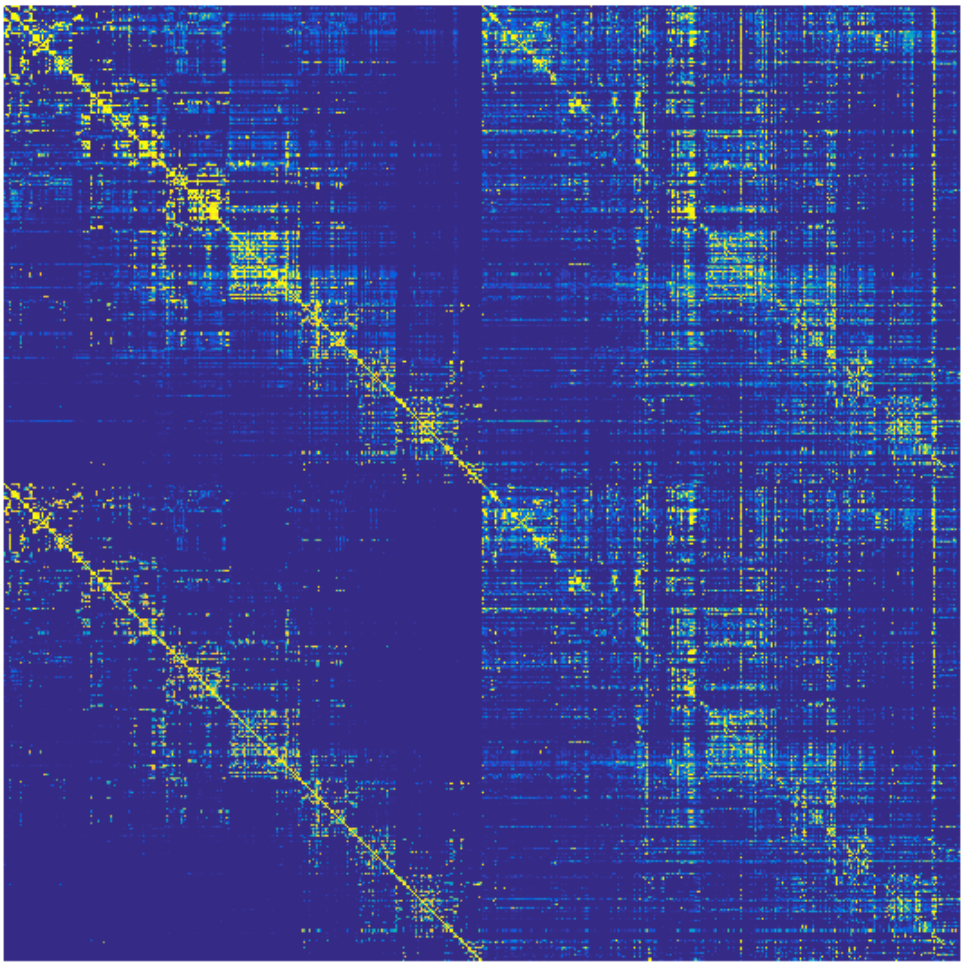
TooCANN: Toolbox for Construction and Analysis of Neural Networks



T. Bergmans, under supervision of T. Celikel
Department of Neurophysiology, Donders Institute, Radboud University, Nijmegen, The Netherlands

