Potsdam Institute for Climate Impact Research

Pyunicorn software

Introduction

Catrin Kirsch

20 September 2017

Leibniz Association

The pyunicorn people

- Developed at PIK's Research Domains I and IV since 2008

- **Contributors:** Jobst Heitzig, Jakob Runge, Alexander Radebach, Hanna Schultz, Marc Wiedermann, Alraune Zech, Jan Feldhoff, Aljoscha Rheinwalt, Hannes Kutza, Boyan Beronov, Paul Schultz, Stefan Schinkel, Jonathan Donges,...

The pyunicorn package

Why Python?

- Easy to learn
- Full-fledged programming language
- Usable on different levels of sophistication
- It's free: open source!

The pyunicorn package

Why Pyunicorn?

Provides classes and functions for:

- Complex network statistics and models
- Nonlinear time series analysis
- Focus on spatial and functional networks: climate networks, recurrence networks, visibility graphs, ...

The pyunicorn structure

- **Core:**
- **Climate:**
- **Timeseries:**
- **Funcnet:**
- **Utils:**

The pyunicorn structure

- **Core:** basic data handling, general network analysis and modeling, power grids

  data.py, grid.py, network.py, geo_network.py, interacting_networks.py
- **Climate:**
- **Timeseries:**
- **Funcnet:**
- **Utils:**

The pyunicorn structure

- **Core:**
- **Climate:** processing climate data, climate network construction and analysis

  climate_data.py, climate_network.py, coupled_climate_network.py
- **Timeseries:**
- **Funcnet:**
- **Utils:**

The pyunicorn structure

- **Core:**
- **Climate:**
- **Timeseries:** recurrence plots, recurrence networks, multivariante extensions and visibility graph analysis of time series, time series surrogates for significance testing

  recurrence_plot.py, recurrence_network.py, surrogates.py, visibility_graph.py
- **Funcnet:**
- **Utils:**

The pyunicorn structure

- **Core:**
- **Climate:**
- **Timeseries:**
- **Funcnet:** constructing and analyzing general functional networks

  coupling_analysis.py
- **Utils:**

The pyunicorn structure

- **Core:**
- **Climate:**
- **Timeseries:**
- **Funcnet:**
- **Utils:** parallelization, interactive network navigator, helpers

The pyunicorn dependencies

- Numpy 1.8+
- Scipy 0.14+
- Weave 0.15+
- igraph and Python-igraph 0.7+

The pyunicorn platforms

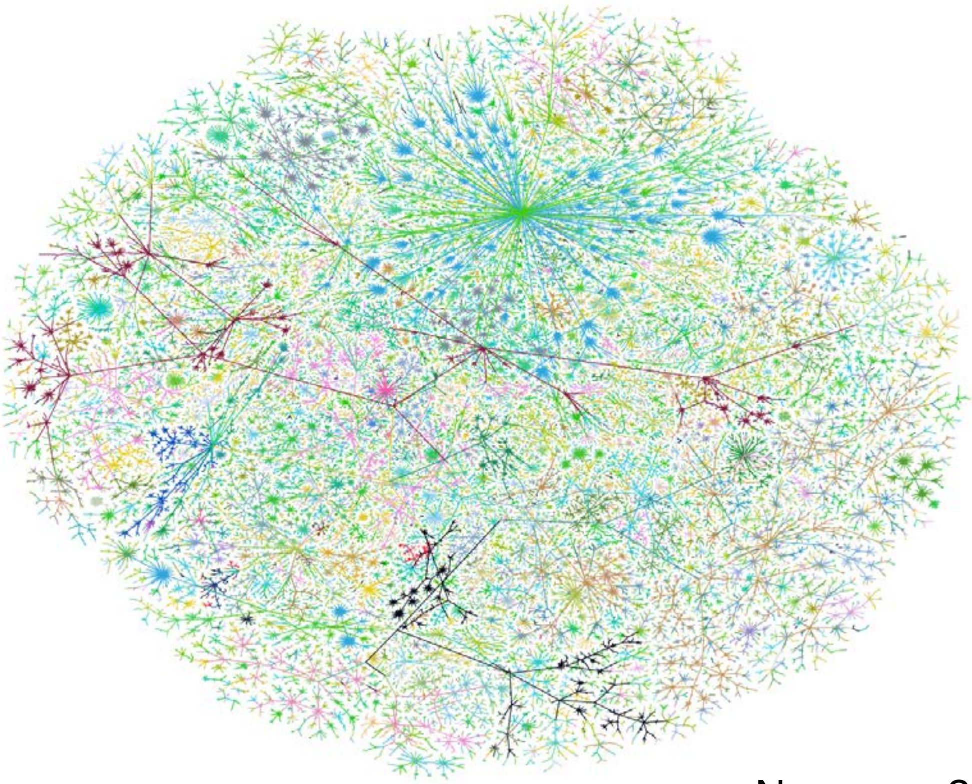
- pyunicorn runs on
  - Unix: Linux, IPLEX, Mac OS X
  - Windows

