate whether the \( i \)th node is the causal anomaly \( (e_i = 1) \) or not \( (e_i = 0) \). At the end of propagation, the system status is represented by the anomaly score vector \( r \), whose entries tell us how severe each node of the network has been impaired. The propagation from \( e \) to \( r \) can be modeled by the following optimization problem

\[
\min_{r \geq 0} \sum_{i,j=1}^{n} A_{ij} \left( \frac{1}{\sqrt{D_{ii}}} r_i - \frac{1}{\sqrt{D_{jj}}} r_j \right)^2 + (1 - c) \sum_{i=1}^{n} \| r_i - e_i \|_2^2,
\]

where \( D \in \mathbb{R}^{n \times n} \) is the degree matrix of \( A \), \( c \in (0, 1) \) is the regularization parameter, \( r \) is the anomaly score vector after the propagation of the initial faults in \( e \). We can re-write the above problem as

\[
\min_{r \geq 0} \sum_{i,j=1}^{n} A_{ij} \left( \frac{1}{\sqrt{D_{ii}}} r_i - \frac{1}{\sqrt{D_{jj}}} r_j \right)^2 + (1 - c) \sum_{i=1}^{n} \| r_i - e_i \|_2^2,
\]

where \( 1 \) is the identity matrix. \( A = D^{-1/2} A D^{-1/2} \) is the degree-normalized version of \( A \). Similarly we will use \( P \) as the degree-normalized \( P \) in the sequel. The first term in Eq. (5) is the smoothness constraint [26], meaning that a good ranking function should assign similar values to nearby nodes in the network. The second term is the fitting constraint, which means that the final status should be close to the initial configuration. The trade-off between these two competing constraints is controlled by a positive parameter \( c \): a small \( c \) encourages a sufficient propagation, and a big \( c \) actually suppresses the propagation. The optimal solution of problem (5) is [26]

\[
r = (1-c)(I_n - cA)^{-1}e,
\]

which establishes an explicit and closed-form solution between the initial configuration \( e \) and the final status \( r \) through fault propagation.

To encode the information of the broken network, we propose to use \( r \) to reconstruct the broken network \( \hat{G}_t \). The intuition is illustrated in Figure 2. If there exists a broken

### Table 1: Summary of notations

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>( n )</td>
<td>the number of nodes in the invariant network</td>
</tr>
<tr>
<td>( c, \lambda, \tau )</td>
<td>the parameters ( 0 &lt; c &lt; 1, \lambda &gt; 0 )</td>
</tr>
<tr>
<td>( \sigma (\cdot) )</td>
<td>the softmax function</td>
</tr>
<tr>
<td>( \theta )</td>
<td>the invariant network</td>
</tr>
<tr>
<td>( \hat{G}_t )</td>
<td>the broken network for ( \hat{G}_t )</td>
</tr>
<tr>
<td>( A(\hat{A}) \in \mathbb{R}^{n \times n} )</td>
<td>the (normalized) adjacency matrix of ( \hat{G}_t )</td>
</tr>
<tr>
<td>( P(\hat{P}) \in \mathbb{R}^{n \times n} )</td>
<td>the (normalized) adjacency matrix of ( \hat{G}_t )</td>
</tr>
<tr>
<td>( M \in \mathbb{R}^{n \times n} )</td>
<td>the logical matrix of ( \hat{G}_t )</td>
</tr>
<tr>
<td>( d(i) )</td>
<td>the degree of the ( i )th node in network ( G_t )</td>
</tr>
<tr>
<td>( D \in \mathbb{R}^{n \times n} )</td>
<td>the degree matrix: ( D = \text{diag}(d(i), ..., d(n)) )</td>
</tr>
<tr>
<td>( r \in \mathbb{R}^{n \times 1} )</td>
<td>the prorogated anomaly score vector</td>
</tr>
<tr>
<td>( e \in \mathbb{R}^{n \times 1} )</td>
<td>the ranking vector of causal anomalies</td>
</tr>
<tr>
<td>RCA</td>
<td>the basic ranking causal anomalies algorithm</td>
</tr>
<tr>
<td>RCA-SOFT</td>
<td>the RCA with softmax normalization</td>
</tr>
<tr>
<td>RCA-SOFT</td>
<td>the RCA with temporal smoothing</td>
</tr>
<tr>
<td>RCA-SOFT</td>
<td>the RCA with temporal smoothing</td>
</tr>
<tr>
<td>RCA-SOFT</td>
<td>the RCA with temporal smoothing</td>
</tr>
<tr>
<td>RCA-SOFT</td>
<td>the RCA with temporal smoothing</td>
</tr>
</tbody>
</table>