Contents

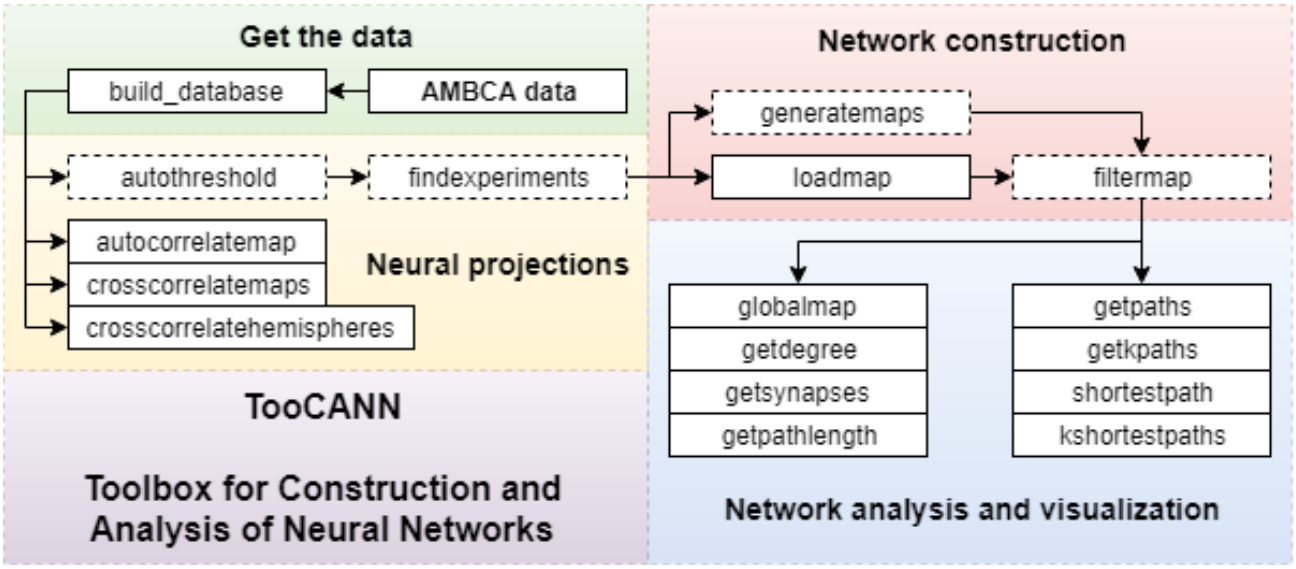
1. About

This toolbox can be used for network construction from neural projections. It is designed to work with data from the Allen Mouse Brain Connectivity Atlas (AMBCA), which is a database featuring quantifications of targeted monosynaptic projections throughout the mouse brain. These projections are extracted from imaging experiments with fluorescent protein expressing viral tracers injected in anatomically defined structures. At the time of writing the AMBCA consists of 2918 experiments. More information can be found on the website:
http://connectivity.brain-map.org/

Using this toolbox the monosynaptic projections from single experiments can be combined into polysynaptic networks. These networks represent anatomical networks in the mouse brain and can be further investigated with graph theoretical tools and tools for visualization.

The toolbox is written in MATLAB and uses the data from the AMBCA. There are functions available for importing, processing and visualizing individual experiments, as well as combining the experiments into networks, and functions for quantitative analysis of networks.

2. General workflow and files organization



Brief description of workflow and files organization

Building the database

After downloading this toolbox consists only of its functions and the file structures.mat. The function **build_database.m** can be used to automatically download and import the entire Mouse Brain Connectivity Atlas. When this function is executed, experiment metadata is downloaded from the ABA website and imported into MATLAB. Then the experiments are downloaded and imported, one by one, in JSON format. Using the search function on the AMBCA website a subset of experiments can be selected and stored in a custom experiments.csv file. If this file is present the build_database function will only download the specified subset.

File organization

The AMBCA data is stored in the 'Data' directory. The experiment metadata, structures list and list of network nodes are stored in the 'Structures' directory. This organization is used by many functions that depend on the metadata or experiment files.

Filtering and visualizing experiments

Individual experiment files can be filtered from noise using **autothreshold.m**. The functions **autocorrelatemap.m** and **crosscorrelatemaps.m** create diagrams in which general statistics for individual experiments are shown.

Creating networks

The function **loadmap.m** compiles experiment files into bilateral adjacency matrices. In an adjacency matrix, the value on position (i,j) represents the projection strength from node i to node j (sometimes called adjacency, correlation or connection). These values are determined by projection Density, Energy, Volume or Intensity. Weak connections can be filtered from the network using **filtermap.m**.

The function **generatemaps.m** loops over parameter space generating many networks with different properties; varying thresholds and different projections. It also calculates degree of separation matrices for every network.

Analyzing networks

A path between a node i and j is a list of edges, starting with an edge from i to the second node and ending with an edge ending at j. The length of a path is calculated by taking the element wise inverse of the projections matrix and summing these values for all edges that are in the path. This way, the higher the correlation between two nodes, the lower the pathlength is. Dijkstra's algorithm makes it possible to calculate the shortest path between two nodes in a network. Yen's algorithm can be used to calculate the K shortest paths, where K is some finite positive value.

The function **getkpaths.m** calculates the K shortest paths between any combination of two nodes, and stores the paths and path lengths in a matrix. The function **getsynapses.m** does the same, except it returns the number of synapses (or number of edges, which is the same) for any two nodes. This is done by using the Dijkstra algorithm on a binarized network. The result is the minimum number of edges required to get from one node to another. The function **getpathlength.m** gives the length of a path in a particular network.

3. List of functions

For a detailed description and syntax of each function, type 'help <functionname>' in MATLAB.

Build

- build_database — Download Allen Brain Atlas data and metadata files

Experiments

- autothreshold — Filter experiments by setting a threshold
- findexperiments — Find all experiments that satisfy a certain condition
- autocorrelatemap — Create figure with overview of experiment details
- crosscorrelatemaps — Create figure with comparative overview of two experiments
- crosscorrelatehemispheres — Create figure with comparison of cross-hemispheric projections

Networks

- loadmap — Compile experiment files into one network representation
- generatemaps — Calculate maps based on different projections
- filtermap — Filter weakest connections from map
- trimmap — Remove disconnected nodes from map

Network analysis

- shortestpath — Calculate shortest path between pair of nodes

- getpaths — Calculate shortest path for all node pairs
- kshortestpaths — Calculate K shortest paths between pair of nodes
- getkpaths — Calculate K shortest paths for all node pairs
- getsynapses — Calculate minimum number of synapses (edges) between nodes
- getpathlength — Get pathlength of weighted path in network
- globalmap — Compute distribution of map into larger anatomical regions
- getdegree — Return node degree (number of in- or out-connections)