The pyunicorn links to other packages and software

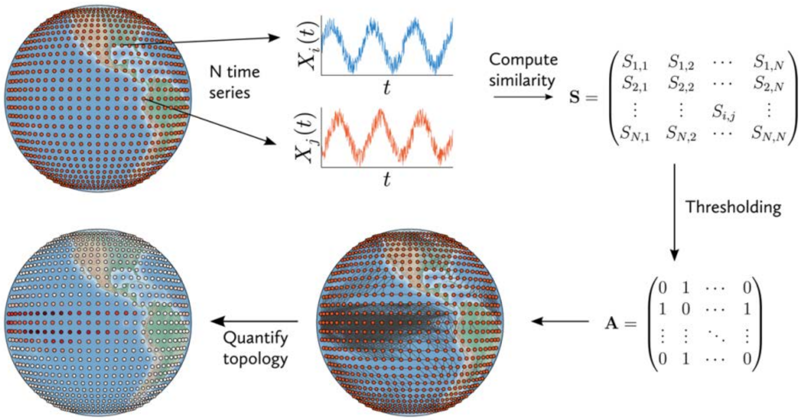
- Easy exchange with standard Python packages: *numpy, scipy, scikit-learn, matplotlib.*
- Exchange network data with *igraph, networkx, graph-tool* through various data formats.
- Save to and load from various standard graph formats, e.g, for visualization in *CGV, Gelphi.*