where \( \theta = [a_1, ..., a_n, b_0, ..., b_m, d] \in \mathbb{R}^{n+\theta+2} \), \( \varphi(t) = [y(t-1), ..., y(t-n), x(t-k-1), ..., x(t-k-m)] \in \mathbb{R}^{n+\theta+2} \). For a given setting of \( (n, m, k) \), the parameter \( \theta \) can be estimated with observed time points \( t = 1, ..., N \) in the training data, via least-square fitting. In real-world applications such as anomaly detection in physical systems, 0 \( \leq n, m, k \leq 2 \) is a popular choice [6, 13]. We can define the “goodness of fit” (or fitness score) of an ARX model as

\[
F(\theta) = 1 - \frac{\sum_{t=1}^{N} |y(t) - \hat{y}(t; \theta)|^2}{\sum_{t=1}^{N} |y(t) - \tilde{y}(t)|^2},
\]

where \( \tilde{y} \) is the mean of the time series \( y(t) \). A higher value of \( F(\theta) \) indicates a better fitting of the model. An invariant (correlation) is declared on a pair of time series \( x \) and \( y \) if the fitness score of the ARX model is larger than a pre-defined threshold. A network including all the invariant links is referred to as the invariant network. Construction of the invariant network is referred to as the model training. The model \( \theta \) will then be applied on the time series \( x \) and \( y \) in the testing phase to track vanishing correlations.

To track vanishing correlations, we can use the techniques developed in [6, 15]. At each time point, we compute the (normalized) residual \( R(t) \) between the measurement \( y(t) \) and its estimate \( \hat{y}(t; \theta) \) by

\[
R(t) = \frac{|y(t) - \hat{y}(t; \theta)|}{\varepsilon_{\text{max}}},
\]

where \( \varepsilon_{\text{max}} \) is the maximum training error \( \varepsilon_{\text{max}} = \max_{1 \leq t \leq N} |y(t) - \hat{y}(t; \theta)| \). If the residual exceeds a prefixed threshold, then we declare the invariant as “broken”, i.e., the correlation between the two time series vanishes. The network including all broken edges at given time point and all nodes in the invariant network is referred to as the broken network.
link in $G_b$, e.g., $\tilde{P}_{ij}$ is large, then ideally at least one of the nodes $i$ and $j$ should be abnormal, or equivalently, either $r_i$ or $r_j$ should be large. Thus, we can use the product of $r_i$ and $r_j$ to reconstruct the value of $\tilde{P}_{ij}$. In Section 5, we’ll further discuss how to normalize them to avoid extreme values. Then, the loss of reconstructing the broken link $\tilde{P}_{ij}$ can be calculated by $(r_i \cdot r_j - \tilde{P}_{ij})^2$. The reconstruction error of the whole broken network is then $\|rr^\top\circ M - \tilde{P}\|_F^2$. Here, $\circ$ is element-wise operator, and $M$ is the logical matrix of the invariant network $G_i$ (1 with edge, 0 without edge). Let $B = (1 - c)(I_n - cA)^{-1}$, by substituting $r$ we obtain the following objective function.

$$\min_{e_i \in (0,1), 1 \leq i \leq n} \|((Be_i)B^\top) \circ M - \tilde{P}\|_F^2 + \tau \|e\|_1$$  \hspace{1cm} (7)

Considering that the integer programming in problem (7) is NP-hard, we relax it by using the $\ell_1$ penalty on $e$ with parameter $\tau$ to control the number of non-zero entries in $e$ [23]. Then we reach the following objective function.

$$\min_{e \geq 0} \|((Be^\top B^\top) \circ M - \tilde{P}\|_F^2 + \tau \|e\|_1$$  \hspace{1cm} (8)

### 3.2 Learning Algorithm

In this section, we present an iterative multiplicative updating algorithm to optimize the objective function in (8). The objective function is invariant under these updates if and only if $e$ are at a stationary point [17]. The solution is presented in the following theorem, which is derived from the Karush-Kuhn-Tucker (KKT) complementarity condition [3]. Detailed theoretical analysis of the optimization procedure will be presented in the next section.

**Theorem 1.** Updating $e$ according to Eq. (9) will monotonically decrease the objective function in Eq. (8) until convergence.

$$e \leftarrow e \circ \left\{ \frac{4((Be^\top B^\top) \circ M) B e}{4((B^\top B) \circ M) B e + \tau 1_n} \right\}^{\frac{1}{2}},$$  \hspace{1cm} (9)

where $\circ$, $\|\|$ and $(\cdot)^{\frac{1}{2}}$ are element-wise operators.

Based on Theorem 1, we develop the iterative multiplicative updating algorithm for optimization and summarize it in Algorithm 1. We refer to this ranking algorithm as RCA.

