TSUPY: Dynamic Climate Network Analysis Library

Jinshu Liu
University of Rochester
jliu158@ur.rochester.edu

Yunlong Xu
University of Rochester
yxu103@u.rochester.edu

Fatemeh Nargesian
University of Rochester
fnargesian@rochester.edu

Gourab Ghoshal
University of Rochester
gghoshal@pas.rochester.edu

ABSTRACT

A climate network represents the global climate system as a network where nodes are geographical locations each represented by time-series and edges indicate the interactions of time-series. Network science has been applied to climate data to study the dynamics of a climate network. To enable network dynamics analysis on historical and real-time climate data, the core task is the efficient computation and update of correlation matrices and climate networks. We demonstrate TSUPY, a Python library, which extends Jupyter Notebook as instrumentation for performing climate network construction and analysis at interactive speed. This demonstration focuses on how TSUPY enables dynamic network analysis on climate data. We also show how TSUPY can be applied to neuro-imaging to understand the functional connectivity between brain regions.

CCS CONCEPTS

• **Information systems** → *Stream management*.

KEYWORDS

time-series, climate data, climate networks, correlation matrix

ACM Reference Format:
Jinshu Liu, Yunlong Xu, Fatemeh Nargesian, and Gourab Ghoshal. 2022. TSUPY: Dynamic Climate Network Analysis Library. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM '22), October 17–21, 2022, Atlanta, GA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3511808.3557166

1 INTRODUCTION

To identify and analyze patterns in global climate, scientists and climate risk analysts model climate data as complex networks – networks with non-trivial topological properties [1, 5, 6]. The climate network architecture represents the global climate system by a set of anomaly time-series (departure from the usual behavior) of gridded climate data and their interactions [12]. A climate data set includes remote and in-situ sensor measurements (e.g. sea surface temperature and sea level pressure) covering a grid (e.g. with a resolution of $2.5^\circ \times 2.5^\circ$). Nodes in a climate network are geographical locations, characterized by time-series and edges represent information flow between nodes. The edge weights indicate a degree of correlation between the behaviors of time-series (e.g. Pearson's correlation). Note the geographical locality of nodes does not directly imply the topology of a network.

Climate networks have been shown to be powerful tools for gaining insights on earthquakes [1], rainfalls [6], and global climate events such as El Niño [5]. The common way for network dynamics analysis is to construct networks for each hypothesized time-window and analyze them separately [4]. In most analyses, given a query window, a correlation matrix is constructed by computing the pairwise correlation of all time-series on the query window. Pearson's correlation is one of the most dominant measures for studying the pairwise climatical correlation [3]. The correlation matrix enables visualization [8], network dynamics analysis [2], as well as tasks such as community detection [11]. The quadratic complexity of all-pair correlation computation makes network construction a laborious task, particularly, for interactive data analysis.

Example 1: *Climate scientists are particularly interested in detecting and identifying El Niño and La Niña [10]. This requires measuring the spatial organization of high covariability along the Earth's surface by computing the node-weighted transitivity of the global climate network [13]. In the analysis done by Radebach et al., each geographical location in the grid of 10K locations is associated with a time-series recording the temperature of the location over 50 years with a resolution of one day. The first step of this analysis is to compute the correlation of all pairs of time-series in the global grid on a sliding window (a sequence of query windows). The query window size is commonly one year, containing 365 data points with a sliding size of 30 days. This gives us about 600 correlation matrices to compute and analyze. Computing such a matrix is time-consuming, and it becomes even more cumbersome for a sliding window query. The output of the first step is a series of correlation matrices. The second step is to convert each correlation matrix into a matrix that only contains the high correlations (weighted matrix), then compute the node-weighted transitivity of the matrix. An analysis of transitivity allows scientists to decide whether El Niño has happened in a year (query window) or not. Besides sliding the window many times, it might be needed to change the window size and the sliding step in the first step to test the robustness of results. Given a large number of time-series and their lengths, this exploratory process raises the need for computing correlation matrix with different query window sizes efficiently and at interactive speed.* □