The pyunicorn examples

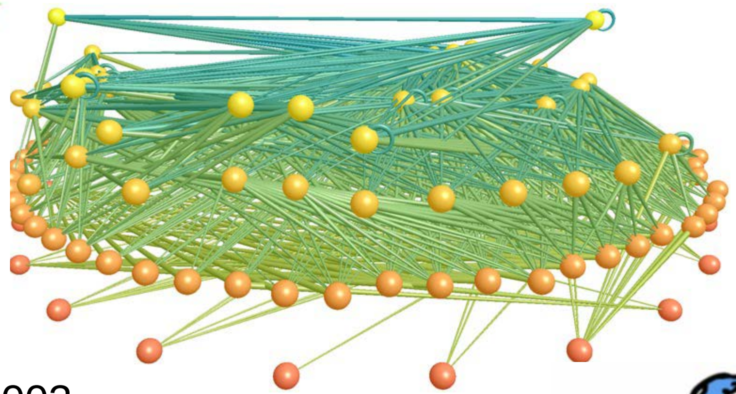
General complex networks



Newman, 2003

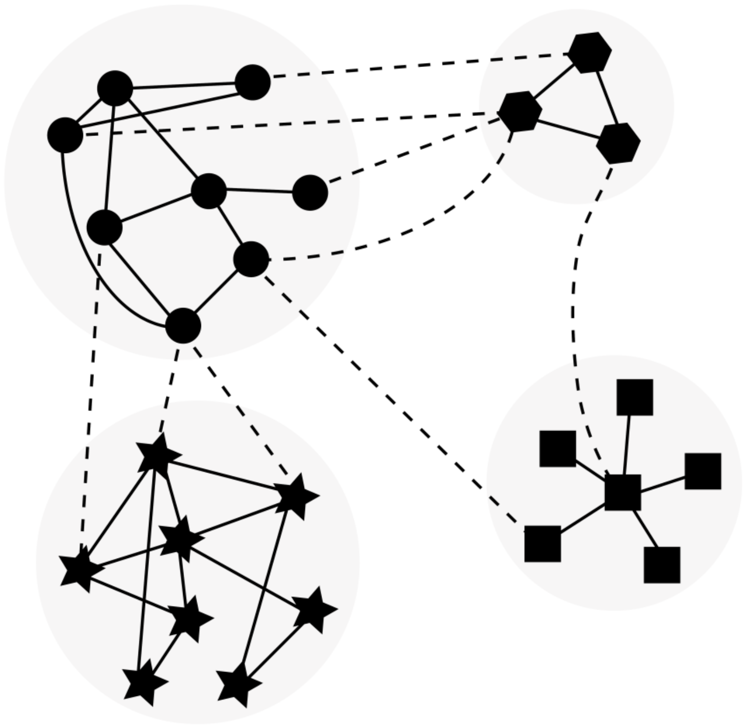


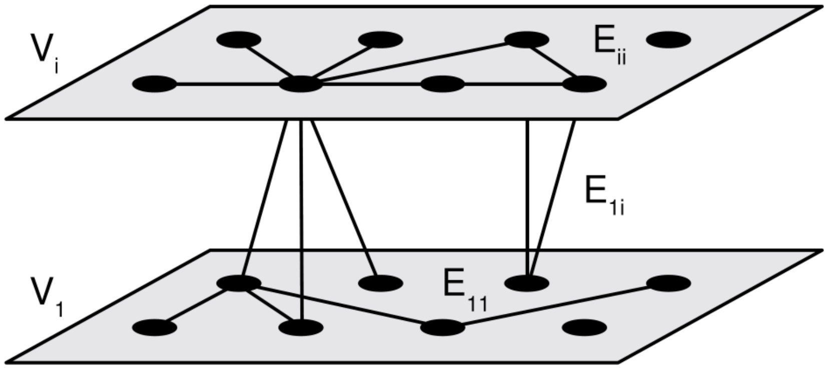
Donner et al., 2017



The pyunicorn examples

Interacting/interdependent networks / networks of networks