<table>
<thead>
<tr>
<th>Algorithm 1: Ranking Causal Anomalies (RCA)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong> Network $G_i$ denoting the invariant network with $n$ nodes, and is represented by an adjacency matrix $A$, $c$ is the network propagation parameter, $\tau$ is the parameter to control the sparsity of $e$, $\tilde{P}$ is the normalized adjacency matrix of the broken network, $M$ is the logical matrix of $G_i$ (1 with edge, 0 without edge)</td>
</tr>
<tr>
<td><strong>Output:</strong> Ranking vector $e$</td>
</tr>
<tr>
<td>1. <strong>Begin</strong></td>
</tr>
<tr>
<td>2. for $i \leftarrow 1$ to $n$ do</td>
</tr>
<tr>
<td>3. $\quad D_{ii} \leftarrow \sum_{j=1}^{n} A_{ii}$;</td>
</tr>
<tr>
<td>4. end</td>
</tr>
<tr>
<td>5. $D \leftarrow diag(D_{11}, ..., D_{nn})$;</td>
</tr>
<tr>
<td>6. $A \leftarrow D^{-1/2}AD^{-1/2}$;</td>
</tr>
<tr>
<td>7. Initialize $e$ with random values between (0,1];</td>
</tr>
<tr>
<td>8. $B \leftarrow (1-c)(I_n - cA)^{-1}$;</td>
</tr>
<tr>
<td>9. repeat</td>
</tr>
<tr>
<td>10. $\quad$ Update $e$ by Eq. (9);</td>
</tr>
<tr>
<td>11. until <strong>convergence</strong>;</td>
</tr>
<tr>
<td><strong>End</strong></td>
</tr>
</tbody>
</table>

### 3.3 Theoretical Analysis

#### 3.3.1 Derivation

We derive the solution to problem (9) following the constrained optimization theory [3]. Since the objective function is not jointly convex, we adopt an effective multiplicative updating algorithm to find a local optimal solution. We prove Theorem 1 in the following.

We formulate the Lagrange function for optimization $L = \|((Be^\top B^\top) \circ M - \tilde{P}\|_F^2 + \tau \|e\|_1$. Obviously, $B$, $M$ and $\tilde{P}$ are symmetric matrix. Let $F = (Be^\top B^\top) \circ M$, then

$$\frac{\partial}{\partial e_m} (F - \tilde{P})^2 = 2(F_{ij} - \tilde{P}_{ij}) \frac{\partial F_{ij}}{\partial e_m}$$

$$= 4(F_{ij} - \tilde{P}_{ij})M_{ij}(B_m^\top B_j e) \text{ (by symmetry)}$$

$$= 4B_m^\top(F_{ij} - \tilde{P}_{ij})M_{ij}(Be)_j.$$  \hspace{1cm} (10)

It follows that

$$\frac{\partial \|F - \tilde{P}\|_F^2}{\partial e_m} = 4B_m^\top(F - \tilde{P})M(Be)_m,$$  \hspace{1cm} (11)

and thereby

$$\frac{\partial \|F - \tilde{P}\|_F^2}{\partial e} = 4B^\top(F - \tilde{P})M(Be).$$  \hspace{1cm} (12)

Thus, the partial derivative of Lagrange function with respect to $e$ is:

$$\nabla_e L = 4B^\top \left( ((Be^\top B^\top) \circ M) Be + \tau 1_n \right),$$  \hspace{1cm} (13)

where $1_n$ is the $n \times 1$ vector of all ones. Using the Karush-Kuhn-Tucker (KKT) complementarity condition [3] for the non-negative constraint on $e$, we have

$$\nabla_e L \circ e = 0.$$  \hspace{1cm} (14)

The above formula leads to the updating rule for $e$ that is shown in Eq. (9).

#### 3.3.2 Convergence

We use the auxiliary function approach [17] to prove the convergence of Eq. (9) in Theorem 1. We first introduce the definition of auxiliary function as follows.
Theorem 2. Let \( L(e) \) denote the sum of all terms in \( L \) containing \( e \). The following function
\[
Z(e, \hat{e}) = -2\sum_{ij} \hat{e}_i \left\{ \left[ (B^T \hat{P} \circ M) B \right]_{ij} \right\} \left( 1 + \log \frac{\hat{e}_i e_j}{e_i \hat{e}_j} \right)
+ \sum_i \left\{ \left[ (B^T \hat{B} e \hat{e}^T B^T) \circ M \right] B e \right\} e_i^{\delta}
+ \frac{\tau}{4} \sum_i \left[ e_i^3 + 3\hat{e}_i^3 \right]
\]
is an auxiliary function for \( L(e) \). Furthermore, it is a convex function in \( e \) and has a global minimum.

Theorem 2 can be proven in a similar way as in [7] by validating \( Z(e, \hat{e}) \geq L(e) \), \( Z(e, e) = L(e) \), and the Hessian matrix \( \nabla^2_{e} Z(e, \hat{e}) \geq 0 \). Due to space limitation, the detail of the proof is omitted.

