Parallel Gradient Boosting based Granger Causality Learning

Pei Guo*, Chen Liu†, Yan Tang‡ and Jianwu Wang*
*Department of Information Systems, University of Maryland, Baltimore County, Baltimore, United States
†School of Information Science and Technology, North China University of Technology, Beijing, China
‡College of Computer and Information, Hohai University, Nanjing, China
*{peiguo1, jianwu}@umbc.edu, †liuchen@ncut.edu.cn, ‡tangyan@hhu.edu.cn

***Abstract*—Granger causality and its learning algorithms have been widely used in many disciplines to study cause-effect relationship among time series variables. In this paper, we address computing challenges of state-of-art Granger causality learning algorithms, specially when facing increasing dimensionality of available datasets. We study how to leverage gradient boosting meta machine learning techniques to achieve accurate causality discovery and big data parallel techniques for efficient causality discovery from large temporal datasets. We propose two main algorithms for gradient boosting based causality learning, and parallel gradient boosting based causality learning. Our experiments show our proposed algorithms can achieve efficient learning in distributed environments with good learning accuracy.**

***Index Terms*—Granger causality, gradient boosting, parallel machine learning, big data analytics**

I. Introduction

As a fundamental research problem in many disciplines including climatology [22] and neuroscience [14], causality studies cause-effect relationship among variables, such as temperature and humidity. Causality learning results could help many other research problems in the discipline. For instance, by studying how the climate system works from causality perspective, its findings could be used for many research areas including climate variability and climate change [29], climate dynamics [12] and climate extreme prediction [15].

With the increasing available datasets, different data-driven graphical causality discovery approaches such as dynamic Bayesian network [25] and Granger causality [18], have been proposed. We have been studying how these various approaches can help discover climate causality [19], [30], [31]. One challenge we encountered is that most state-of-art approaches will face computing challenges when they are used to discover causality from the explosion of available data with increasing dimensionality (i.e. the count of available variables) and increasing temporal resolution. Take the popular ECMWF atmospheric reanalysis data [3] as an example. Its data product transition from its current ERA-Interim to upcoming ERA5 will increase in total volume from 100 TB to 9 PB with 2.7-fold increase in variable count, over 4-fold increase in spatial resolution, 6-fold increase in temporal resolution. Current causality discovery algorithms' execution time grow quadratically or even worse with the increase of either dimensionality or spatial/temporal resolution [9]. It will be prohibitively expensive to learn causality from the whole dataset using existing approaches.

In this paper, we study how to efficiently discover cause-effect relationship from large time series data. We focus on Granger causality [18] because of its natural support of time series. We study how to leverage gradient boosting meta machine learning techniques [23] for causality discovery and parallel techniques for efficient causality discovery from large temporal datasets. Further, many scientific datasets are characterized as nonlinear because of internal nonlinear dynamics of the systems that produce the data [13]. We extend original Granger causality to support both linear and nonlinear datasets. Our open-source implementations of the algorithms can be found in Github at [4]. To the best of our knowledge, this is the first study that leverages gradient boosting and parallel techniques in Granger causality learning and can work with both linear and nonlinear datasets.

In summary, the contributions of this paper are three folds:

- We proposed algorithms to leverage gradient boosting meta-algorithm [23] and its additive model for Granger causality learning. Within the same gradient boosting framework, the algorithms can be easily configured to support linear or nonlinear datasets.
- We explored two levels of parallel opportunities in graphic Granger causality learning and proposed an algorithm to conduct gradient boosting based parallel Granger causality learning. Specifically, we combined thread pool techniques and the Spark big data platform [2] to achieve two level parallelization.
- To evaluate our proposed work, we did experiments on both algorithm learning accuracy and algorithm execution performance. Experiments results show that our algorithms can run on distributed environments and achieve scalable execution with high learning accuracy.

The rest of the paper is organized as follows. In Section II, the background of this paper is introduced. Section III explains our gradient boosting based causality learning algorithms. The parallelization of the learning algorithms is discussed in Section IV. In Section V, our experiments and results of our proposed algorithms are presented, followed by related work discussion in Section VI. Our conclusion and future work are summarized in Section VII.

II. Background

In this section, we explain background of techniques used in this paper: 1) pairwise Granger causality, 2) multivariate graphic Granger causality, and 3) Gradient boosting. Notations of the variables used in this section and later sections can be found at Table I.

TABLE I
VARIABLE NOTATION TABLE

x	Time series variable x
x_t	Variable x at timestamp t
P	Maximum time lag
$X_{l=1}^{P}$	Lagged variables of time series variable x from time lag 1 to maximum lag P: x_{t-1}, x_{t-2}, ..., x_{t-P}
L	A list of time series variables: [$l[0]$, $l[1]$, ...]
$L[i]_{l=1}^{P}$	Lagged variables of time series variable $l[i]$ from time lag 1 to maximum lag P
$L[a:b]_{l=1}^{P}$	$L[a]_{l=1}^{P}$, $L[a+1]_{l=1}^{P}$, ..., $L[b]_{l=1}^{P}$

A. Pairwise Granger Causality Model

Granger causality [18] was proposed in 1969 as a predictive model in economics by Nobel Laureate Clive W. Granger. The Granger causality is defined as follows. One time series x Granger causes another time series y, if and only if regression based prediction for y based on past values of both x and y is statistically significant than regression based prediction of y only based on past values of y. Let lagged variable x be x_{t-i} for x with time lag $l = i$, and $X_{l=1}^{P}$ denoting all the lagged variable x with time lags from 1 to maximum lag P, and similarly, lagged y be y_{t-i} and all lagged y as $Y_{l=1}^{P}$ denoting all the lagged variable y with time lag from 1 to maximum lag P. To evaluate Granger causality, it first does the following two linear regressions:

$$y_t = a_{11} \cdot y_{t-1} + a_{12} \cdot y_{t-2} + ... + a_{1P} \cdot y_{t-P} + \varepsilon_1 \quad (1)$$

$$y_t = a_{21} \cdot y_{t-1} + ... + a_{2P} \cdot y_{t-P} + b_{21} \cdot x_{t-1} + ... + b_{2P} \cdot x_{t-P} + \varepsilon_2 \quad (2)$$

Then it compares whether the regression function in Equation (1) performs better in accuracy than Equation (2) when predicting y_t. To decide which regression has better accuracy, Granger causality often uses a statistical hypothesis test method such as F-test or Chi-squared (χ^2) test to get a p-value to determine statistical significance.

The F-test is mostly used when considering a decomposition of the variability in a collection of data in terms of sums of squares. The ratio of two scaled sums of squares reflecting different sources of variability in the F-test statistic is computed. The F statistic tends to be larger when the null hypothesis is not true. In order for the statistic to follow the F-distribution under the null hypothesis, the sums of squares should be statistically independent, and each should follow a scaled Chi-squared-distribution. The latter condition is guaranteed if the data values are independent and normally distributed with a common variance.