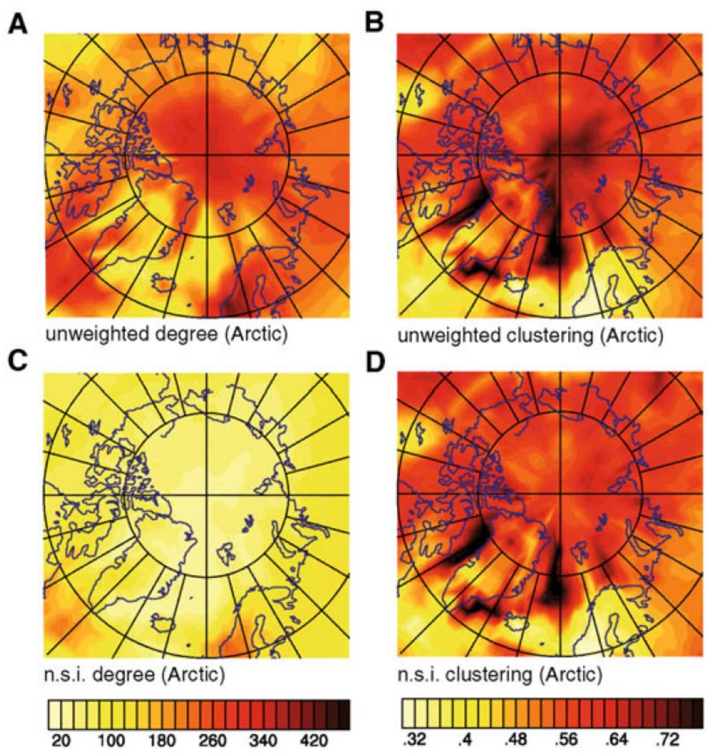


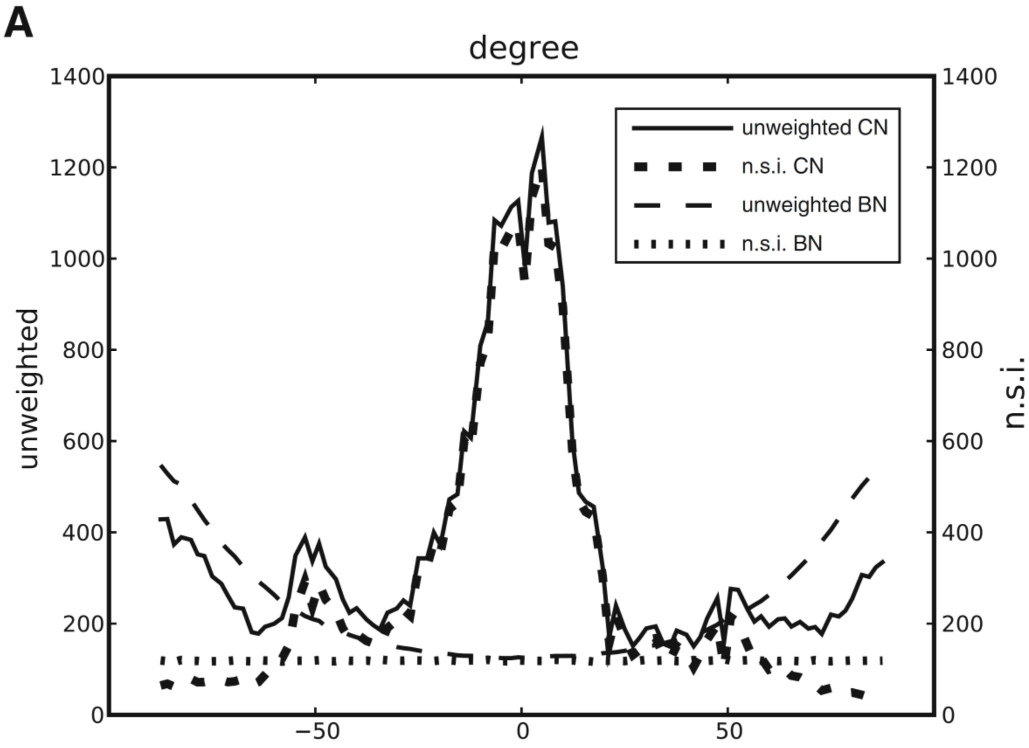


Donges et al., 2010

The pyunicorn examples

Node-weighted network measures / node-splitting invariance



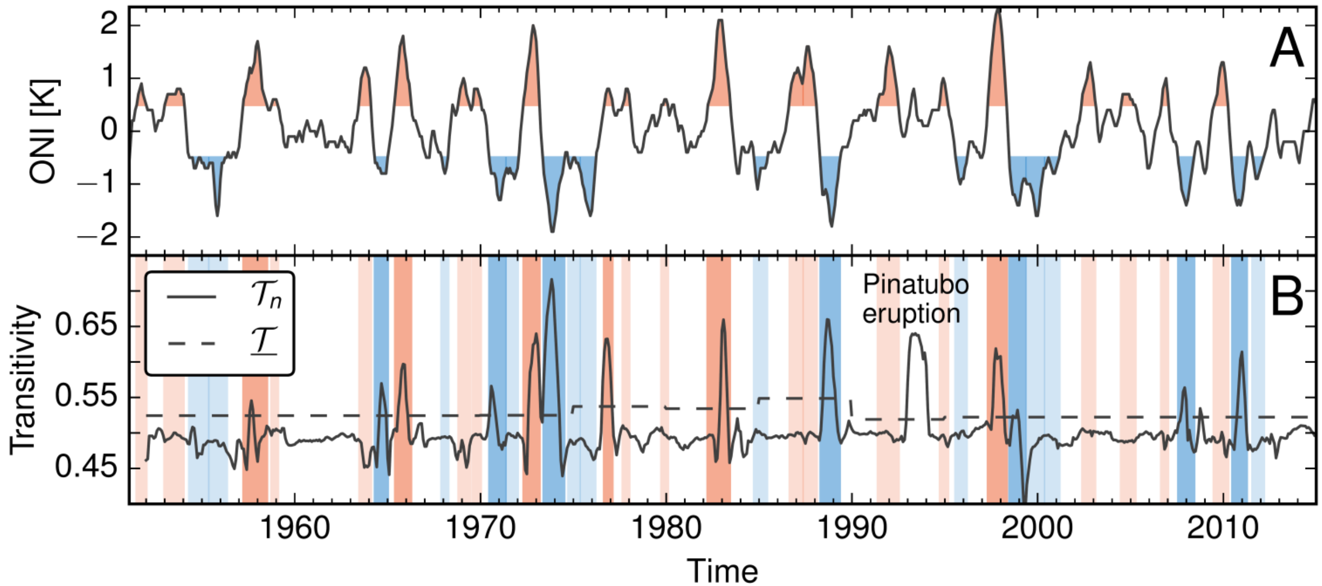


Heitzig et al., 2011

The pyunicorn examples