Based on Theorem 2, we can minimize \( Z(e, \hat{e}) \) with respect to \( e \) with \( \hat{e} \) fixed. We set \( \nabla_{e} Z(e, \hat{e}) = 0 \), and get the following updating formula
\[
e \leftarrow \hat{e} \circ \left\{ \frac{4 \left[ (B^T \hat{P} \circ M) B \hat{e} \right]}{4 \left[ (B^T \hat{B} e \hat{e}^T B^T) \circ M \hat{B} e + \tau 1_n \right]} \right\}^{\frac{1}{4}}
\]
which is consistent with the updating formula derived from the KKT condition aforementioned.

From Lemma 3.1 and Theorem 2, for each subsequent iteration of updating \( e \), we have \( L(e^{(t)}) = Z(e^{(t)}) \geq Z(e^{(t-1)}) \geq Z(e^{(t-2)}) \geq \ldots \geq L(e^{(t+\epsilon)}) \). Thus \( L(e) \) monotonically decreases. Since the objective function Eq. (8) is lower bounded by 0, the correctness of Theorem 1 is proven.

### 3.3.3 Complexity Analysis

In Algorithm 1, we need to calculate the inverse of an \( n \times n \) matrix, which takes \( O(n^3) \) time. In each iteration, the multiplication between two \( n \times n \) matrices is inevitable, thus the overall time complexity of Algorithm 1 is \( O(\text{Iter} \cdot n^3) \), where \( \text{Iter} \) is the number of iterations needed for convergence. In the following section, we will propose an efficient algorithm that reduces the time complexity to \( O(\text{Iter} \cdot n^2) \).

### 4. COMPUTATIONAL SPEED UP

In this section, we will propose an efficient algorithm that avoids the matrix inverse calculations as well as the multiplication between two \( n \times n \) matrices. The time complexity can be reduced to \( O(\text{Iter} \cdot n^2) \).

We achieve the computational speed up by relaxing the objective function in (8) to jointly optimize \( r \) and \( e \). The objective function is shown in the following.

\[
\min_{e \geq 0, r \geq 0} \left( I_n - \hat{A} \right) r + (1 - c) ||r - e||_F^2,
+ \lambda ||(r^T r) \circ M - \hat{P}||_F^2 + \tau ||e||_1.
\] (19)

To optimize this objective function, we can use an alternating scheme. That is, we optimize the objective with respect to \( r \) while fixing \( e \), and vise versa. This procedure continues until convergence. The objective function is invariant under these updates if and only if \( r, e \) are at a stationary point [17]. Specifically, the solution to the optimization problem in Eq. (19) is based on the following theorem, which is derived from the Karush-Kuhn-Tucker (KKT) complementarity condition [3]. The derivation of it and the proof of Theorem 3 is similar to that of Theorem 1.

Theorem 3. Alternatively updating \( e \) and \( r \) according to Eq. (20) and Eq. (21) will monotonically decrease the objective function in Eq. (19) until convergence.

\[
r \leftarrow r \circ \left\{ \frac{Ar + 2M(P \circ M)r + (1 - c)e}{r + 2\lambda (\{rr^T\} \circ M)r} \right\}^{\frac{1}{4}}
\]
(20)

\[
e \leftarrow e \circ \left\{ 2(1 - c)r \right\}^{\frac{1}{2}}
\][17]
(21)

Based on Theorem 3, we can develop the iterative multiplicative updating algorithm for optimization similar to Algorithm 1. Due to page limit we skip the details. We refer to this ranking algorithm as R-RCA. From Eq. (20) and Eq. (21), we observe that the calculation of the inverse of the \( n \times n \) matrix and the multiplication between two \( n \times n \) matrices in Algorithm 1 are not necessary. As we will see in Section 7.4, the relaxed versions of our algorithm can greatly improve the computational efficiency.

### 5. SOFTMAX NORMALIZATION

In Section 3, we use the product \( r_i \cdot r_j \) as the strength of evidence that the correlation between node \( i \) and \( j \) is vanishing (broken). However, it suffers from the extreme values in the ranking values \( r \). To reduce the influence of the extreme values or outliers, we employ the softmax normalization on the ranking values \( r \). The ranking values are nonlinearly transformed using the sigmoidal function before the multiplication is performed. Thus, the reconstruction error is expressed by \( ||(σ(r)σ^T(r)) \circ M - \hat{P}||_F^2 \), where \( σ(·) \) is the softmax function with:

\[
σ(r)_i = \frac{e_i^r}{\sum_{k=1}^n e_k^r}, (i = 1, ..., n).
\]
(22)

The corresponding objective function in Algorithm 1 is modified to the following

\[
\min_{e \geq 0} ||(σ(Be)σ^T(Be)) \circ M - \hat{P}||_F^2 + \tau ||e||_1.
\]
(23)