When we apply F-test in regression problems, it is used to decide whether a model fits the data significantly better than a naive model does. In the case of Granger causality, Equation (1) is the naive model, referred as model 1, and Equation (2) referred as model 2, with Residual Sum of Square RSS_1 and RSS_2, the number of parameters p_1 and p_2 for model 1 and model 2 correspondingly. Let n be the number of total data samples, then the null hypothesis is that model 2 does not fit data better than model 1. And we can calculate the F statistics of these two models as:

$$F = \frac{\frac{RSS_1 - RSS_2}{p_2 - p_1}}{\frac{RSS_2}{n - p_2}} \quad (3)$$

Then we look up the F statistics in F-distribution with its corresponding degree of freedom to get p-value of the null hypothesis and consider rejecting the null hypothesis. If the p-value is less than or equal to the level of significance, we reject the null hypothesis because the test statistic falls in the rejection region. In other words, if we set the level of significance of p-value as 0.05, it means that the probability of "model 2 does not fits data better than model 1" is less than 5%, thus we believe that model 2 is better, which concludes x Granger causes y.

Generally speaking, the Granger causality test has two important parts: 1) regression model between the two time series with their corresponding time lagged variables and 2) the hypothesis test.

B. Multivariate Graphic Granger Causality Model

Even though the above pairwise Granger Causality can work to discover the causality between each pair of variables, when an input dataset contains multiple variables, scientists might be interested in the causality among a subset of the variable or whole variable causal relationships. In such context, pairwise Granger causality ignores the causalities with other untested variables, which could generate spurious causal relationships such as confounding variable [26] and indirect causal relationship [22].

Multivariate conditional Granger causality analysis is to address this problem by fitting a vector autoregressive model (VAR) to time series data [21]. To demonstrate conditional Granger causality in VAR model, using the same notions used in Section II.A, the joint VAR model is as follows:

$$\begin{cases} y_t = A_1 \cdot Y_{l=1}^{P} + B_1 \cdot X_{l=1}^{P} + \varepsilon_{1t} \\ x_t = C_1 \cdot X_{l-1}^{P} + D_1 \cdot Y_{l=1}^{P} + \varepsilon_{2t} \end{cases} \quad (4)$$

with the prediction error covariance matrix being:

$$CovMatrix = \begin{bmatrix} var(\varepsilon_{1t}) & cov(\varepsilon_{1t}, \varepsilon_{2t}) \\ cov(\varepsilon_{2t}, \varepsilon_{1t}) & var(\varepsilon_{2t}) \end{bmatrix} \quad (5)$$

Besides lagged variables $X_{l=1}^{P}$ and $Y_{l=1}^{P}$, when a new variable z is taken into account, the new VAR model is:

$$\begin{cases} y_t = A_2 \cdot Y_{l=1}^{P} + B_2 \cdot Z_{l=1}^{P} + C_2 \cdot X_{l=1}^{P} + \varepsilon_{3t} \\ z_t = D_2 \cdot Y_{l=1}^{P} + E_2 \cdot Z_{l=1}^{P} + F_2 \cdot X_{l=1}^{P} + \varepsilon_{4t} \\ x_t = G_2 \cdot Y_{l=1}^{P} + H_2 \cdot Z_{l=1}^{P} + I_2 \cdot X_{l=1}^{P} + \varepsilon_{5t} \end{cases} \quad (6)$$

Correspondingly, the prediction error covariance matrix of VAR model in (6) is:

$$\Sigma = \begin{bmatrix} var(\varepsilon_{3t}) & cov(\varepsilon_{3t}, \varepsilon_{4t}) & cov(\varepsilon_{3t}, \varepsilon_{5t}) \\ cov(\varepsilon_{4t}, \varepsilon_{3t}) & var(\varepsilon_{4t}) & cov(\varepsilon_{4t}, \varepsilon_{5t}) \\ cov(\varepsilon_{5t}, \varepsilon_{3t}) & cov(\varepsilon_{5t}, \varepsilon_{4t}) & var(\varepsilon_{5t}) \end{bmatrix} \quad (7)$$

Similar to the pairwise Granger causality testing, we care about whether introducing z can improve the prediction of y and how significant the improvement is. From the VAR model in Equation (4) of variable y and x, and the VAR model in Equation (6) of variable y, z, and x, the conditional Granger causality test from z to y conditioned on x, denoted as $(z \rightarrow y|x)$, is:

$$F\text{-}test(var(\varepsilon_{1t}), var(\varepsilon_{3t})) \quad (8)$$

From F-test in Equation 8, we can get a p-value and compare the p-value to a threshold to conclude whether z Granger causes y conditioned on x.

Using the above multivariate conditional Granger causality, the original pairwise causality model can be extended to a graphic model so that it can measure whether and how one variable/node of the graph is caused by multiple other variables in the graph [9], [32]. Then the final learning output can be represented as a directed graph $G = (V, E)$ where V is a set of time series variable nodes and E is a set of directed edges connecting two nodes for discovered cause-effect relationships. For instance, a directed edge in a graph from one variable node x_1 to another variable node x_2 presents that a cause-effect relationship from x_1 to x_2 is discovered conditioned on all other variable nodes in the graph, denoted as $x_1 \rightarrow x_2|(x_3, x_4, ...)$ where the orders of conditional variables do not matter. Graphic Granger causality learning algorithms are to test every possible directed edge and add the edge to the graph model if the edge satisfies the conditional Granger causality definition in Equation 8. For a dataset with N time series variables, $N*(N-1)$ number of edges need to be tested for possible causality.

C. Gradient Boosting

Gradient boosting [23] is a meta machine learning technique for regression and classification problems. On top of an additive model, the principle of gradient boosting is to eventually get a strong learner by adding multiple weak learners and classification/regression prediction error can be reduced by iteratively adding new learners to fit the prediction errors of current models.

As an ensemble model, gradient boosting combines a set of weak prediction models and can work with different types of prediction models. For gradient boosting based regression, different regression models, such as linear regression and decision tree regression, could be used in gradient boosting.

Algorithm 1 shows the process of gradient boosting regression. It requires a training dataset of variable x and y as $\{(x_i, y_i)\}_{i=1}^n$, the loss function $L(y, f(x))$, and the restriction number of iterations M to generate the output function for regression of y as $F_M(x)$. Starting from line 1, the first step of gradient boosting regression is to fit an initialization function $f_0(x)$ to predict y by minimizing the loss function. In lines 2 to 6, in each iteration m of total iteration number M (called a stage in gradient boosting), the residual is first computed using the gradient as in line 3. Then, in line 4, a regression γ_m is fitted on the residuals using training set $\{(x_i, r_{im})\}_{i=1}^n$. The gradient boosting model can be updated then in line 5 as $f_m(x) = f_{m-1}(x) + \gamma_m$ by adding the function from last iteration and the new one fitted on the residuals. After M iterations, the final output function is as $F_M(x)$ to predict y.

