Reverifying accuracy and baseline for the methods established in causal gateways and mediators in complex spatio-temporal system.

Abstract

In a complex spatio-temporal system such as Earth's climate, extream natural event or geo engineering can have a spreading or mediating perturbations through causal gateway regions. Article proposes a data-driven approach based on dimension reduction and causal reconstruction along with a network measures based on causal effect theory. In this article we will establish baseline and re verify accuracy for the work done in causal gateways and mediators in complex spatio-temporal system. We will first explore the process of data gathering, then analyze the data, reduce the dimension for processing, apply causal reconstruction techniques, estimate the causal effects and finally quantify the uncertanties.

1. Data collection

All the data sets used in this article can be collected through NOAA Physical Sciences labaratory under NCEP/NCAR Reanalysis. I have written a small script download_data.py for downloading all the data we will need through concurrent threads. We can gather all the data from this resource.

2. Data analysis

The data we collected in the previous step are in cdf format and we will use python [netCDF4] (https://unidata.github.io/netcdf4-python/) package to handle the data. In climate research, spatio-temporal data sets are typically given on a regular grid. Here we consider a reanalysis data set of surface pressures for period 9448-2012. At a resolution of 2.5^o in latitude and longitude. As an introductory data analysis we will also consider surface air temperature data. These data sets contain four primary variables and they are time, latitude, longitude and a variable of interest (temperature, pressure etc). Some data sets might have additional component called level which reperesnts the altitude from the surface. Let's load the air temperature for the year 1950 Jan and visualize it.

In [1]:

```
import sys
sys.path.append('venv/lib/python3.9/site-packages')

import numpy as np
import pandas as pd
import geopandas
import datetime
import tigramite
import causal_framework
from geo_data_loader import Daily4x, MonthlyMean
```

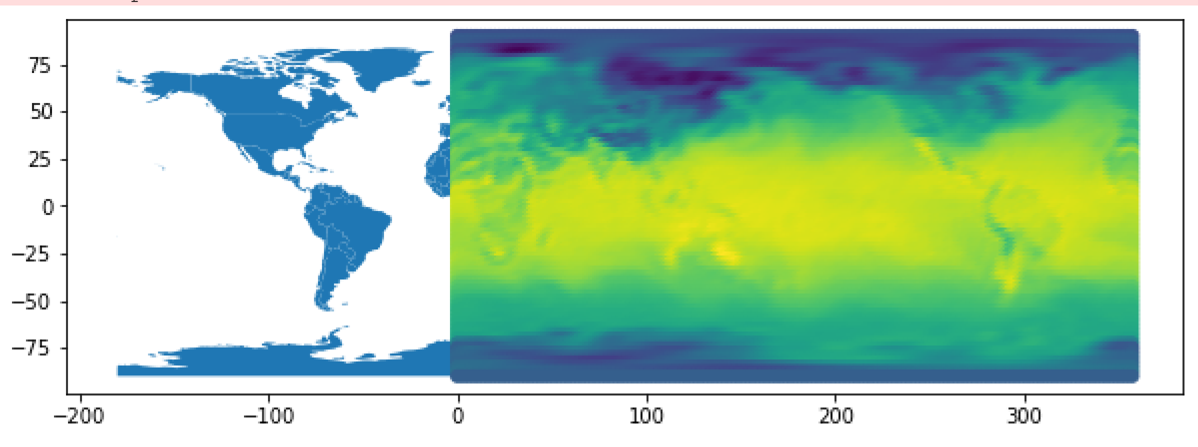
I have written a custom wrapper Daily4x under geo_data_loader for handling data of different sampling. I have also introduced visulization componenent through geopandasto understand our data better.

In [2]:

```
cdf_air_temperature = Daily4x(netcd4_loc='/Volumes/ext_data/ncep/air.sig995.1980.nc', variable='air', year=198(
print(cdf_air_temperature)
cdf_air_temperature.render(variable='air', time_slice=0)
```

```
INFO:Loaded air of shape (1464, 73, 144)
<class 'netCDF4._netCDF4.Dataset'>
root group (NETCDF4_CLASSIC data model, file format HDF5):
    Conventions: COARDS
    title: 4x daily NMC reanalysis (1980)
    description: Data is from NMC initialized reanalysis
(4x/day).  These are the 0.9950 sigma level values.
    platform: Model
    history: created 95/02/06 by Hoop (netCDF2.3)
Converted to chunked, deflated non-packed NetCDF4 2014/09
    dataset_title: NCEP-NCAR Reanalysis 1
    References: http://www.psl.noaa.gov/data/gridded/data.ncep.reanalysis.html
    dimensions(sizes): lon(144), lat(73), time(1464)
    variables(dimensions): float32 lat(lat), float32 lon(lon), float64 time(time), float32 air(time, lat, lon)
    groups:

INFO:Elapsed time 4.399535
```



3. Dimension reduction

In the original article the authors use dimension reduction based on Varimax-rotated principal components.They make a claim thet rotation of principal components better represents regionally confined process than regular principal components since they maximize the sum of variances of the squared pricipal components. This needs to verfied.