Similarly, the objective function for Eq. (19) is modified to the following

\[
\min_{e \geq 0, r \geq 0} \left( I_n - \hat{A} \right) r + (1 - c) ||r - e||_F^2,
+ \lambda ||(σ(r)σ^T(r)) \circ M - \hat{P}||_F^2 + \tau ||e||_1.
\]
(24)

The optimization of these two objective functions are based on the following two theorems.
Theorem 4. Updating $\mathbf{e}$ according to Eq. (25) will monotonically decrease the objective function in Eq. (23) until convergence.

$$
\mathbf{e} \leftarrow \mathbf{e} \circ \left\{ \frac{4}{3} \left[ (\mathbf{B}^\top \mathbf{P}) \circ \mathbf{M} \right] \sigma(\mathbf{Be}) + \frac{4}{3} \left[ \mathbf{B}^\top \mathbf{P} \sigma(\mathbf{Be}) \circ (\mathbf{Be}) \circ \mathbf{M} \right] \sigma(\mathbf{Be}) + \tau \mathbf{1}_n \right\},
$$

where $\mathbf{P} = \{ \mathbf{\text{diag}}[\sigma(\mathbf{Be})] - \sigma(\mathbf{Be}) \mathbf{\Sigma} \}$.

Theorem 5. Updating $\mathbf{r}$ according to Eq. (26) will monotonically decrease the objective function in Eq. (24) until convergence.

$$
\mathbf{r} \leftarrow \mathbf{r} \circ \left\{ \frac{\mathbf{A} \mathbf{r} + 2\lambda \left[ (\sigma(\mathbf{r}) \mathbf{1}_n^\top) \circ \mathbf{P} - \rho \mathbf{A} \circ \mathbf{M} \right] \sigma(\mathbf{r}) + (1 - c) \mathbf{e}}{\tau + 2\lambda \left[ (\sigma(\mathbf{r}) \circ \sigma(\mathbf{r}) \sigma^\top(\mathbf{r}) + \sigma(\mathbf{r}) \sigma^\top(\mathbf{r}) \circ \mathbf{P} \circ \mathbf{M} \sigma(\mathbf{r}) \right]}\right\},
$$

where $\mathbf{A} = \sigma(\mathbf{r}) \sigma^\top(\mathbf{r})$ and $\rho = \sigma^\top(\mathbf{r}) \sigma(\mathbf{r})$.

Theorem 4 and Theorem 5 can be proven with a similar strategy to that of Theorem 1. We refer to the ranking algorithms with softmax normalization (Eq. (23) and Eq. (24)) as RCA-SOFT and R-RCA-SOFT respectively.

6. TEMPORAL SMOOTHING ON MULTIPLE BROKEN NETWORKS

As discussed in Section 1, although the number of anomaly nodes could increase due to fault propagation in the network, the root cause anomalies will be stable within a short time period $T$[4]. Based on this intuition, we further develop a smoothing strategy by jointly considering the temporal broken networks. Specifically, we add a smoothing term $\|\mathbf{e}(t) - \mathbf{e}(t-1)\|_2^2$ to the objective functions. Here, $\mathbf{e}(t-1)$ and $\mathbf{e}(t)$ are causal anomaly ranking vectors for two successive time points. For example, the objective function of algorithm RCA with temporal broken networks smoothing is shown in Eq. (27).

$$
\min_{\mathbf{e}(t) \geq 0, 1 \leq t \leq T} \sum_{t=1}^{T} \left[ \| \left( \mathbf{Be}(t) \times \mathbf{B}^\top \right) \circ \mathbf{M} - \tilde{\mathbf{B}}^{(t)} \|_2^2 + \tau \| \mathbf{e}(t) \|_1 \right] + \alpha \| \mathbf{e}(t) - \mathbf{e}(t-1) \|_2^2
$$

(27)

Here, $\tilde{\mathbf{B}}^{(t)}$ is the degree-normalized adjacency matrix of broken network at time point $t$. Similar to the discussion in Section 3.3, we can derive the updating formula of Eq. (27) in the following.

$$
\mathbf{e}(t) \leftarrow \mathbf{e}(t) \circ \left\{ \frac{4}{3} \left[ (\mathbf{B}^\top \tilde{\mathbf{B}}^{(t)}) \circ \mathbf{M} \mathbf{Be}(t) + 2\alpha \mathbf{e}(t-1) \right] + \frac{4}{3} \left[ \mathbf{B}^\top \mathbf{Be}(t) \circ \mathbf{Be}(t) \circ \mathbf{M} \mathbf{Be}(t) + \tau \mathbf{1}_n + 2\alpha \mathbf{e}(t-1) \right] \right\},
$$

(28)

The updating formula for R-RCA, RCA-SOFT, and R-RCA-SOFT with temporal broken networks smoothing is similar. Due to space limit, we skip the details. We refer to the algorithms with temporal smoothing as T-RCA, T-R-RCA, T-RCA-SOFT and T-R-RCA-SOFT respectively.