Algorithm 1: Gradient Boosting Regression

Input: Training dataset: $\{(x_i, y_i)\}_{i=1}^n$;
Loss function: $L(y, f(x))$;
Number of iterations: M;
Output: A regression function $F_M(x)$

1: Fit regression for y based on x_i by minimizing the loss function: $f_0(x) = \arg\min_f \sum_{i=1}^n L(y_i, f_0(x_i))$
2: **for** $m = 1, 2, ..., M$ **do**
3: Compute residual: $r_{im} = -[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}]_{f(x_i)=f_{m-1}(x_i)}$
4: Fit regression γ_m for residual r_{im} using training set $\{(x_i, r_{im})\}_{i=1}^n$
5: Update gradient boosting model: $f_m(x) = f_{m-1}(x) + \gamma_m$
6: **end for**
7: Output $F_M(x)$

III. Gradient Boosting based Causality Learning

A. Gradient Boosting based Causality Learning Design

From the previous section, it can be concluded that Granger causality is to compare regression accuracy by adding new lagged time series variables to the regression model and gradient boosting approach employs an additive model to add new learners iteratively by training each new learner based on the current learning errors (i.e., regression residual). Thus we apply gradient boosting approach to $iteratively$ evaluate the regression accuracy by adding lagged variables to the existing regression model. By comparing how regression accuracy changes when different lagged variables are added to the model, we could know whether a variable can significantly improve the regression of the target variable, and therefore conclude whether the variable Granger causes the target variable. Here, we want to highlight two main differences between our approach and regular gradient boosting approaches.

The first difference is about choosing which variables to be used for each iteration/stage of gradient boosting. Most current gradient boosting approaches train a new regression model consisting of the same variables using the current residual as learning target in each iteration. Since Granger causality tests how regression accuracy improves by adding new variables, we choose to train a new model only using one new variable for the current residual as target in each iteration. In this way,

we can directly compare the difference between the prediction accuracy of the current iteration and that of the previous iteration to conclude whether the new variable improves the regression prediction, and therefore Granger causes the target variable.

The second difference is Granger causality does not use a time series variable itself, but use the lagged variables for each original variable. For a single time series variable x with maximum lag P, we now have P variables: x_{t-1}, ..., x_{t-P}. We use $X_{l=1}^{P}$ to denote all the lagged variable x with time lags from 1 to maximum lag P. In this paper, we use all $X_{l=1}^{P}$ for each iteration/stage of gradient boosting to determine whether variable x can improve regression accuracy significantly.

B. Gradient Boosting based Causality Learning Algorithms

We first propose a Gradient Boosting based Causality discovery (GBC) algorithm to learn Granger causal relationships among multiple time series variables. Gradient boosting is utilized in the GBC algorithm by having residual based additive model in its iteration.

As part of the GBC algorithm, Algorithm 2 (GBC_Edge) shows how to learn a certain Granger causality graph edge for a specific pair of cause/effect variables conditioned on all other variables.

An important input of Algorithm 2 is called *Variable Test List*, which is used in the conditional Granger causality test mentioned in Section II-B. The sequence of variable appearing in the variable test list denotes cause variable, conditional variables and effect variable. The variable at the first position is the effect variable to be tested, the one at the last position is the cause variable, and the ones in the middle are conditional variables. Because variables are added to the existing regression model additively without recomputing the current regression, one behavior we observed is that different orders of conditional variables could produce different causality conclusions. Our current solution is to check all possible conditional variable orders and create a cause-effect edge if a causality is discovered using any variable order. So a permutation list of conditional variables is used here to compute all possible causal relationships between a cause variable and an effect variable. One variable test list represents one conditional Granger causality test. After all tests of the permutation list are done, the edges associated to the test lists with same cause-effect pairs are merged into one edge and added to Granger causality graph. For example, suppose we have 4 time series variables: x_1, x_2, x_3, x_4. We want to test one causal relationship with x_1 as effect and x_4 as cause with x_2 and x_3 as conditions, so x_1 would be located at the first position, and x_4 at the end. Since x_2 and x_3 are conditional variables, we get the permutation of x_2 and x_3 and put it in the middle of the variable test list. So, the list is created as [x_1, permutation(x_2, x_3), x_4]. The permutation of (x_2, x_3) includes [x_2, x_3], and [x_3, x_2]. So the test lists are [x_1, x_2, x_3, x_4] and [x_1, x_3, x_2, x_4]. If any of these two tests is passed, the edge $x_4 \to x_1 | (x_2, x_3)$ is added to the graph.

Algorithm 2: Gradient Boosting based Causality Discovery for a Single Edge (GBC_Edge)

Input: Variable test list containing N time series variables: $L = [\ l[0],\ l[1], ...,\ l[N-1]\ ]$;
Maxlag: P;
Learning rate: v.
Output: A directed edge in Graph:
$l[N-1] \to l[0] \mid (l[1], ..., l[N-2])$.

1: Generate lagged variables for each original time series variable in L: $L[0]_{l=1}^{P}$, $L[1]_{l=1}^{P}$, ..., $L[N-1]_{l=1}^{P}$
2: Fit regression f_0 for target variable $l[0]_t$ based on all lagged $l[0]$: $L[0]_{l=1}^{P}$, by minimizing the loss between the actual value of $l[0]_t$ and its predicted value $\widehat{l[0]}_t = f_0(L[0]_{l=1}^{P})$:
$f_0(L[0]_{l=1}^{P}) = \arg\min_f \frac{1}{2} \sum (l[0]_t - \widehat{l[0]}_t)^2$
3: **for** $n = 1, 2, ..N-1$ **do**
4: Compute residual:
$r_n = l[0]_t - f_{n-1}(L[0:n-1]_{l=1}^{P})$
5: Fit regression γ_n for residual r_n based on next lagged variables $L[n]_{l=1}^{P}$:
$\gamma_n(L[n]_{l=1}^{P}) = \arg\min_\gamma \frac{1}{2} \sum (r_n - \hat{r}_n)^2$
6: Update gradient boosting model: $f_n(L[0:n]_{l=1}^{P}) = f_{n-1}(L[0:n-1]_{l=1}^{P}) + \gamma_n(L[n]_{l=1}^{P}) * v$
7: Compute prediction error variance ε_n
8: **end for**
9: Calculate p-value from F-test:
p-value $= F(\varepsilon_{N-1},\ \varepsilon_{N-2})$
10: **if** p-value $== 0$ **then**
11: Output edge: $l[N-1] \to l[0] \mid (l[1], ..., l[N-2])$
12: **end if**

To better denote the variable test list, besides using a list of variable names, the variable at index i of the list L can be also represented by $l[i]$. Then the variable test list becomes $L = [\ l[0],\ l[1], ...,\ l[N-1]]$ for N variables.

Besides the variable test list $L = [\ l[0],\ l[1], ...,\ l[N-1]]$, other inputs of Algorithm 2 are the maximum lag we want to test, denoted as P and the learning rate in each stage of gradient boosting, denoted as v. The final output of the algorithm is a directed causality graph edge such as $l[N-1] \to l[0] \mid (l[1], ..., l[N-2])$.