CIKM '22, October 17–21, 2022, Atlanta, GA, USA

ACM ISBN 978-1-4503-9236-5/22/10…$15.00
https://doi.org/10.1145/3511808.3557166

To bridge the gap between climate data and network analysis, we present TSUPY[1], a data platform that enables climate scientists and decision-makers to efficiently and interactively construct and analyze climate networks on historical and real-time data. TSUPY is designed to be used as a library for interactive climate network analysis. Due to the popularity of Jupyter notebooks for data science, we chose to implement TSUPY as an extension of Jupyter notebook. TSUPY is a Python wrapper around our core library[2] implemented in golang for efficiency purposes. This library currently provides an API for 1) construction of Pearson correlation matrix on time-series, 2) weighted and unweighted network construction using a user-defined threshold, 3) construction of correlation matrices and climate networks over sliding window queries, and 4) computation of basic network properties such as transitivity. The TSUPY library is organized in a modular and extensible way to accommodate new functionalities such as clustering and community detection in a climate network. In this demo paper, we first introduce the architecture of TSUPY library (§ 2), then provide an overview of concepts related to efficient correlation matrix construction (§ 3). Next, we discuss our demonstration that will show how TSUPY can be used to perform the main steps for generating well-known results [10] in the climate network community in Jupyter Notebook at interactive speed.

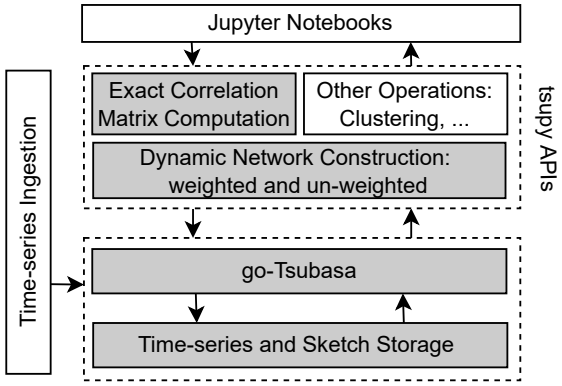

Figure 1: TSUPY Architecture.

2 ARCHITECTURE OF TSUPY

Figure 1 shows the architecture of TSUPY. The core and first task in climate network analysis is the computation of correlation matrix and construction of a climate network. The existing work [7, 15] identifies the pairs of time-series with a correlation higher that a threshold in an approximate way by reducing the correlation of time-series to the distance of their Discrete Fourier Transform (DFT) coefficients and applying grid-based indexing [15] and I/O-aware techniques [7]. Unlike the existing work, TSUPY enables exact correlation matrix computation in a speed on par with the approximate techniques. TSUPY includes the following API components.

Exact Correlation Matrix Calculation From the mathematical perspective, the analysis of the evolution of a complex system depends on the robustness and exactness of the initial weights in the complex network [5]. Moreover, in some analyses, the threshold required for filtering meaningless edges in a network is obtained based on an analysis of the entire correlation matrix [10]. Since interactivity is the main requirement in exploratory workloads, the efficiency and interactivity of correlation calculation on historical data and correlation update for real-time data is a key functionality of TSUPY, which is supported through `GetCorrelationMatrix`.

Dynamic (Un)weighted Network Construction The common way for network dynamics analysis is to construct networks for each hypothesized time-window and analyze them separately [4]. This is usually done by applying a sliding window model and computing the correlation matrix and network for a sequence of query windows [10]. For example, to analyze the temperature over years 1990 to 2020, a user may define the range `[01-1970, 12-2020]`, sliding window size of 10, and a sliding step of 5 years. This is translated into nine query windows of `[01-1970, 12-1980]`, `[01-1975, 12-1985]`, `[01-1980, 12-1990]`, `[01-1985, 12-1995]`, . . ., `[01-2010, 12-2020]`. Supporting a sliding window analysis requires a series of correlation matrix calculation and network construction.

Methods such as `GetNetworkUnweighted` and `GetNetworkWeightedRatio` support network construction. The former generates an unweighted network, where an edge exists only if the correlation of its adjacent nodes is higher than a threshold. The latter generates a weighted network, where an edge exists in the network if the correlation of its adjacent nodes is higher than a threshold and the weight of the edge becomes the correlation value. The required threshold is either provided by the user or is some ρ quantile of the correlation values in the computed matrix.