7. EMPIRICAL STUDY

In this section, we perform extensive experiments to evaluate the performance of the proposed methods (summarized in Table 1). We use both simulated data and real-world monitoring datasets for validation. For comparison, we select several state-of-the-art methods, including mRank and gRank in [8, 13], and LBP [22]. For all the methods, the tuning parameters were tuned using cross validation. We use several evaluation metrics including precision, recall, and nDCG [12] to measure the performance. The precision and recall are computed on the top-10 ranking result, where $K$ is typically chosen as twice the actual number of ground-truth anomalies [12, 22]. The nDCG of the top-p ranking result is defined as $\text{nDCG}_p = \frac{\text{DCG}_p}{\text{DCG}_p}$, where $\text{DCG}_p = \sum_{i=1}^{p} \frac{2^{y(i)} - 1}{\log_2(i+1)}$. Here, $\text{DCG}_p$ is the DCG score on the ground-truth, and $p$ is smaller than or equal to the actual number of ground-truth anomalies. The rel is the anomaly score of the $i$th item in the ranking list of the ground-truth.

7.1 Simulation Study

We first evaluate the performance of the proposed methods using simulations. We have followed [8, 22] in generating the simulation data.

7.1.1 Data Generation

We first generate 5000 synthetic time series data to simulate the monitoring records. Each time series contains 1,050 time points. Based on the invariant model introduced in Section 2.1, we build the invariant network by using the first 1,000 time points in the time series. This generates an invariant network containing 1,551 nodes and 157,371 edges. To generate invariant network of different sizes, we randomly sample 200, 500, and 1000 nodes from the whole invariant network and evaluate the algorithms on these sub-networks.

To generate the root cause anomaly, we randomly select 10 nodes from the network, and assign each of them an anomaly score between 1 and 10. The ranking of these scores is used as the ground-truth. To simulate the anomaly propagation, we further use these scores as the vector $\mathbf{e}$ in Eq. (6) and calculate $\mathbf{r}$ ($c = 0.9$). The values of the top-30 time series with largest values in $\mathbf{r}$ are then modified by changing their amplitude value with the ratio $1+\mathbf{r}_i$. That is, if the observed values of one time series is $y(t)$, after changing it from $y(t)$ to $y(t)+\mathbf{r}_i$, the manually-injected degree of anomaly $|y(t)-y(t)|$ is equal to $1+\mathbf{r}_i$. We denote this anomaly generation scheme as amplitud-based anomaly generation.

7.1.2 Performance Evaluation

Using the simulated data, we compare the performance of different algorithms. In this example, we only consider the training time series as one snapshot; multiple snapshot cases involving temporal smoothing will be examined in the real datasets. Due to the page limit, we report the precision, recall and nDCG for only the top-10 items considering that the ground-truth contains 10 anomalies. Similar results can be observed with other settings of $K$ and $p$. For each algorithm, reported result is averaged over 100 randomly selected subsets of the training data.

From Figure 3, we have several key observations. First, the proposed algorithms significantly outperform other competing methods, which demonstrates the advantage of taking into account fault proration in ranking causal anomalies. We also notice that performance of all ranking algorithms will decline on larger invariant networks with more nodes, indicating that anomaly ranking becomes more challenging on
networks with more complex behaviour. However, the ranking result with softmax is less sensitive to the size of the invariant network, suggesting that the softmax normalization can effectively improve the robustness of the algorithm. This is quite beneficial in real-life applications, especially when data are noisy. Finally, we observe that RCA and RCA-SOFT outperform R-RCA and R-RCA-SOFT, respectively. This implies that the relaxed versions of the algorithms are less accurate. Nevertheless, their accuracies are still very comparable to those of the RCA and RCA-SOFT methods. In addition, the efficiency of the relaxed algorithms is greatly improved, as discussed in Section 4 and Section 7.4.

7.1.3 Robustness Evaluation

Practical invariant network and broken edges can be quite noisy. In this section, we further examine the performance of the proposed algorithms w.r.t. different noise levels. To do this, we randomly perturb a portion of non-broken edges in the invariant network. Results are shown in Figure 4. We observe that, even when the noise ratio approaches 50%, the precision, recall and nDCG of the proposed approaches still attain 0.5. This indicates the robustness of the proposed algorithms. We also observe that, when the noise ratio is very large, RCA-SOFT and R-RCA-SOFT work better than RCA and R-RCA, respectively. This is similar to those observations made in Section 7.1.2. As has been discussed in Section 5, the softmax normalization can greatly suppress the impact of extreme values and outliers in r, thus improves the robustness.