In line 1 of Algorithm 2, the algorithm is initialized by creating time lagged variables for each original time series variable in the variable test list from input. In line 2, an initialization regression function f_0 is created for gradient boosting. To create this regression function, the lagged variables are first generated from rotating the values of effect variable at index 0 of variable test list L, $l[0]$, from 1 to maxlag P as $L[0]_{l=1}^{P}$. These lagged variables are fitted into a regression function to predict $l[0]_t$ to get the initialization function in gradient boosting process. The regression is trained with the minimal residual sum of squares $\arg\min_f \frac{1}{2} \sum (l[0]_t - \widehat{l[0]}_t)^2$

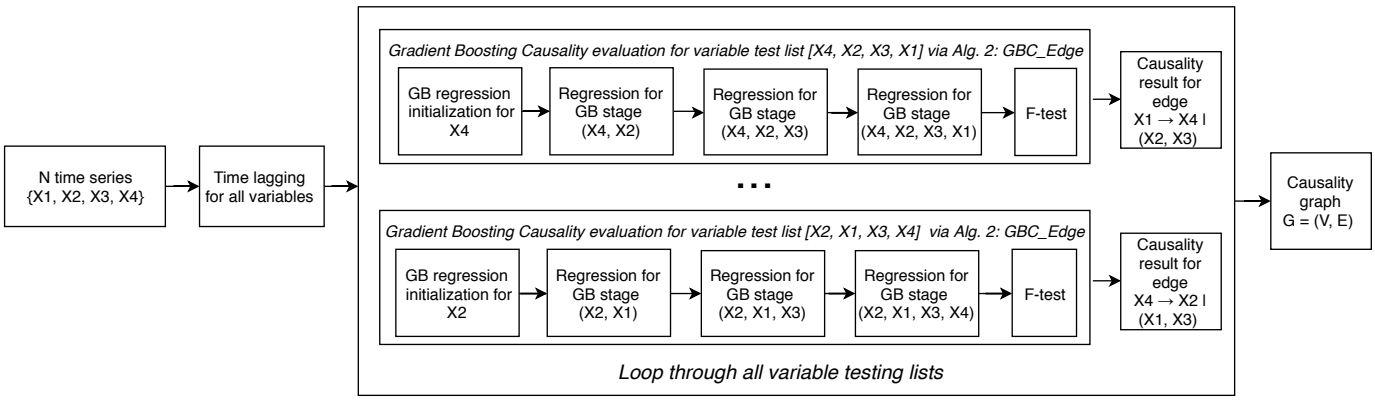

Fig. 1. Illustration of Gradient Boosting based Granger Causality Discovery Algorithm ($Algorithm\ 3:\ GBC$).

as objective function.

The gradient descent part of the algorithm is at its for loop (lines 3-8). After getting the regression model of the effect variable $l[0]$, as the first element at index 0 in the variable test list, in line 4, the gradient boosting iteration loop starts from calculating the residual r_n by subtracting $l[0]_t$'s predicted value $f_{n-1}(L[0:n-1]_{l=1}^{P})$ from its original value. Next step, in line 5, is to fit a new regression function γ_n on the residual value ($a.k.a.$ prediction error) of the target variable r_n using the lagged variables of next variable $L[n]_{l=1}^{P}$ in the variable test list using minimal residual sum of squares as loss function. Now the gradient boosting model can be updated as $f_n(L[0:n]_{l=1}^{P}) = f_{n-1}(L[0:n-1]_{l=1}^{P}) + \gamma_n(L[n]_{l=1}^{P}) * v$ in line 6. Note that the learning rate v is used to avoid over fitting and to keep enough residual value for the upcoming variables to fit the regression function in next stage. Using the new model $f_n(L[0:n]_{l=1}^{P})$ and the value of the input variables, the prediction error variance ε_n is calculated in line 7. Then the current iteration is finished, and the next variable in the variable test list is going to be put into the loop for next iteration. After the gradient boosting for loop is done, the regression part the GBC_Edge is finished.

The next important step is to do F-test in lines 9-12. From the final gradient boosting regression function and the function of its previous step, the two mean prediction error variances $\varepsilon_{N-1},\ \varepsilon_{N-2}$ perform as parameters in the F-test. Next step is to compute the F-score and get the p-value of F-test, following Equation 8. With all needed values prepared, in line 9, the F-score is calculated and a p-value of its F-distribution is derived. The hypothesis test happens in lines 10-12, where the p-value is compared to the input threshold to check if the causal relationship exists or not. In our definition, if p-value equals 0, the last element of the variable test list, namely $l[N-1]$, causes target variable $l[0]$. The causal relationship is denoted by the edge $l[N-1] \rightarrow l[0] \mid (l[1], ..., l[N-2])$.

After getting a directed edge $l[N-1] \rightarrow l[0] \mid (l[1], ..., l[N-2])$ of the graph, Algorithm 3 (GBC) is used to loop through all the variable test lists and to generate the complete gradient boosting Granger causality graph as illustrated in Figure 1.

At the beginning of Algorithm 3, for every variable, the variable test list with x_i as effect variable and x_j as cause variable will be created as $[x_i, permutation(X - \{x_i, x_j\}), x_j]$ in line 2. By looping through all variable test lists in lines 3-5, we can get all the causality edges. To generate the complete causality graph, Algorithm 2 is executed for every variable as the target variable. After all the edges are generated, the edges with same cause-effect pair are merged. For instance, if there are two edges in results from their corresponding variable test lists: $[x_1, x_2, x_3, x_4]$ and $[x_1, x_3, x_2, x_4]$, since they both have causing variable as x_4 and effect variable as x_1, we merge these two results as one causality edge: $x_4 \rightarrow x_1 | (x_2, x_3)$. The final output of Algorithm 3 is graph $G = (V, E)$ which includes V as nodes denoting all variables and a directed edge set E for all discovered causal relationships.

Algorithm 3: Gradient Boosting based Causality Discovery for All Variables (GBC)

Input: Time series with N variables:
$X = \{x_1, x_2, ..., x_N\}_{t=0}^{T}$, $t = 0, 1, ..., T$;
Maxlag: P;
Learning rate: v.
Output: Directed causality graph: $G = (V, E)$.