Lets load monthly mean pressure data for years 1948 to 2000.

In [3]:

```
def pca_eigenvals(d):
    """
    Compute the eigenvalues of the covariance matrix of the data d.
    The covariance matrix is computed as d * d^T.
    """
    from scipy.linalg import svdvals, svd
    # remove mean of each row
    d = d - np.mean(d, axis=1)[:, np.newaxis]
    return 1.0/(d.shape[1] - 1) * svdvals(d, True)**2

def pca_eigenvals_gf(d):
    """
    Compute the PCA for a geo-field that will be unrolled into one dimension.
    axis[0] must be time, other axes are considered spatial
    and will be unrolled so that the PCA is performed on a 2D matrix.
    """
    # reshape by combining all spatial dimensions
    # np.prod(d.shape[1:]) -> (lat * long)
    d = np.reshape(d, (d.shape[0], np.prod(d.shape[1:])))

    # we need the constructed single spatial dimension to be on axis 0
    d = d.transpose()

    return d

cdf_m_slp = MonthlyMean(netcd4_loc='/Volumes/ext_data/ncep/pres.mon.mean.nc', start_year=1948, end_year=2000, v
d = np.asarray(cdf_m_slp.get_data())
d = pca_eigenvals_gf(d)
d = pca_eigenvals(d)
print(f'Eigen Values: {d[:10]}...')
#print(d)
#d = pca_eigenvals_gf(d)
#print(d.shape)
```

```
Eigen Values: [8793740.      674700.1     232180.95    184004.27    108097.25     75106.66
   69818.17     67611.79     53730.28     47706.547]...
```

4. Casual Reconstruction

To reconstruct the causal network from the components time series, we will utilize PC causal discovery algorithm . To accomplish this we will leverage a python package called Tigramite. Tigramite is a time series analysis module that allows us to reconstruct conditinal indpenence graphical models from discerete or continuous time series based on PCMCI framework. Before we can apply the framework for the data gathered in the previous step we need to understand how the framework is working

4.1 Background

Let's say a system has 4 variables X^0, X^1, X^2, X^3 and we collected some observations through time for these variables.

Example:

Time	X^0	X^1	X^2	X^3
T=0	-0.6440979	-1.04341823	0.34727786	0.33279255
T=1	0.82125751	-0.18338834	-1.96378392	0.24609568
T=2	0.65193283	-0.83690681	2.23499787	-1.14000882
T=3	...	...	...	...
T=.	...	...	...	...
T=t-1	X^0_{t-1}	X^0_{t-1}	X^0_{t-1}	X^0_{t-1}
T=t	X^0_t	X^1_t	X^2_t	X^3_t

We do not know the true process that is generating the data. The PCMCI methods will take this data and conduct a conditional indepedence test to establish underlying causal structure. Let's generate some observational data and see this in action.

In [4]:

```
observations = causal_framework.generate_sample_observation()
print(observations.values[:10])
```

```
[[-1.18379466 -1.84759579 -0.18239229  0.15602285]
 [ 1.1089801  -1.30191425 -0.86482542 -0.25066518]
 [ 0.21923932  1.28096174  1.79884782  0.76427155]
 [ 0.21990259  0.43981213  1.44344251  0.47093879]
 [-0.93528768  1.28527828  1.62397535  2.42163065]
 [-1.46725956  3.73568783  0.30371923 -0.21979454]
 [-3.67639193  1.17032891 -0.03832578  1.82940314]
 [-2.42313568  2.71916198  3.79066395  1.28690431]
 [-4.63974627  2.05524538  3.39112678  0.81165969]
 [-4.55628977  1.79532817  3.18570216  0.54366787]]
```