Evolving climate networks

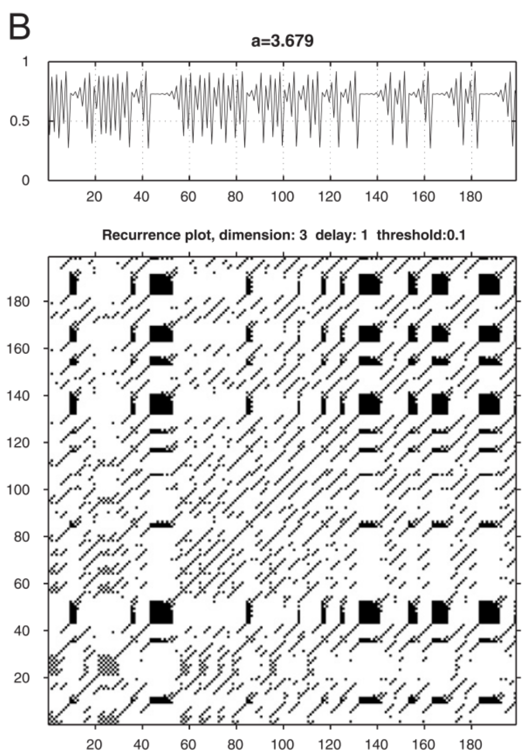
Evolving climate networks consist of a sequence on networks from running window cross-correlation. A window *n* is characterized by its size *w* and offset *d* to the previous window. Here: d=365 days, d=30 days.