1: **for** $i, j = 1, 2, ..., N, i \neq j$ **do**
2: Create variable test list vtl for x_i as effect variable, and x_j as cause variable, with permutation of all other x as conditional variables:
$vtl = [x_i, permutation(X - \{x_i, x_j\}), x_j]$
3: **for** each variable test list in vtl **do**
4: $GBC\_Edge(vtl = [x_i, ..., x_j], P, v)$
5: **end for**
6: **end for**
7: Merge edges with the same cause-effect pair
8: Output Graph $G = (V, E)$

The overall process of gradient boosting Granger causality learning algorithm is illustrated in Figure 1. The input dataset contains N time series variables and every variable has corresponding lagged variables from 1 to maxlag P. Using the time series and lagged variables, the causality edges are created

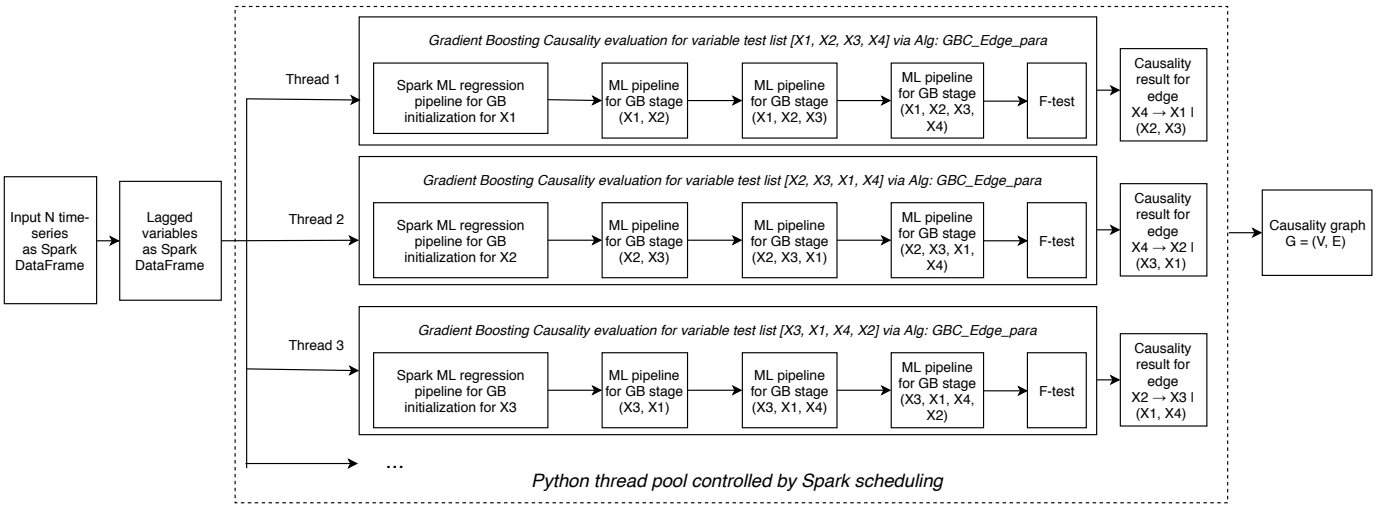

Fig. 2. Illustration of Parallel Gradient Boosting based Granger Causality Algorithm ($Algorithm\ 4:\ GBC\_para$).

by looping every variable test list using gradient boosting causality for edge algorithm GBC_Edge. After the loop, the edges are merged and the final complete causality graph $G = (V, E)$ is created.

1) Gradient Boosting Causality for Linear Dataset: If the input time series dataset is characterized as linear, the GBC algorithm will fit linear regression in its gradient boosting learning stages. That means in line 2 and line 5 of Algorithm 2, the regression should be in linear format. For instance, the linear regression fitting can be done using the $sklearn.linear\_model$ Python library, and the hypothesis test can be executed using the F-test from the $scipy.stats$ library.

2) Gradient Boosting Causality for Nonlinear Dataset: To address the nonlinearity in the nonlinear input dataset, the regression function in the algorithm will be replaced by regression trees. For instance, the $sklearn.tree$ library can be used to generate regression tree.

IV. Parallel Gradient Boosting based Granger Causality Learning

Since the complexity of the gradient boosting algorithm is relatively high, we propose two parallel algorithms to speed up the causality graph generation. The implementation of our parallel gradient boosting Granger causality discovery (Algorithm 4: GBC_para) is a parallel version of Algorithm 3 (GBC). In side of Algorithm GBC_para, it calls a parallel version of Algorithm GBC_Edge, called Algorithm GBC_Edge_para. Due to space limitation, we only describe Algorithm GBC_Edge_para in text. Besides, we illustrate the two parallel algorithms in Figure 2.

The parallelization of gradient boosting causality is implemented mainly in two levels: 1) parallel regression in every gradient boosting stage (GBC_Edge_para), and 2) parallel causality test for all possible variable test lists in order to generate the whole causality graph (GBC_para).

The first level of parallelization is within Algorithm GBC_Edge_para which extends on top of Algorithm 2 (GBC_Edge). Specifically, lines 2 and 5 in Algorithm 2 are updated using Spark DataFrame and ML pipeline [6], [24]. The algorithm creates and fits a Spark regression ML pipeline in each gradient boosting stage. For instance, as shown in the long block for Thread 1 in Figure 2, the algorithm first generates one ML regression pipeline for predicting $X1$ using lagged variables of $X1$, then gradually adds $X2$, $X3$, and $X4$ to the additive model through three regression ML pipelines. In the end, F-test is done to determine whether $X4 \rightarrow X1$ is a causality edge. By employing Spark ML pipeline, the execution can run in parallel through Spark parallelization.

The second level parallelization is within Algorithm 4:

Algorithm 4: Parallel Gradient Boosting based Causality Discovery for All Variables (GBC_para)

Input: Time series with N variables:
$X = \{x_1, x_2, ..., x_N\}_{t=0}^{T}$, $t = 0, 1, ..., T$;
Maxlag: P;
Learning rate: v.
Output: Directed causality graph: $G = (V, E)$.

1: Load data to Spark as DataFrame df, then generate lagged variables and update df
2: Create empty list $list$
3: **for** $i, j = 1, 2, ..., N, i \neq j$ **do**
4: Create a variable test list (vtl) for x_i as effect variable, and x_j as cause variable, with permutation of all other x as conditional variables: $vtl = [x_i, permutation(X - \{x_i, x_j\}), x_j]$
5: $list = list + vtl$
6: **end for**
7: Create thread pool: $threadPool$
8: $threadPool.map(GBC\_Edge\_para(df, P, v), list)$
9: Merge edges with the same cause-effect pair
10: Output Graph $G = (V, E)$

GBC_para, which runs Algorithm GBC_Edge_para in parallel for different variable test lists. In line 1, the algorithm (GBC_para) starts from loading data into Spark Dataframe and get lagged variables using Spark SQL window function. As shown in Figure 2 and lines 8-9 of the algorithm, a threading pool is created to run Algorithm GBC_Edge_para in parallel via map function. In this way, multiple Spark jobs can be submitted simultaneously. Each Spark job runs Algorithm GBC_Edge_para with assigned variable test list as one input. By using threading pool [16] along with Spark job scheduling [5] and Apache Hadoop YARN [1] as the resource manager, we can achieve parallel Spark job submission and execution. After getting every causality edge, the algorithm merges the result to generate a complete Granger causality graph.

1) Parallel Gradient Boosting Causality for Linear Dataset: The linear regression part is implemented in Spark utilizing the Linear Regression library from Spark MLlib.

2) Parallel Gradient Boosting Causality for Nonlinear Dataset: The gradient boosting tree regression algorithm is implemented in Spark utilizing the decision tree regression library from Spark MLlib.

