Using The Code

# TLDR

Make sure you have Python and NumPy. Then, in the command line, enter

python verification.py

# Introduction

Greetings! This is a brief guide on how to use the code Jack and I have been working on for the last month. You’ll notice that there are many class files: Package, Container, Experiment, and Ensemble. To understand the first two classes is good, but at this point in time, you only need Experiment.

# Prerequisites

To run the code, you will need Python. You will also need the following packages (1) NumPy and (2) MatPlotLib.

We have been running from the command line.

# Overview

Code used in Doctors Hao and Graham’s “Creative Destruction” relied on vectors and matrices. We replace scalars with Packages and vectors with Containers.

An Experiment object should be thought of as one trial. It is given a csv containing an adjacency matrix, the load parameter, the time horizon, and a choice for sending rule. If choice is 0, then an unbiased random walk is used. If choice is 1, then information spreading is used.

The Ensemble class creates a population of random graphs for normalization procedures. Currently, it is not used.

# Packages and Containers

A package is a length two array. The first component is an array containing *tags*, unique identifiers assigned to each message upon birth. The second component is a 2-D array containing message *histories*, a record of where a given copy of a message has been. The first tag corresponds to the first history, the second to the second, and so on.

This convoluted structure allows us to neatly handle the presence of multiple messages at a node.

These classes are auxiliary to Experiment.

# Experiment

This class is the user’s primary concern.

These commands are meant for a text editor, not the command line.

An Experiment is initialized as follows:

a = Experiment(“adj\_mat\_file.csv”, load, time\_horizon, choice)

load and time\_horizon are integers. If choice is 0, then an unbiased random walk is used. If choice is 1, then information spreading is used.

To simulate activity up to time\_horizon, the command is

a.execute()

Experiment has many (hopefully descriptively named) output methods. All such methods take a file name and write the results to that file. The two of interest to you are Experiment.write\_verif\_output\_1 and Experiment.write\_verif\_output\_2.