7.2 Ranking Causal Anomalies on Bank Information System Data

In this section, we apply the proposed methods to detect causal abnormal components on a Bank Information System (BIS) data set [8, 22]. The monitoring data are collected from a real-world bank information system logs, which contain 11 categories. Each category has a varying number of time series, and Table 2 gives five categories as examples. The data set contains the flow intensities collected every 6 seconds. In total, we have 1,273 flow intensity time series. The training data is collected at normal system states, where each time series has 168 time points. The invariant network is then generated on the training data as described in Section 2.1. The testing data of the 1,273 flow intensity time series are collected during abnormal system states, where each time series contain 169 time points. We track the changes of the invariant network with the testing data using the method described in Section 2.1. Once we obtain the broken networks at different time points, we perform causal anomaly ranking in these temporal slots jointly. Properties of the networks constructed are summarized in Table 3.

Based on the knowledge from system experts, the root cause anomaly at t = 120 in the testing data is related to “DB16”. An illustration of two “DB16” related monitoring data are shown in Figure 5. We highlight t = 120 with red square. Obviously, their behaviour looks anomalous from that time point on. Due to the complex dependency among different monitoring time series, it is impractical to obtain a full ranking of abnormal measurement. Fortunately, we have a unique semantic label associated with each measurement. For example, some semantic labels read “DB16:DISK hdx Request” and “WEB26 PAGEOUT RATE”. Thus, we can extract all measurements whose titles have the prefix “DB16” as the ground-truth anomalies. The ranking score is
Table 4: Top 12 anomalies detected by different methods on BIS data ($t=120$).

<table>
<thead>
<tr>
<th>mRank</th>
<th>gRank</th>
<th>LBF</th>
<th>RCA</th>
<th>RCA-SOFT</th>
<th>RCA</th>
<th>RCA-SOFT</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISK hda Request</td>
<td>DISK hdba Block</td>
<td>DISK hdx Request</td>
<td>DISK hdm Block</td>
<td>DISK hda Busy</td>
<td>DISK hdbu Block</td>
<td>DISK hda Busy</td>
</tr>
<tr>
<td>DISK hdbu Request</td>
<td>DISK hdbv Block</td>
<td>DISK hdx Request</td>
<td>DISK hdm Block</td>
<td>DISK hdbu Request</td>
<td>DISK hdbu Block</td>
<td>DISK hdbu Request</td>
</tr>
<tr>
<td>DISK hdx Request</td>
<td>DISK hdbv Block</td>
<td>DISK hdx Request</td>
<td>DISK hdm Block</td>
<td>DISK hdbv Block</td>
<td>DISK hdbv Block</td>
<td>DISK hdbv Block</td>
</tr>
<tr>
<td>DISK hdbv Block</td>
<td>DISK hdx Request</td>
<td>DISK hdbv Block</td>
<td>DISK hdm Block</td>
<td>DISK hdbv Block</td>
<td>DISK hdbv Block</td>
<td>DISK hdbv Block</td>
</tr>
<tr>
<td>DISK hdbv Block</td>
<td>DISK hdx Request</td>
<td>DISK hdbv Block</td>
<td>DISK hdm Block</td>
<td>DISK hdbv Block</td>
<td>DISK hdbv Block</td>
<td>DISK hdbv Block</td>
</tr>
<tr>
<td>DISK hdbv Block</td>
<td>DISK hdx Request</td>
<td>DISK hdbv Block</td>
<td>DISK hdm Block</td>
<td>DISK hdbv Block</td>
<td>DISK hdbv Block</td>
<td>DISK hdbv Block</td>
</tr>
</tbody>
</table>

Table 5: Number of “DB16” related monitors in top 32 results on BIS data ($t=120$).

<table>
<thead>
<tr>
<th>DB16</th>
<th>DB17</th>
<th>DB18</th>
<th>DB19</th>
<th>DB20</th>
<th>DB21</th>
<th>DB22</th>
<th>DB23</th>
<th>DB24</th>
<th>DB25</th>
<th>DB26</th>
<th>DB27</th>
<th>DB28</th>
<th>DB29</th>
<th>DB30</th>
<th>DB31</th>
<th>DB32</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>12</td>
<td>24</td>
<td>36</td>
<td>48</td>
<td>60</td>
<td>72</td>
<td>84</td>
<td>96</td>
<td>108</td>
<td>120</td>
<td>132</td>
<td>144</td>
<td>156</td>
<td>168</td>
<td>180</td>
<td>192</td>
</tr>
</tbody>
</table>

Table 6: Top 12 anomalies on BIS data ($t=122$).

Table 7: Top 12 anomalies reported by methods with temporal smoothing on BIS data ($t=120$-121).

Table 8: Comparison on the number of “DB16” related anomalies in top-12 results on BIS data.