In [6]:

```
def conditional_independence_test(dataframe):
    from tigramite.pcmci import  PCMCI
    from tigramite.independence_tests import ParCorr
    parcorr = ParCorr(significance='analytic')
    pcmci = PCMCI(dataframe=dataframe, cond_ind_test=parcorr, verbosity=1)
    results = pcmci.run_pcmci(tau_min=2, tau_max=10, pc_alpha=None)
    return results

results = conditional_independence_test(dataframe=observations)
```

```
##
## Step 1: PC1 algorithm with lagged conditions
##

Parameters:
independence test = par_corr
tau_min = 2
tau_max = 10
pc_alpha = [0.05, 0.1, 0.2, 0.3, 0.4, 0.5]
max_conds_dim = None
max_combinations = 1



## Resulting lagged parent (super)sets:

    Variable $X^0$ has 7 link(s):
    [pc_alpha = 0.5]
        ($X^1$ -2): max_pval = 0.00000, min_val = -0.625
        ($X^0$ -2): max_pval = 0.00932, min_val =  0.295
        ($X^3$ -3): max_pval = 0.04149, min_val = -0.234
        ($X^1$ -6): max_pval = 0.24339, min_val =  0.135
        ($X^2$ -3): max_pval = 0.43241, min_val =  0.092
        ($X^1$ -8): max_pval = 0.44011, min_val = -0.089
        ($X^3$ -4): max_pval = 0.48257, min_val =  0.081

    Variable $X^1$ has 2 link(s):
    [pc_alpha = 0.05]
        ($X^3$ -2): max_pval = 0.00000, min_val =  0.613
        ($X^1$ -2): max_pval = 0.00032, min_val =  0.397

    Variable $X^2$ has 3 link(s):
    [pc_alpha = 0.3]
        ($X^1$ -2): max_pval = 0.00000, min_val =  0.610
        ($X^1$ -4): max_pval = 0.01593, min_val =  0.272
        ($X^3$ -4): max_pval = 0.22954, min_val =  0.137

    Variable $X^3$ has 2 link(s):
    [pc_alpha = 0.1]
        ($X^1$ -4): max_pval = 0.05127, min_val = -0.220
        ($X^3$ -8): max_pval = 0.06152, min_val = -0.211

##
## Step 2: MCI algorithm
##

Parameters:

independence test = par_corr
tau_min = 2
tau_max = 10
max_conds_py = None
max_conds_px = None

## Significant links at alpha = 0.05:

    Variable $X^0$ has 4 link(s):
        ($X^1$ -2): pval = 0.00000 | val = -0.737
        ($X^3$ -2): pval = 0.00002 | val = -0.484
        ($X^0$ -2): pval = 0.00005 | val =  0.471
        ($X^3$ -3): pval = 0.03030 | val = -0.256

    Variable $X^1$ has 3 link(s):
        ($X^3$ -2): pval = 0.00000 | val =  0.616
        ($X^1$ -2): pval = 0.00030 | val =  0.401
        ($X^2$ -2): pval = 0.03954 | val = -0.238

    Variable $X^2$ has 7 link(s):
        ($X^1$ -2): pval = 0.00000 | val =  0.687
        ($X^1$ -4): pval = 0.00005 | val =  0.446
        ($X^0$ -3): pval = 0.00062 | val =  0.399
        ($X^3$ -3): pval = 0.00264 | val =  0.342
        ($X^3$ -4): pval = 0.00443 | val =  0.323
        ($X^2$ -9): pval = 0.04608 | val = -0.233
        ($X^2$ -10): pval = 0.04627 | val = -0.232

    Variable $X^3$ has 1 link(s):
        ($X^0$ -6): pval = 0.03467 | val = -0.251
```

The algorithm outputs the predicted graph structure with certain probablities. For instance it predicts variable X^0 has a causal link with variable X^1 from time step t=t-2, X^3 from time step t=t-3, X^0 from time step t=t-2 and so on. We can structurally represent them in this form.

$$X^0_t = 0.6 * X^0_{t-1} - 0.8 * X^1_{t-1} + \mu^0_t$$

$$X^1_t = 0.8 * X^1_{t-1} + 0.8 * X^3_{t-1} + \mu^0_t$$

$$X^2_t = 0.2 * X^2_{t-1} - 0.4 * X^1_{t-2} - 0.1 * X^3_{t-3} + \mu^0_t$$

$$X^3_t = 0.6 * X^2_{t-2} + \mu^0_t$$

Note, these structural equations are just for representation to convery the ideas to interpret the results.

4.2 Applying causal reconstruction to surface pressure data

To be continued.

In [ ]: