pyPTE Documentation

Release 0.1dev

Patryk Mikulski

Jun 09, 2022

CONTENTS

pyPTE is an open-source implementation of the Phase Transfer Entropy method based on the publications [Lobier2014] and [Hillebrand2016]. It is fully implemented in Python and requires only the Python libraries NumPy and SciPy.

This implementation estimates the Phase Transfer Entropy which is defined for two given time-series X and Y with a known analysis lag delta as:

$$PTE_{X \to Y} = H(\theta_Y(t), \theta_Y(t-\delta) + H(\theta_Y(t-\delta), \theta_X(t-\delta)) - H(\theta_Y(t-\delta)) - H(\theta_Y(t), \theta_Y(t-\delta), \theta_X(t-\delta))$$

, where the entropy terms H are defined as:

$$H(\theta_Y(t), \theta_Y(t)) = -\sum p(\theta_Y(t), \theta_Y(t-\delta)) logp(\theta_Y(t), \theta_Y(t-\delta))$$
$$H(\theta_Y(t-\delta), \theta_X(t-\delta)) = -\sum p(\theta_Y(t), \theta_X(t-\delta)) logp(\theta_Y(t), \theta_X(t-\delta))$$
$$H(\theta_Y(t-\delta)) = -\sum p(\theta_Y(t)) logp(\theta_Y(t))$$
$$H(\theta_Y(t-\delta), \theta_Y(t-\delta), \theta_X(t-\delta)) = -\sum p(\theta_Y(t), \theta_Y(t-\delta), \theta_X(t-\delta)) logp(\theta_Y(t), \theta_Y(t-\delta), \theta_X(t-\delta))$$

CHAPTER

ONE

INSTALLATION GUIDE

The prerequisites for this package are:

Mandantory:

- a working Python installation, version 3.6 or higher
- git
- NumPy
- SciPy

Optional:

- mne-python
- pandas
- seaborn

Recommended:

To prevent Python module incompatibilities using a virtual environment like

- conda
- pyenv

is highly recommendable. If you are planning to use mne-python, an Anaconda3 installation is mandandory.

1.1 Step 1: Download pyPTE via GitHub

Clone into the public GitHub repository using:

```
> git clone https://github.com/patrk/pyPTE.git
```

1.2 Step 2: Build pyPTE

Build the pyPTE package and make it available to your Python interpreter by:

```
> cd pyPTE
> python setup.py install
```

1.3 Step 3: Test pyPTE

To test the installation of pyPTE simply run:

```
> cd test
> py.test
```

CHAPTER

TWO

USAGE

The main functionality of this package, calculating the Phase Transfer Entropy (PTE) for a set of time-series is accessible via the following functions:

If you are using MNE for analyzing EEG or MEG recordings an mne.io.Raw object can be passed to:

```
from pyPTE.utils.mne_tools import PTE_from_mne

dPTE, rPTE = PTE_from_mne(raw)
```

which returns a tuple of the normalized dPTE, containing information about the directionality and the raw PTE matrix, whereas the matrices are pandas DataFrames indexed by the channel names from the mne.io.Raw object.

In other domains the PTE calculation can be called directly by either passing a pandas.DataFrame to:

```
from pyPTE import pyPTE
dPTE, rPTE = pyPTE.PTE_from_dataframe(dataframe)
```

or by passing a (m x n) numpy.ndarray, where m is the number of samples and n is the number of time-series:

```
from pyPTE import pyPTE
dPTE, rPTE = pyPTE.PTE(timeseries)
```

where the returned tuple consists of the above mentioned dPTE, rPTE matrices as (n x n) numpy.ndarray objects ordered in the same way as the input object.

If you are interested in further aspects of the implementation see Developer's documentation.

CHAPTER

THREE

EXAMPLES

Currently the pyPTE package delivers three examples which shall demonstrate the usage of the phase transfer entropy method.

3.1 Standard Kuramoto model

The standard Kuramoto model is a globally coupled system of linear differential equations of first order. It represents the phase behaviour of a set of coupled oscillators with respect to their intrinsinc frequencies and a global coupling strength.

3.2 Neural mass model

This example incorporates an implementation of the stochastic non-linear dynamics of coupled cortical columns based on [Jansen1995] and [Wendling2000]

3.3 mne-python sample data set

This example illustrates how to extract data from a mne raw object, which can be fed into pyPTE. If you want to incorporate data from other software packages than MNE, please refer to the MNE documentation how to import raw data from other fileformats.