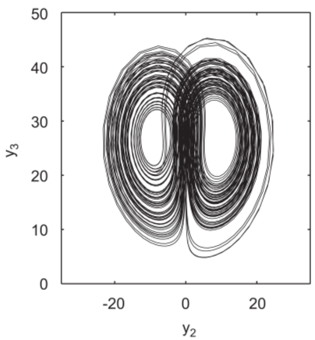


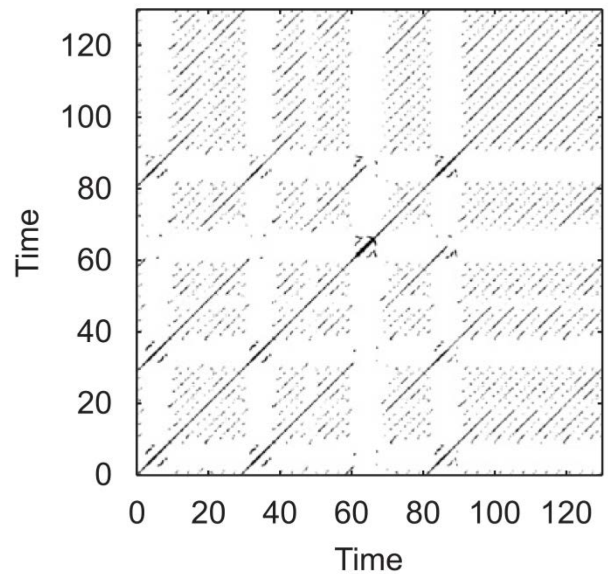
Wiedermann et al., 2016

The pyunicorn examples

Recurrence plots





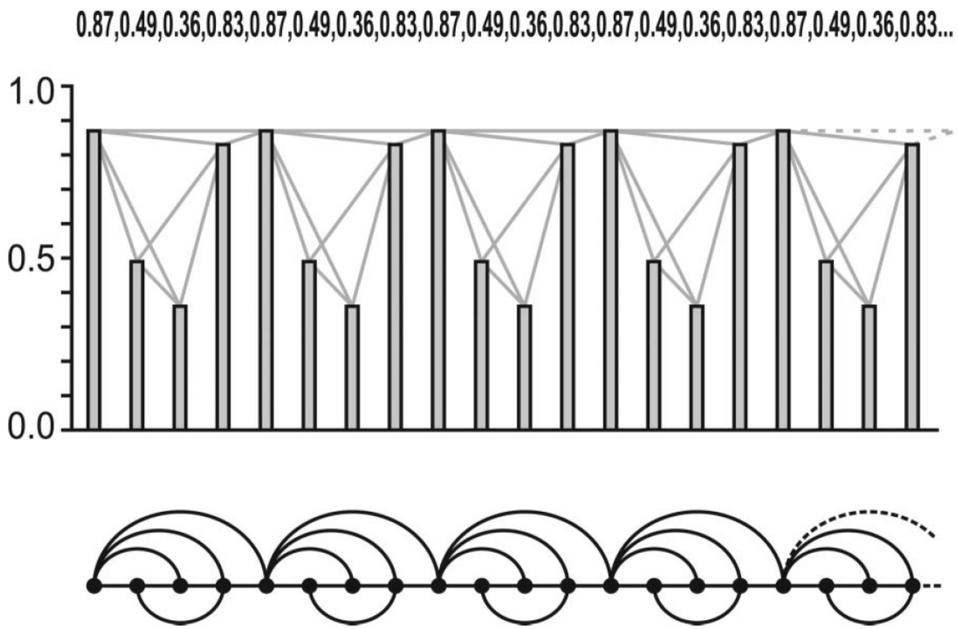


Marwan et al., 2005

Recurrence can be exploited to characterise the system's behaviour in phase space. A powerful tool for their visualisation and analysis called *recurrence plot.*