Additionally, TSUPY supports the efficient calculation of some basic network properties including the measurements of a network as well its transitivity through methods such as `GetTransitivity`, `GetTimeSeriesNum`, and `GetTimeSeriesLength`.

Implementation TSUPY performs the parallel computation of both sketching and matrix computation. It has two modes: in-memory and disk-based, depending on the storage chosen for sketches. The disk-based version of TSUPY is built on top on PostgreSQL. Methods `InitDB` and `SketchInDB` allow users to store the sketch data in database. TSUPY is a Python binding for tsubasa Golang package. It is generated by gopy. TSUPY can be easily extended with useful functions such as clustering and community detection.

3 SOLUTION SKETCH

Exact Correlation Matrix Computation We are given a collection $\mathcal{L} = \{x^1, \ldots, x^n\}$ of geo-labeled time-series, where x^i denotes the time-stamped values of a climatic variable collected at location i. A time-series x^i is defined as $[\mathbf{x}^i_1, \ldots, \mathbf{x}^i_m]$, where $\mathbf{x}^i_j$ is the observed value at time j. We assume all time-series in $\mathcal{L}$ are synchronized, i.e. each time-series has a value available at every periodic time interval, namely time resolution. This can be achieved by aggregation and interpolation on non-synchronized series.

At query time, a user defines a query time-window $w = (e, l)$, where e is end timestamp and l is the length of window. We consider the data points within w for each time-series $x = [\mathbf{x}_1, \ldots, \mathbf{x}_k]$. For example, $[\mathbf{x}_{k-m+1}, \ldots, \mathbf{x}_m]$ is the sequence we consider for x on the query window $w = (k, m)$. When clear from the context, we call

[1] https://github.com/DataIntelligenceCrew/tsupy
[2] https://github.com/DataIntelligenceCrew/tsubasa

the sequence of a time-series x, for a given query window simply query window or time-series x.

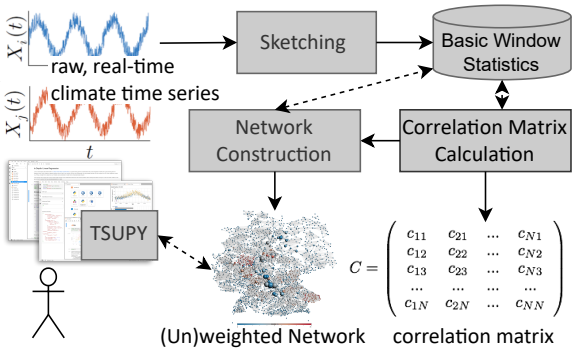

Figure 2: TSUPY Correlation Matrix/Network Computation.

The weighted climate network of $\mathcal{L}$ for a given query time-window w is a graph $\mathcal{N} = (G, V)$, where a node in G corresponds to a location i and is represented by time-series x^i. An edge in V between nodes i and l indicates that the correlation between time-series x^i and x^l is above a user-defined threshold θ. Our focus is on the most commonly used correlation measure i.e. Pearson's correlation coefficients [9].

Figure 2 illustrates a high-level overview of our prior work for constructing and updating correlation matrices on historical and real-time data [14]. The (disk-based or in-memory) storage contains a collection of frequently updated time-series accessible through locations. During the pre-processing, every time-series is divided into basic windows. The common way is to process time-series in batches of fixed size B, i.e. the stream $[\mathbf{x}_1, \ldots, \mathbf{x}_n]$ is equally divided into n/B basic windows, where the j-th basic window contains data $[\mathbf{x}_{(j-1)*B}, \ldots, \mathbf{x}_{j*B}]$. Similarly, a query window is a sequence of basic windows. The fixed-size basic window imposes limitations on the start, end, and length of query windows. To support queries with arbitrary length and position, we propose a mathematical model that works with arbitrary basic window sizes.