V. Experiments

This section explains how we conduct experiments for the proposed algorithms to compare 1) causality learning accuracy, and 2) scalability. Our open-source implementations of the algorithms can be found at [4].

The experiments are conducted on top of the Google Cloud Dataproc cloud service [17], which provides clusters for running Apache Hadoop (v2.9.2) and Spark (v2.3.3) programs. The cluster mode in our experiments is standard, which contains one master nodes and several worker nodes. And the test machine type is standard 8v high-mem CPUs clusters with 52 GB memory for every master or worker node and 1500 GB standard persistent disk.

$$\begin{cases} x_1(t) = 0.95 \cdot \sqrt{2} \cdot x_1(t-1) - 0.90 \cdot x_1(t-2) \\ \quad + \varepsilon_1 \\ x_2(t) = 0.5 \cdot x_2(t-1) + \varepsilon_2 \\ x_3(t) = -0.5 \cdot x_1(t-1) + 0.25 \cdot \sqrt{2} \cdot x_3(t-1) \\ \quad + 0.25 \cdot \sqrt{2} \cdot x_2(t-1) + \varepsilon_3 \\ x_4(t) = -0.95 \cdot x_4(t-1) - 0.25 \cdot \sqrt{2} \cdot x_3(t-1) \\ \quad + \varepsilon_4 \end{cases} \tag{9}$$

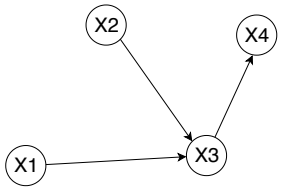

Fig. 3. Causality graph of linear synthetic data ground truth.

For test data, we created three synthetic datasets to evaluate how our proposed algorithms perform for linear, nonlinear and hybrid data. One more reason for synthetic dataset generation is to know causality ground truth so we could evaluate learning result accuracy. Similar to the synthetic dataset generation approach for Granger causality evaluation in [32], our datasets were generated based on Equations 9-11 for linear, nonlinear and hybrid of linear and nonlinear data respectively, where εs are random noises. The ground truth of causality graphs are illustrated as in Figures 3-5, respectively. Three datasets (100K, 1M and 10M for row number) were generated for all three datasets.

$$\begin{cases} x_1(t) = 0.125 \cdot \sqrt{2} \cdot \exp(-x_1(t-1)^2/2) + \varepsilon_1 \\ x_2(t) = 1.2 \cdot \exp(-x_1(t-1)^2/2) + \varepsilon_2 \\ x_3(t) = -0.5 \cdot x_1(t-1) + 0.25 \cdot \sqrt{2} \cdot x_3(t-1) \\ \quad + 0.25 \cdot \sqrt{2} \cdot x_2(t-1) + \varepsilon_3 \\ x_4(t) = -1.15 \cdot \exp(-x_1(t-1)^2/2) \\ \quad + 0.2 \cdot \sqrt{2} \cdot \exp(-x_4(t-1)^2/2) \\ \quad + 1.35 \cdot \exp(-x_3(t-1)^2/2) + \varepsilon_4 \end{cases} \tag{10}$$

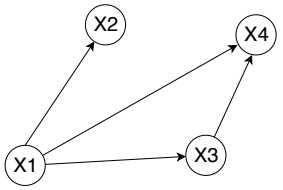

Fig. 4. Causality graph of nonlinear synthetic data ground truth.

$$\begin{cases} x_1(t) = 0.95 \cdot \sqrt{2} \cdot x_1(t-1) - 0.90 \cdot x_1(t-2) \\ \quad + \varepsilon_1 \\ x_2(t) = 1.2 \cdot \exp(-x_1(t-1)^2/2) + \varepsilon_2 \\ x_3(t) = -0.5 \cdot x_1(t-1) + 0.25 \cdot \sqrt{2} \cdot x_3(t-1) \\ \quad + 0.25 \cdot \sqrt{2} \cdot x_2(t-1) + \varepsilon_3 \\ x_4(t) = -1.15 \cdot \exp(-x_1(t-1)^2/2) \\ \quad + 0.2 \cdot \sqrt{2} \cdot \exp(-x_4(t-1)^2/2) \\ \quad + 1.35 \cdot \exp(-x_3(t-1)^2/2) + \varepsilon_4 \end{cases} \tag{11}$$

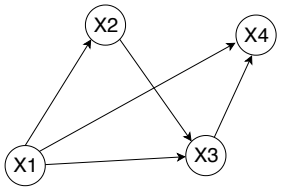

Fig. 5. Causality graph of hybrid synthetic data ground truth.

For experiment parameter setting, we set the maximum time lagging as 3. And the learning rate in the experiments is set as 0.1. The thread number set in multi-threading pool is 24. In nonlinear gradient boosting causality tests, we set the tree depth as 6 and the minimal instance at each node as 10 to avoid overfitting and get relatively stable results.

A. Causality Learning Accuracy

The first experiment compares our proposed new Granger causality algorithms with VAR based algorithm on learning

TABLE II
ACCURACY COMPARISON OF CAUSALITY GRAPHS OF GROUND TRUTH, BASELINE (VAR) AND GBC_PARA USING SYNTHETIC DATA

Data	Ground Truth	Baseline (VAR)	GBC_para
Linear (100K)	1.0	0.75	0.75 (linear)
Linear (1M)	1.0	1	1 (linear)
Nonlinear (100K)	1.0	0.67	1 (nonlinear)
Nonlinear (1M)	1.0	0.8	1 (nonlinear)
Hybrid (100K)	1.0	0.625	1 (nonlinear)
Hybrid (1M)	1.0	0.625	1 (nonlinear)

result accuracy. We choose VAR based algorithm as our baseline because it is commonly used in many studies [9], [32] and supported in a popular statistics library in Python, called $StatsModels$ [7], for Granger causality. The comparison was done through matrix distance calculation. We first converted each graph to a matrix based on the connections among its edges. Then we calculated Jaccard similarity coefficient between each learned graph with the ground truth graph.

The comparison results of causality graph for 100K and 1M datasets are shown in Table II. We ran the tests on linear datasets using the gradient boosting causality with linear regression, and utilized the nonlinear gradient boosting causality on nonlinear and hybrid datasets with decision tree regression. From the accuracy comparison table, we could see that for linear datasets, our proposed algorithms performed the same or better than the VAR based baseline approach [21] in terms of result accuracy. For nonlinear and hybrid datasets, our proposed algorithms correctly found all edges and performed better than the VAR baseline. It verifies that our proposed gradient boosting causality learning approach is valid for Granger causality discovery in these three types of datasets.

B. Scalability

Scalability with different compute nodes. We conducted experiments for our proposed algorithms for different sizes of datasets at a distributed computing environment mentioned above with 4, 6 and 8 worker nodes. The first scalability experiment is to test the speed up of our program with different numbers of worker nodes. We used the 10M row dataset for this experiment. The results are shown in Table III and Figure 6. In Figure 6, the red dotted line indicated the Nonlinear GBC_para algorithm, and the blue solid line represented the Linear GBC_para algorithm. We could see that the linear algorithm always ran a little bit faster than the nonlinear one. That was because the linear regression trained faster than the tree regression. Moreover, it was significant that with more worker nodes, the computation time decreased, which indicated the scalability of our program.