The pyunicorn examples

The visibility graph

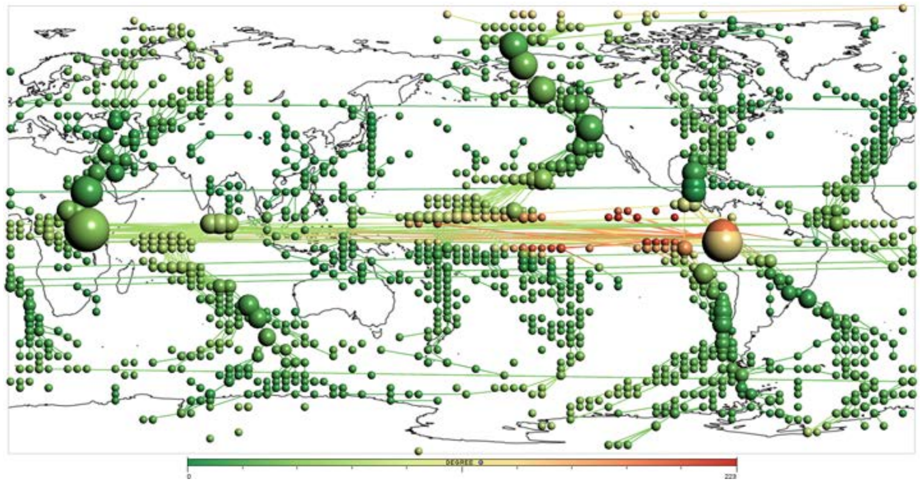


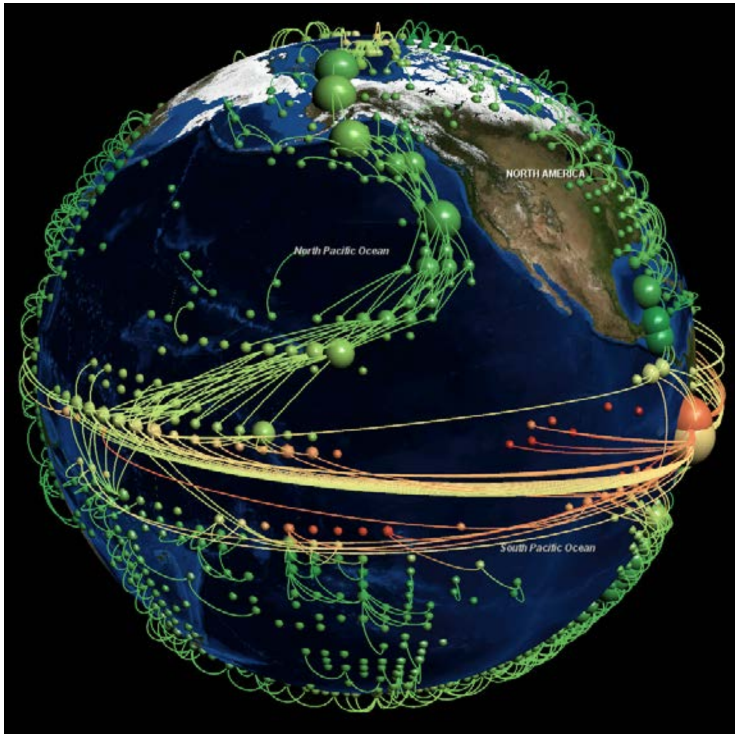
The *visibility algorithm*, converts a time series into a graph.

Lacasa et al., 2008

The pyunicorn examples

Visualization of climate networks





Tominski et al., 2011

The pyunicorn package

Documentation:

tinyurl.com/pyunicorn

install via pip

$ pip install pyunicorn

The pyunicorn examples

Start the python script:

```
#!/usr/bin/python

from pyunicorn.climate import ClimateData
from pyunicorn.core import Network
from pyunicorn.climate import ClimateNetwork
from netCDF4 import Dataset
import numpy as np
```

The pyunicorn examples

Get the data:

```
file_name = "../data_eraint/eraint_gph_dm_NH_197901_201312.nc"
# Name of the observable in the input file (obtain with ncdump -h <filename>)
observable_name = "z"
# The number of temporal sampling points per year (12 for monthly data)
time_cycle = 365
# The density of desired links in the corresponding climate network
link_density = 0.005
# Load the input data
data = ClimateData.Load(file_name=file_name, observable_name=observable_name,
                time_cycle=time_cycle, latitude_name="latitude",
                longitude_name="longitude")
#  remove the annual cycle
anomaly_all = data.anomaly()
n_times, n_index = anomaly.shape
```

The pyunicorn examples

Get an adjacency matrix:

```
# Compute the pearson correlation coefficient at zero lag and
# obtain theeshold T above which grid points are treated as connected.
pearson_correlation = corrcoef(anomaly.T)
T = percentile(pearson_correlation, (1 - link_density) * 100)

# Obtain the network's adjacency matrix from thresholding and set the diagonal
# line to zero since nodes are not allowed to be linked with itself
adjacency = pearson_correlation > T
for i in range(n_index):
    adjacency[i, i] = 0
```

The pyunicorn examples

Get latitudinal note weights:

```
# First, obtain the latitudinal positions of all grid points
latitudes = data.grid.lat_sequence()

# calculate latitudinal node weights
radian = latitudes* math.pi / 180
node_weights = map(math.cos, radian)
```