Subdividing a series into basic windows allows us to process data in smaller batches. We sketch basic windows of time-series, in one pass, and store the collected statistics. This can also be done at data ingestion time. At query time, the statistics of the basic windows corresponding to a given query window of all time-series are retrieved and all-pair correlations are calculated without the need to access the raw data. For real-time data, the system constructs the initial matrix and ingests the real-time raw data in chunks. The sketching of the newly ingested basic window is done on the fly and the correlations are updated incrementally without computing the correlation from scratch.

Given query windows $x = [\mathbf{x}_1, \ldots, \mathbf{x}_m]$ and $y = [\mathbf{y}_1, \ldots, \mathbf{y}_m]$ and the sizes of basic windows $\mathbf{B} = [B_1, B_2, \ldots, B_m]$, where B_i is the size of the i-th basic window. The exact Pearson's correlation of x and y is:

$$Corr(x,y) = \frac{\sum_{j=1}^{n_s} B_j(\sigma_{x_j}\sigma_{y_j}c_j + \delta_{x_j}\delta_{y_j})}{\sqrt{\sum_{i=1}^{n_s} B_i({\sigma_{x_i}}^2 + \delta_{x_i}^2)}\sqrt{\sum_{i=1}^{n_s} B_i({\sigma_{y_i}}^2 + \delta_{y_i}^2)}} \quad (1)$$

$$\delta_{x_i} = \overline{x_i} - \frac{\sum_{k=1}^{n_s} \overline{x_k}}{n_s},\ \delta_{y_i} = \overline{y_i} - \frac{\sum_{k=1}^{n_s} \overline{y_k}}{n_s}$$

where, σ_{x_i} (σ_{y_i}) is the standard deviation of basic window of x_i (y_i), c_i is the correlation of basic windows x_i and y_i, $\overline{x_i}$ ($\overline{y_i}$) is the mean of basic window x_i (y_i), and n_s is the number of basic windows in a query window. Using Equation 1, we can pre-compute and store the statistics of basic windows and compute the correlation for arbitrary query windows and sizes.

A user-defined query window on real-time data, $w = (\text{“}now\text{”}, m)$, indicates the sequence of the m most recently observed data points of time-series. That is, the size of the query window is fixed while the end timestamp is changing as new data arrives. Due to space limits, we refer the interested reader to Lemma 2 in our prior work [14] for incremental calculation of correlation for real-time data.

Contrast to Related Work Existing techniques for correlated time-series search assume that a query window is divisible by the size of a basic window and approximate the correlation using the Discrete Fourier Transform (DFT) of basic windows [7, 15]. Computing DFT coefficients has a time complexity of $O(n^2)$ in the size of a basic window. For normalized time-series, DFT preserves the Euclidean distance between two sequences. The approximation techniques consider the first few DFT coefficients to capture the shape and properties of time-series. It has been shown that the correlation of two time-series can be reduced to the Euclidean distance of the DFT coefficients of their normalized time-series [15].

We remark that these techniques do not compute exact correlation of time-series and aim at constructing networks rather than computing the complete correlation matrix which is crucial to network analysis on climate data. In our prior work [14], we describe a way of computing the correlation matrix using the DFT-based approximate techniques. Similar to the exact solution, we sketch the basic windows and use the statistics such as distance and correlation of basic windows at query time for correlation approximation.

For N time-series of each length L, the space overhead of the exact solution is $\frac{L}{B}(2 + \frac{N(N-1)}{2})$, where B is the basic window size and $\frac{L}{B}$ is the number of basic windows and the space overhead of the approximate solution is $O(\frac{LN^2}{B})$. The sketch time complexity of the exact solution is $O(L \cdot N^2)$, while the sketch time complexity of the approximate algorithm is $O(L^2 \cdot N^2)$, since the calculation of DFT coefficients for a time-series of length L is $O(L^2)$ and coefficients are required for calculating the distance of aligned basic windows in all pairs of time-series. The approximate and exact algorithms are on par in terms of query time and both have complexity of $O(\frac{l^*}{B} \cdot N^2)$.

4 DEMONSTRATION DESCRIPTION

Climate Network Analysis We will show how TSUPY can be used to efficiently replicate the steps of an existing analysis of El Niño and La Niña events in the climate network science community [10].