Table 9: Top anomalies on coal plant data.

We further validate the effectiveness of proposed methods with temporal smoothing. We report the top-12 results of different methods with smoothing at two successive time points $t = 120$ and $t = 121$ in Table 7. The number of “DB16”-related monitors in the top-12 results is summarized in Table 8. From Tables 7 and 8, we observe a significant performance improvement of our methods with temporal broken networks smoothing compared with those without smoothing. As discussed in Section 6, since causal anomalies of a system usually do not change within a short period of time, utilizing such smoothness can effectively suppress noise and thus give better ranking accuracy.

7.3 Fault Diagnosis on Coal Plant Data

In this section, we test the proposed methods in the application of fault diagnosis on a coal plant cyber-physical...
In this section, we study the efficiency of proposed methods using the following metrics: 1) the number of iterations for convergence; 2) the running time (in seconds); and 3) the scalability of the proposed algorithms. Figure 9(a) shows the value of the objective function with respect to the number of iterations on different data sets. We can observe that the objective value decreases steadily with the number of iterations. Typically less than 100 iterations are needed for convergence. We also observe that our method with soft-max normalization takes fewer iterations to converge. This is because the normalization is able to reduce the influence of extreme values [21]. We also report the running time of each algorithm on the two real data sets in Figure 10. We can see that the proposed methods can detect causal anomalies very efficiently, even with the temporal smoothing module.

To evaluate the computational scalability, we randomly generate invariant networks with different number of nodes (with network density=10) and examine the computational cost. Here 10% edges are randomly selected as broken links. Using simulated data, we compare the running time of RCA, R-RCA, RCA-SOFT, and R-RCA-SOFT. Figure 9(b) plots the running time of different algorithms w.r.t. the number of nodes in the invariant network. We can see that the relaxed versions of our algorithm are computationally more efficient than the original RCA and RCA-SOFT. These results are consistent with the complexity analysis in Section 4.
8. RELATED WORK

In this section, we review the related work on anomaly detection and system diagnosis. In particular, we focus on the following two categories: 1) fault detection in distributed systems; and 2) graph-based methods.

For the first category, Yemini et al. [25] proposed to model event correlation and locate system faults using known dependency relationships between faults and symptoms. In real applications, however, it is usually hard to obtain such relationships precisely. To alleviate this limitation, Jiang et al. [13] developed several model-based approaches to detect the faults in complex distributed systems. They further proposed several Jaccard Coefficient based approaches to locate the faulty components [14, 15]. These approaches generally focus on locating the faulty components, they are not capable of spotting or ranking the causal anomalies.

Recently, graph-based methods have drawn a lot of interest in system anomaly detections [2, 5], either in static graphs or dynamic graphs [2]. In static graphs, the main task is to spot anomalous network entities given the graph structure [4, 11]. For example, Akoglu et al. [1] proposed OddBall to detect anomalous nodes in weighted graphs. Liu et al. [18] proposed to use frequent subgraph mining to detect non-crashing bugs in software flow graphs. However, these approaches only focus on a single graph; in comparison, we take into account both the invariant graph and the broken correlations, which provides a more dynamic and complete picture for anomaly ranking. In dynamic graphs, anomaly detection aims at detecting abnormal events [19]. Most approaches along this direction are designed to detect anomaly time-stamps in which suspicious events take place, but not to perform ranking on a large number of system components. Sun et al. proposed to use temporal graph- s for anomaly detection [20]. In their approach, a set of initial suspects need to be provided; then internal relationship among these initial suspects is characterized for better understanding of the root cause of these anomalies.

In using the invariant graph and the broken invariance graph for anomaly detection, Jiang et al. [14] used the ratio of broken edges in the invariant network as the anomaly score for ranking; Ge et al. [8] proposed InRank and gRank to rank causal anomalies; Tao et al. [22] used the loopy belief propagation method to rank anomalies. As has been discussed, these algorithms rely heavily on the percentage of broken edges in an edge of a node. Such local approaches do not take into account the global network structures, neither the global fault propagation spreading on the network. Therefore the resultant rankings can be sub-optimal.

9. CONCLUSIONS

Detecting causal anomalies on monitoring data of distributed systems is an important problem in data mining research. Robust and scalable approaches that can model the potential fault propagation are highly desirable. We develop a network diffusion based framework, which simultaneously takes into account fault propagation on the network as well as reconstructing anomaly signatures using propagated anomalies. Our approach can locate causal anomalies more accurately than existing approaches; in the meantime, it is robust to noise and computationally efficient. Using both synthetic and real-life data sets, we show that the proposed methods outperform other competitors by a large margin.

10. ACKNOWLEDGMENTS

We thank the National Science Foundation for financial support under grant number IIS-1313606, DBI-1565137, by National Institutes of Health under the grant number R01GM115833-01.

11. REFERENCES