The pyunicorn examples

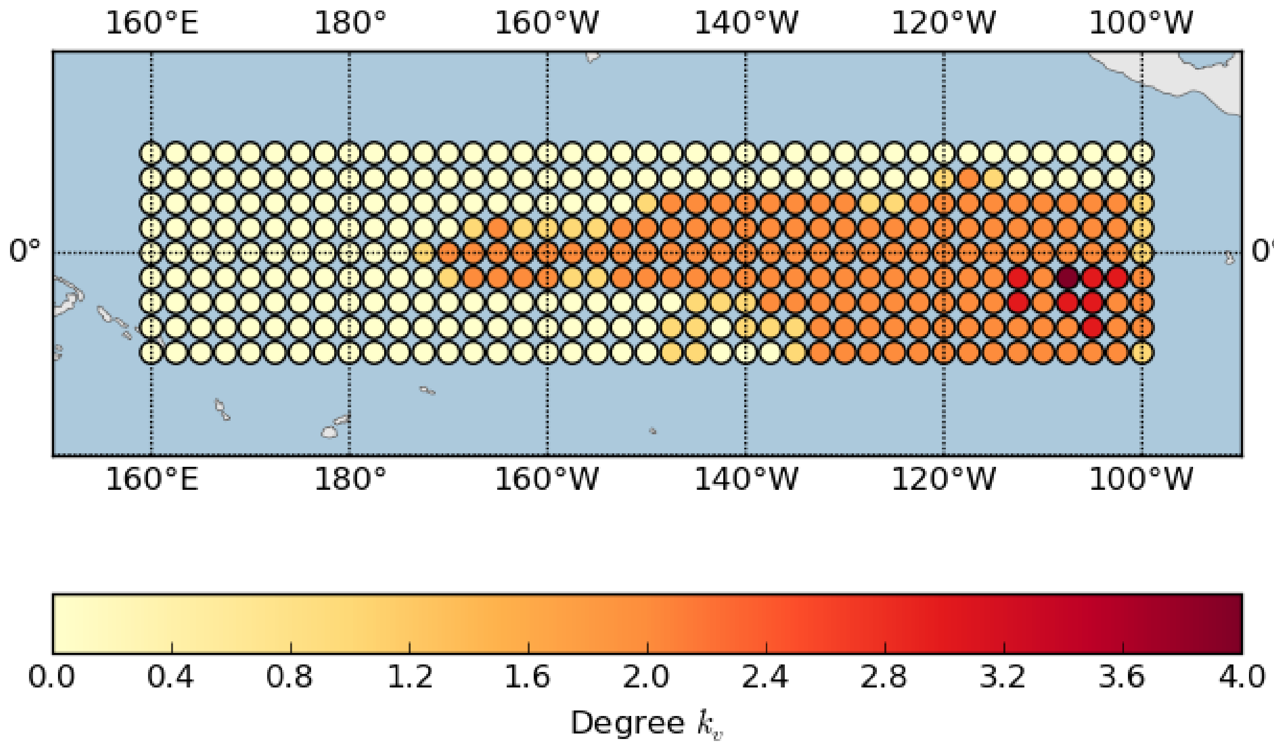
Get a note weighted climate network:

```
# get a climate network
net = Network(adjacency,node_weights=node_weights)
transi_all = net.nsi_transitivity()

# get degree
degree = net.degree()
degree_all = degree.reshape(lat_index,lon_index)
# get the nsi degree, with node weights
nsi_degree = net.nsi_degree()
nsi_degree_all = nsi_degree.reshape(lat_index,lon_index)
```

The pyunicorn examples

Plot with Basemap:



Fin

Thank you for your attention!

JASMIN

First steps:

- exec ssh-agent $SHELL
- ssh-add ~/.ssh/id_rsa_jasmin
  - passwort
- ssh –A –X *username*@jasmin_login1.ceda.ac.uk
- ssh –X jasmin-sci1
- cd /group_workspaces/jasmin2/qboi

JASMIN

python and pyunicorn

- cd
- virtualenv /home/users/*username*/my_python
- source /home/users/username/my_python/bin/activate
    - ➢ write the comment above into .bashrc
- pip install weave
- pip install pyunicorn
- pip install python-igraph

JASMIN

python and tigramite

- cd
- get tigramite-master.zip from https://github.com/jakobrunge/tigramite or from me
- unzip tigramite-master.zip
- cd tigramite-master
- python setup.py install
- git clone git://github.com/statsmodels/statsmodels.git

JASMIN

R

- R
- install.packages(“maps”)
- install.packages(“mapdata”)
- install.packages(“RNetCDF”)
- quit()
- n

JASMIN

else

(Windows: If you have installed Xming)

- open a script -> vim, geany, emacs, ...
- open a pdf -> xdg-open, ...

QBOi data at /group_workspaces/jasmin2/qboi