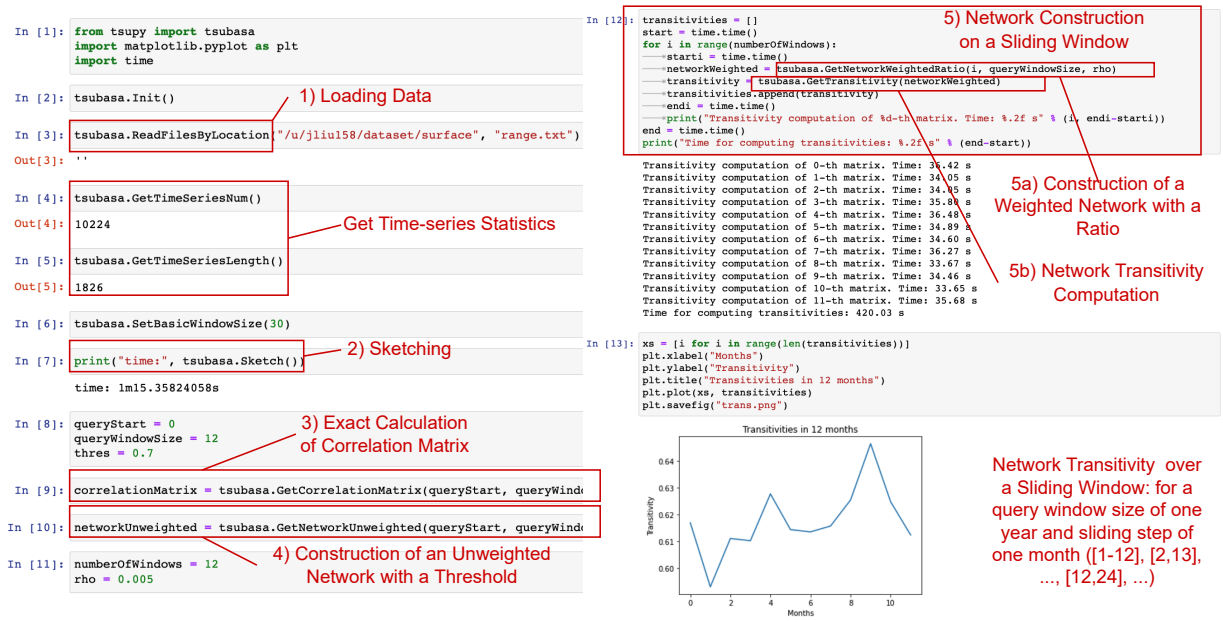

Figure 3: TSUPY Demonstration Scenario.

We make a video of an example scenario of TSUPY available[3]. We will work data set[4] used by the original work. This data set contains 10,224 temperature time-series over 1,826 days (50 years) with a time resolution of one day. User activities are shown in Figure 3.

Step 1: Loading Data Time-series data can be loaded from a data set covering the globe. In this demo, we use the data for five years. Currently, TSUPY supports netCDF file format. The user can then ask for statistics of the data set including number of time-series and their length.

Step 2: Sketching The user performs sketching of time-series which involves computing the statistics of basic windows and storing them in memory or a disk-based database.

Step 3: Correlation Matrix Calculation The user can indicate a query window by its end time and length and TSUPY computes the exact and complete correlation matrix.

Step 4: Unweighted Network Construction The user can indicate a query window and a threshold and TSUPY returns a binary matrix where a cell gets the value one if the correlation of corresponding time-series is higher than the threshold, otherwise, zero.

Step 5: Sliding Window Analysis To perform dynamic network analysis over a sliding window, the user can use simple Python commands to iterate over query windows in a sliding window. At each iteration, an unweighted correlation matrix is calculated with a user-defined parameter (e.g. $\rho = 0.005$) and the transitivity of the network is computed. Note that further analysis is required to detect an El Niño/La Niña. At a high level, for a specific year, the transitivity of its five-year period is computed from the correlation matrix generated by a query window of size five years. If the transitivity of a window starting with a month in the specified year exceeds this threshold, the year is reported as El Niño/La Niña.

Demonstration engagement In addition to our guided demonstration, participants can ask to compute the correlation matrix on their data set of choice and construct a network on it.