Scalability with different data sizes. We also ran tests on different sizes of datasets (100K, 1M and 10M) with linear and nonlinear GBC_para programs on 8 worker nodes and recorded the execution time. The results were shown in Table IV and Figure 7. In Figure 7, the dotted red line represented the nonlinear GBC_para program, and the solid blue line represented linear GBC_para program. The x-axis shows data

TABLE III
EXECUTION TIME OF LINEAR AND NONLINEAR PARALLEL GRADIENT BOOSTING CAUSALITY ALGORITHM FOR 10M DATA (GBC_PARA)

Worker Nodes Number	GBC_para Linear	GBC_para Nonlinear
4	0:15:23	0:17:50
6	0:09:54	0:11:36
8	0:07:58	0:09:28

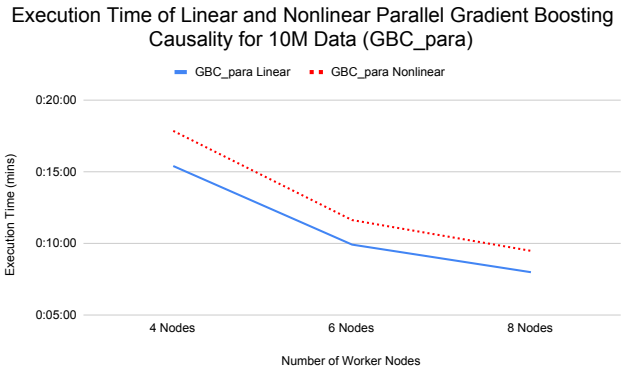

Fig. 6. Execution time of linear and nonlinear parallel gradient boosting based causality discovery algorithm for 10M row data.

size, and the y-axis indicates execution time in minutes. From the experiment, we can conclude our algorithm's execution time increases when processing larger datasets, but much slower than data size increase rate. The total execution time only increases by about 4 folds when data sizes increase by 100 folds. We also found that the minimum execution time of our algorithm was around 2 minutes due to i) the need of testing all the variable test list, ii) the overhead of executing Spark ML regression pipelines. Although each regression only took several seconds, the total number of regression training processes inside all gradient boosting regression was large. So the total execution time for smaller dataset had a bottle neck. Also, the execution time for 100K and 1M datasets were quite similar for both linear and nonlinear GBC_para programs. But for 10M dataset, since each small task in Spark ML pipeline took relatively longer time than that for the smaller dataset, the execution time differences started to show out. To solve this issue, instead of testing all the variable test list, we could apply early stop criteria in the tests in future work to improve the execution speed.

Scalability with two-level parallelization. To show the differences of *one-level* parallelization with only Spark ML regression pipelines (called GBC_Spark) and *two-level* parallelization implementation with not only Spark but also Python thread pool (GBC_para), we tested the execution time of linear and nonlinear GBC_Spark and GBC_para programs. The experiments were executed on 8 worker nodes with 100K and 1M datasets. The results were shown in Table V and Figure 8. In Figure 8, the blue bar represented dataset with the size of 100K and the light gray bar represented the 1M row dataset. It was clear that the GBC_para programs executed faster than

TABLE IV
EXECUTION TIME OF LINEAR AND NONLINEAR PARALLEL GRADIENT BOOSTING CAUSALITY FOR 100K, 1M AND 10M DATA (GBC_PARA)

Data Row Number	GBC_para Linear	GBC_para Nonlinear
100K	0:02:11	0:02:19
1M	0:02:33	0:02:32
10M	0:07:58	0:09:28

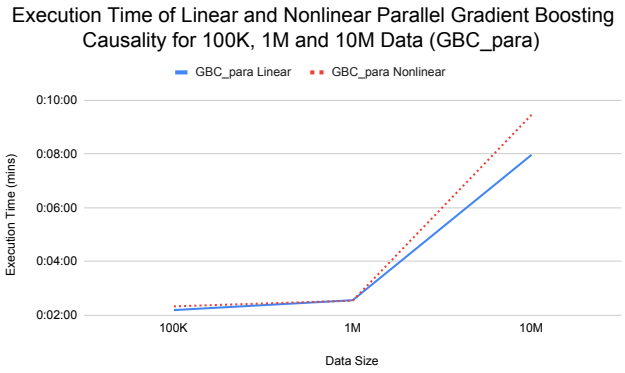

Fig. 7. Execution time of linear and nonlinear parallel gradient boosting causality for 100K, 1M and 10M data with 8 worker nodes.

TABLE V
GRADIENT BOOSTING CAUSALITY ONE-LEVEL AND TWO-LEVEL PARALLELIZATION EXECUTION TIME COMPARISON ON 100K AND 1M ROW DATA

Program	100K	1M
Two-level Parallel Linear	0:02:11	0:02:33
One-level Parallel Linear	0:07:54	0:11:24
Two-level Parallel Nonlinear	0:02:19	0:02:32
One-level Parallel Nonlinear	0:08:18	0:13:24

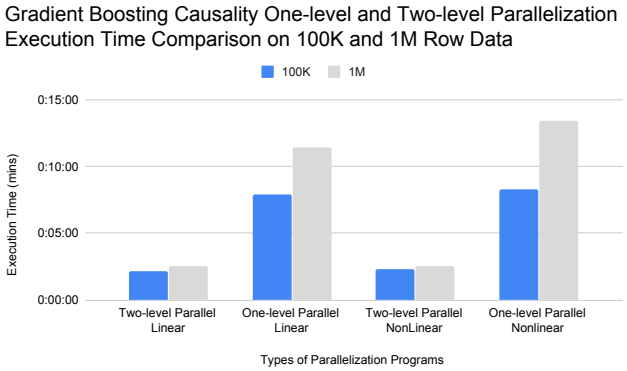

Fig. 8. Gradient boosting causality one-level and two-level parallelization execution time comparison on 100K and 1M row data.

the one-level Spark ML only GBC_Spark program in both linear and nonlinear cases.

VI. RELATED WORK

In this section, we discuss related work through two aspects: parallel machine learning and nonlinear Granger causality learning.

A. Parallel Machine Learning

As explained before, a core step in Granger causality learning is regression. Many systems now support parallel linear and/or nonlinear regression. For instance, Spark MLlib [2] supports linear regression, random forest regression and gradient-boosted tree regression. XGBoost [11] supports gradient-boosted tree regression. Our work leverages Spark Dataframe, Spark ML regression and Spark ML Pipeline in model learning. But we have not seen related studies on conducting Granger causality learning in parallel directly.