CHAPTER

FOUR

DEVELOPER'S DOCUMENTATION

4.1 To-do

Different approaches to estimate the predicition delay used for the Phase Transfer Entropy method shall be implemented and evaluated. Especially, a more sophisticated estimation as suggested by [Wibral2013] is desirable.

pyPTE is an open-source implementation of the Phase Transfer Entropy method based on the publications [Lobier2014] and [Hillebrand2016]. It is fully implemented in Python and requires only the Python libraries NumPy and SciPy.

This implementation estimates the Phase Transfer Entropy which is defined for two given time-series X and Y with a known analysis lag delta as:

$$PTE_{X \to Y} = H(\theta_Y(t), \theta_Y(t-\delta) + H(\theta_Y(t-\delta), \theta_X(t-\delta)) - H(\theta_Y(t-\delta)) - H(\theta_Y(t), \theta_Y(t-\delta), \theta_X(t-\delta))$$

, where the entropy terms H are defined as:

$$H(\theta_Y(t), \theta_Y(t)) = -\sum p(\theta_Y(t), \theta_Y(t-\delta)) logp(\theta_Y(t), \theta_Y(t-\delta))$$

$$H(\theta_Y(t-\delta), \theta_X(t-\delta)) = -\sum p(\theta_Y(t), \theta_X(t-\delta)) logp(\theta_Y(t), \theta_X(t-\delta))$$

$$H(\theta_Y(t-\delta)) = -\sum p(\theta_Y(t)) logp(\theta_Y(t))$$

$$H(\theta_Y(t-\delta), \theta_Y(t-\delta), \theta_X(t-\delta)) = -\sum p(\theta_Y(t), \theta_Y(t-\delta), \theta_X(t-\delta)) logp(\theta_Y(t), \theta_Y(t-\delta), \theta_X(t-\delta))$$

CHAPTER

FIVE

INDICES AND TABLES

- genindex
- modindex
- search

BIBLIOGRAPHY

[Lobier2014] Muriel Lobier, Felix Siebenhühner, Satu Palva and J. Matias Palva. Phase transfer entropy: A novel phase-based measure for directed connectivity in networks coupled by oscillatory interactions. *NeuroImage*, 2014. doi:10.1016/j.neuroimage.2013.08.056

[Hillebrand2016] Arjan Hillebrand, Preejaas Tewarie, Edwin van Dellen, Meichen Yu, Ellen W. S. Carbo, Linda Douw, Alida A. Gouw, Elisabeth C.W. van Straaten and Cornelies J. Stam. Direction of information flow in large-scale resting-state networks is frequency-dependent. *Proceedings of the National Academy of Sciences of the United States of America*, 113(14):3867-72, 2016. doi:10.1073/pnas.1515657113

[Jansen1995] Ben H. Jansen and Vincent G. Rit. Electroencephalogram and visual evoked potential generation in a mathematical model of coupled cortical columns. *Biological Cybernetics*, 73:357-366, 1995 doi:10.1007/BF00199471

[Wendling2000] Wendling, F. and Bellanger, J. J. and Bartolomei, F. and Chauvel, P. Relevance of nonlinear lumped-parameter models in the analysis of depth-EEG epileptic signals. *Biological Cybernetics*. 2000 doi:10.1007/s004220000160

[Wibral2013] Wibral, Michael and Pampu, Nicolae and Priesemann, Viola and Siebenhühner, Felix and Seiwert, Hannes and Lindner, Michael and Lizier, Joseph T and Vicente, Raul. Measuring Information-Transfer Delays. *PLOS ONE 8(2): e55809*, 2013. doi:10.1371/journal.pone.0055809

[Lobier2014] Muriel Lobier, Felix Siebenhühner, Satu Palva and J. Matias Palva. Phase transfer entropy: A novel phase-based measure for directed connectivity in networks coupled by oscillatory interactions. *NeuroImage*, 2014. doi:10.1016/j.neuroimage.2013.08.056

[Hillebrand2016] Arjan Hillebrand, Preejaas Tewarie, Edwin van Dellen, Meichen Yu, Ellen W. S. Carbo, Linda Douw, Alida A. Gouw, Elisabeth C.W. van Straaten and Cornelies J. Stam. Direction of information flow in large-scale resting-state networks is frequency-dependent. *Proceedings of the National Academy of Sciences of the United States of America*, 113(14):3867-72, 2016. doi:10.1073/pnas.1515657113