Description for Source Codes and Data

Ri-Qi Su,[1] Wen-Xu Wang,[2] Xiao Wang,[3] and Ying-Cheng Lai[1,*]

[1]*School of Electrical, Computer, and Energy Engineering,*
Arizona State University, Tempe, AZ 85287, USA

[2]*Department of Systems Science, School of Management and Center for Complexity Research,*
Beijing Normal University, Beijing 100875, China

[3]*School of Biological and Health Systems Engineering,*
Arizona State University, Tempe, AZ 85287, USA

*Electronic address: `Ying-Cheng.Lai@asu.edu`

To use the source codes, one need to have **gcc** complier and **Matlab** installed on a linux/unix based operator system. The test environment is Ubuntu *14.04* with **gcc** version *4.9.2* and **Matlab** version *8.4.0.150421*, on a personal PC powered by i7-2700 CPU and equipped with 32G memories.

Before simulation, edit the Bash script **simulate.sh** and specify the actual **Matlab** instalation path in Line 12, together with the predefined parameters from Line 15 to Line 21. Within these parameters, **Stamp** refers to time stamp and is used to specify different simulation realizations. **Nodes** and **Degree** refer to the network size and degree, respectively, for the generated scale-free networks. **DataDef** specifies the total sampling data points collected from the generated time series. **NmDef** is the list of R_m, which is the required measurement amount normalized by the number of total unknown parameters, and **EpsDef** is the list of ϵ, or the tolerance for compressive sensing algorithm (see Method for detail description.) Both of **NmDef** and **EpsDef** have to be written in the vector format defined in **Matlab**. The **RepTimeDef** defines the independent realizations in network reconstruction.

Then one may open a terminal and change to the directory where the files locate, and change the permission for **simulate.sh**:

chmod +x ./simulate.sh

Now one can execute the Bash script to perform simulation using:

./simulate.sh

The script will automatically generate a geospatial scale free network and simulate time series for the coupled dynamics on it. It will also generate the **Matlab** code to reconstruct networks and geospatial information from simulated time series, store the reconstruction results in **Matlab** data storage format (***.mat** file) and plot results by calling **Matlab**.

The required source codes for this script and their functionalities are listed as follows.

1. **dataioV1.h** is a header file written by Dr. Xuan Ni (see http://chaos34.eas.asu.edu/) and revised by Dr. Riqi Su. It provides functions in C programming environment to read and write **Matlab** data storage format, to realize efficient data structure for matrix and vector,

and to generate high quality random variables.

2. **BANet.cpp** was used to generate specific scale free random networks of given network sizes and average degrees.

3. **DDEv3.cpp** was used to generate time series for Rossler oscillators with delayed coupled on given scale free networks. It will return a **TimeSer*.mat** file.

4. **l1-magic** codes located in directory **/Optimization**. The source codes were previously downloaded from http://users.ece.gatech.edu/justin/l1magic/ and they are used here to solve the sparse regulation problem.

5. **localize2d.m** was provided by A. H. Sayed et al., in Network-based Wireless Location, *IEEE Signal Processing Magazine*, Vol. 24, No. 4, 2005. It is called by **triLocation.m** to solve the triangle localization problem with given distance matrix.

6. **matlab_BANet.m** is **Matlab** code to reconstruct geospatial network from time series and was written by the authors.

All the simulation results presented in this paper are given as ***.mat** files. In the final stored ***.mat** file, there are several importance variables. The **StoreCoup** stores W_{nz}, which is the avarage normalized errors for reconstructed coupling weights of the existed links. The **StoreDelay** stores D_{nz}, the average normalized errors for coupling delays of the existed links. And **zStoreCoup** and **zStoreDelay** represent W_z and D_z, respectively, which are the average of absolute error for the weights or delays associated with non-existed links. Please refer to the Method section and Figure 3 for detailed discussion.