B. Nonlinear Granger Causality Learning

The original definition of Granger causality and most current Granger causality learning algorithms are based on linear regression. Because scientific data, such as climate data, are often characterized as nonlinear, there have been studies on nonlinear Granger causality discovery which can be categorized into two main types. The first category of approaches uses nonlinear (but parametric) models, such as nonlinear Fourier and wavelet transformations [10], radial basis functions [8] for nonlinear analysis of Granger causality. The second category employs information theoretical and hence non-parametric approaches including transfer entropy [27] and conditional mutual information [20], [28]. Our work belongs to the first category and our novelty lies in employing gradient boosting tree based model to discover Granger causality from nonlinear data.

VII. CONCLUSION AND FUTURE WORK

As a popular data-driven causality discovery approach, many algorithms have been proposed to learn Granger causality. Yet these existing studies rarely address their computing challenges, especially when facing large datasets. In this paper, we leverage gradient boosting technique and parallel computing technique for Granger causality discovery and proposed a set of algorithms to learn causality efficiently. We believe it is the first work that can learn Granger causality both using gradient boosting approach and in parallel. Even though the data used in our current experiments are still relatively small, we still can see clear benefits of proposed algorithms and we believe the trend will continue for bigger datasets.

For future work, we plan to mainly work on the following aspects. First, we will apply the algorithms to larger and real-world datasets, such as global climate simulation, observation and reanalysis data, to evaluate algorithm efficiency and causality learning result usefulness to the discipline. Second, we will study how to reduce the time complexity of the algorithms further by only selecting a subset of possible variable test lists, instead of full permutation of conditional variables currently used in Section III-B.

ACKNOWLEDGMENT

This work is supported by the grant CyberTraining: DSE: Cross-Training of Researchers in Computing, Applied Math-

ematics and Atmospheric Sciences using Advanced Cyberinfrastructure Resources from the National Science Foundation (grant no. OAC–1730250). The execution environment is provided through a grant from the Google Cloud Platform research credits program.

REFERENCES

[1] Apache hadoop yarn. https://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html. Accessed: 2019-08-10.

[2] Apache Spark Project. http://spark.apache.org. Accessed: 2019-07-11.

[3] ECMWF Reanalysis Datasets. https://www.ecmwf.int/en/forecasts/datasets/browse-reanalysis-datasets. Accessed: 2019-07-11.

[4] Github Repository for Big Data Climate Causality Analytics Platform - Big Data Analytics Lab - UMBC. https://github.com/big-data-lab-umbc/Big-Data-Climate-Causality-Analytics. Accessed: 2019-7-9.

[5] Job Scheduling in Apache Spark. https://spark.apache.org/docs/latest/job-scheduling.html. Accessed: 2019-08-11.

[6] Machine Learning Pipeline in Apache Spark. https://spark.apache.org/docs/latest/ml-pipeline.html. Accessed: 2019-08-11.

[7] Statistics in Python. https://www.statsmodels.org. Accessed: 2019-08-11.

[8] N. Ancona, D. Marinazzo, and S. Stramaglia. Radial basis function approach to nonlinear granger causality of time series. *Physical Review E*, 70(5):056221, 2004.

[9] A. Arnold, Y. Liu, and N. Abe. Temporal causal modeling with graphical granger methods. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, pages 66–75, New York, NY, USA, 2007. ACM.

[10] F. Benhmad. Modeling nonlinear granger causality between the oil price and us dollar: A wavelet based approach. *Economic Modelling*, 29(4):1505–1514, 2012.

[11] T. Chen and C. Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794. ACM, 2016.

[12] K. H. Cook. *Climate Dynamics*. Princeton University Press, 2013.

[13] H. A. Dijkstra. *Nonlinear climate dynamics*. Cambridge University Press, 2013.

[14] M. Ding, Y. Chen, and S. L. Bressler. 17 granger causality: basic theory and application to neuroscience. *Handbook of time series analysis: recent theoretical developments and applications*, 437, 2006.

[15] D. R. Easterling, G. A. Meehl, C. Parmesan, S. A. Changnon, T. R. Karl, and L. O. Mearns. Climate extremes: observations, modeling, and impacts. *science*, 289(5487):2068–2074, 2000.

[16] P. S. Foundation. multiprocessing - process-based parallelism. https://docs.python.org/3/library/multiprocessing.html. Accessed: 2019-08-10.

[17] Google. Cloud dataproc. https://cloud.google.com/dataproc/. Accessed: 2019-08-10.

[18] C. W. Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica*, 37(3):424–438, 1969.

[19] S. Hussung, S. Mahmud, A. Sampath, M. Wu, P. Guo, and J. Wang. Evaluation of Data-driven Causality Discovery Approaches among Dominant Climate Modes. *Technical Report HPCF-2019-12, UMBC High Performance Computing Facility, University of Maryland, Baltimore County*, 2019.

[20] M. Jafari-Mamaghani. Non-parametric analysis of granger causality using local measures of divergence. *Applied Mathematical Sciences*, 7(83):4107–4236, 2013.

[21] H. Luetkepohl. *The New Introduction to Multiple Time Series Analysis*. 01 2005.

[22] M. C. McGraw and E. A. Barnes. Memory matters: A case for granger causality in climate variability studies. *J. Clim.*, 31(8):3289–3300, Jan. 2018.

[23] R. Meir and G. Rätsch. An introduction to boosting and leveraging. In *Advanced lectures on machine learning*, pages 118–183. Springer, 2003.

[24] X. Meng, J. Bradley, B. Yavuz, E. Sparks, S. Venkataraman, D. Liu, J. Freeman, D. Tsai, M. Amde, S. Owen, D. Xin, R. Xin, M. J. Franklin, R. Zadeh, M. Zaharia, and A. Talwalkar. Mllib: Machine learning in apache spark. *J. Mach. Learn. Res.*, 17(1):1235–1241, Jan. 2016.

[25] K. P. Murphy. *Dynamic bayesian networks: representation, inference and learning*. PhD thesis, University of California, Berkeley, 2002.

[26] J. Pearl. Simpson's paradox, confounding, and collapibility. *Causality: models, reasoning and inference*, pages 173–200, 2009.

[27] K. Schindlerova. Equivalence of granger causality and transfer entropy: A generalization. 2011.

[28] A.-K. Seghouane and S.-i. Amari. Identification of directed influence: Granger causality, kullback-leibler divergence, and complexity. *Neural computation*, 24(7):1722–1739, 2012.

[29] S. Simard. *Climate Change and Variability*. IntechOpen, Aug. 2010.

[30] H. Song, J. Tian, J. Huang, P. Guo, Z. Zhang, and J. Wang. Hybrid causality analysis of ensos global impacts on climate variables based on data-driven analytics and climate model simulation. *Frontiers in Earth Science*, 7:233, 2019.

[31] H. Song, J. Wang, J. Tian, J. Huang, and Z. Zhang. Spatio-Temporal climate data causality analytics: An analysis of ENSO's global impacts. In *Proceedings of the 8th International Workshop on Climate Informatics*, pages 45–48, 2018.

[32] C. Zou, K. J. Denby, and J. Feng. Granger causality vs. dynamic bayesian network inference: a comparative study. *BMC Bioinformatics*, 10:122, Apr. 2009.