Other Domains In addition to the above scenario, the audience will be able to choose from a list of datasets from other domains or upload their own datasets and play with the library to write code for network analysis. For example, we will make datasets from neuroscience[5] and finance[6] domains available to the audience.

ACKNOWLEDGEMENTS

This work was partially supported by the Goergen Institute for Data Science at the University of Rochester.

REFERENCES

[1] Sumiyoshi Abe and Norikazu Suzuki. 2004. Scale-free network of earthquakes. *EPL (Europhysics Letters)* 65, 4 (2004), 581.
[2] Y. Berezin, A. Gozolchiani, O. Guez, and S. Havlin. 2012. Stability of Climate Networks with Time. *Scientific Reports* 2 (2012).
[3] Jonathan F Donges, Yong Zou, Norbert Marwan, and Jürgen Kurths. 2009. Complex networks in climate dynamics. *The European Physical Journal Special Topics* 174, 1 (2009), 157–179.
[4] James H. Faghmous and Vipin Kumar. 2014. A Big Data Guide to Understanding Climate Change: The Case for Theory-Guided Data Science. *Big Data* 2, 3 (2014), 155–163.
[5] Avi Gozolchiani, Kazuko Yamasaki, Oz Gazit, and Shlomo Havlin. 2008. Pattern of climate network blinking links follows El Niño events. *EPL (Europhysics Letters)* 83, 2 (2008), 28005.
[6] Kyunghun Kim, Hongjun Joo, Daegun Han, Soojun Kim, Taewoo Lee, and Hung Soo Kim. 2019. On complex network construction of rain gauge stations considering nonlinearity of observed daily rainfall data. *Water* 11, 8 (2019), 1578.
[7] Abdullah Mueen, Suman Nath, and Jie Liu. 2010. Fast approximate correlation for massive time-series data. In *SIGMOD*. 171–182.

[3] https://youtu.be/sCZJ12OwpvU
[4] https://psl.noaa.gov/cgi-bin/db_search/DBSearch.pl?Dataset=NCEP+Reanalysis+Daily+Averages&Variable=Air+Temperature&group=0&submit=Search
[5] https://www.humanconnectome.org/study/hcp-young-adult/project-protocol/task-fmri
[6] https://www.andrew.cmu.edu/user/rwerner/timeseries.htmi

[8] Thomas Nocke, Stefan Buschmann, Jonathan Friedemann Donges, Norbert Marwan, H-J Schulz, and Christian Tominski. 2015. visual analytics of climate networks. *Nonlinear Processes in Geophysics* 22, 5 (2015), 545–570.
[9] Karl Pearson. 1895. VII. Note on regression and inheritance in the case of two parents. *proceedings of the royal society of London* 58, 347-352 (1895), 240–242.
[10] Alexander Radebach, Reik V. Donner, Jakob Runge, Jonathan F. Donges, and Jürgen Kurths. 2013. Disentangling different types of El Niño episodes by evolving climate network analysis. *Phys. Rev. E* 88 (Nov 2013), 052807. Issue 5. https://doi.org/10.1103/PhysRevE.88.052807
[11] Alexis Tantet and Henk A Dijkstra. 2014. An interaction network perspective on the relation between patterns of sea surface temperature variability and global mean surface temperature. *Earth System Dynamics* 5, 1 (2014), 1–14.
[12] A. A. Tsonis and P. J. Roebber. 2004. The architecture of the climate network. *Physica A* 333 (Feb. 2004), 497–504.
[13] Marc Wiedermann, Alexander Radebach, Jonathan F Donges, Jürgen Kurths, and Reik V Donner. 2016. A climate network-based index to discriminate different types of El Niño and La Niña. *Geophysical Research Letters* 43, 13 (2016), 7176–7185.
[14] Yun Long Xu, Jinshu Liu, and Fatemeh Nargesian. 2022. TSUBASA: Climate Network Construction on Historical and Real-Time Data. In *SIGMOD*.
[15] Yunyue Zhu and Dennis E. Shasha. 2002. StatStream: Statistical Monitoring of Thousands of Data Streams in Real Time. In *VLDB*. 358–369.