**Content of the Repo**

***Files*** All codes are commented with descriptions, reference and Usage.

*Python*

1. 00\_download\_scrape\_openstack\_git.py
2. 01\_code\_embeddedness\_developers.py
3. 02\_code\_diversity\_network\_developers.py
4. 03\_spatio\_temporal\_developer\_network.py
5. 03\_create\_monte\_carlo\_spatio\_temporal\_network.py
6. 04\_code\_complexity\_motifs.py
7. 05\_create\_DSM\_motif\_counts\_propagation\_cost.py
8. algo\_community\_detection\_cartography.py

*R*

1. 03\_ERGM\_cumulative\_spatio-temporal\_networks.R
2. 03\_mediation\_monte\_carlo\_method.R
3. 03\_spatio-temporal\_networks\_descriptives.R
4. R\_script.sh

*Stata*

1. imputation\_code\_DV\_ratio.do
2. code\_diversity\_interactions.do
3. Idea03\_code\_mediation.do

***Data***

(Due to excessive file size of some of the data, we have included only a part of the full sample.

You may run the codes to generate the full data in your local drive)

1. **00\_data\_from\_git**/status\_q\_merged\_pages\_html/status\_q\_merged\_800.html
2. **00\_data\_from\_git**/subject\_pages\_html/subject\_page\_nova\_4743.html
3. **00\_data\_from\_git**/gitweb\_links\_html\_commits/commits\_2766\_\_01e6f7575a3c75bd73e297f3d9d003292e0a0e1e.xhtml
4. **00\_data\_from\_git**/github\_archives/nova-eab5851b0b55c4230cc11460f9efc6b617ae2e68.zip
5. **00\_data\_from\_git**/data\_commits\_developers\_from\_html\_text/openstack\_NOVA\_codes\_lines\_added\_comitter\_author\_code\_details.txt
6. **00\_data\_from\_git**/data\_commits\_developers\_from\_html\_text/openstack\_NOVA\_codes\_lines\_removed\_comitter\_author\_code\_details.txt
7. **01\_network\_developers**/02\_processd\_all\_firms\_data\_\_committer\_network\_from\_common\_feature\_modified.gml
8. **03\_spatio\_temporal**/ERGM\_cumulative\_networks/nova\_spatial\_temporal\_network\_cumulative\_2012\_2013.gml
9. **03\_spatio\_temporal**/networks\_per\_year\_mediation/nova\_committer\_spatial\_temporal\_network\_weights\_2012.gml
10. **03\_spatio\_temporal**/Null\_model\_monte\_carlo\_example/sim\_\_100.gml
11. **04\_network\_of\_files**/network\_of\_files\_example/nova-eab5851b0b55c4230cc11460f9efc6b617ae2e68.gml
12. **04\_network\_of\_files**/dsm\_examples\_\_2012.gephi

**Comment:** The variables and parameters calculated from this data are commented in the actual source files (python,R, STATA). A potential codebook is considered.

**Analyses completed**

1. **Download html/xhtml and *Nova* repositories**

**File:** 00\_download\_scrape\_openstack\_git.py

* + 1. *download\_status\_q\_merged\_pages\_html*

Downloads the pages from “https://review.openstack.org/#/q/status:merged,”

* + 1. *download\_subject\_pages\_html*

Read the html pages and save in a text formatted table

* + 1. *download\_status\_pages*

Download the status pages from the data of merged pages

* + 1. *download\_gitweb\_pages*

From the subject pages, get the commit ID and save the commit pages. We get the committer ID from these commit pages.

* + 1. *extract\_from\_gitweb\_commit\_lines\_code\_info*

From the commit pages get the information of commiter, lines of code added/removed and codenames and commit ID

* + 1. *download\_codes\_github*

Download the zip folders of each commit in Nova

Same code could be used for other OpenStack packages

1. **Manuscript:** Central or in a Nucleus? Joint Problem Solving Relationships and Individual Knowledge Creation in Open Collaboration (*Under Review 2017*)

**File:** 01\_code\_embeddedness\_developers.py

* + 1. *create\_edge\_list*

Creates an edgelist from the input file containing Developer data and feature ID (obtained from Albert Armisen’s dataset)

* + 1. *network\_create\_gml*

Generate network in gml format and add attributes

* + 1. *network\_measures*

Algorithms for degree, betweenness centrality and k-shell index

* + 1. *network\_attribute\_node\_type*

Add node attributes – types of developers

* + 1. *create\_blueprint\_bug\_feauture\_committer*

Create dataset for exploration-exploitation analysis

* + 1. *neighbor\_attribute\_node\_type*

Add neighbor’s contribution in terms of lines of code added into the gml file and save in table format as well

* + 1. *tenure\_committer*

Get the tenure of a developer

* + 1. *create\_output\_from\_gml*

Convert gml data to table data for regression analysis

* + 1. *plot\_network\_nx*

Plot developers’ network in networkx module

1. **Manuscript:** Temporal and Spatial Distance and Knowledge Contribution (*Under Preparation*)

**File:** 03\_spatio\_temporal\_developer\_network.py

* + 1. *countDuplicatesInList*

Counts the number of times a tuple appears in a list. This function is used to count the weight of an edge during formation of network links.

* + 1. *network\_measures*

Network parameters – degree, betweenness centrality, *k-*shell

* + 1. *create\_spatial\_temporal\_network\_of\_developers*

Create the spatio-temporal network of developers separated in time and space. We generate the network from the commit time data of developers.

* + 1. *measure\_developer\_attributes*

Get the node attributes – lines of code added/removed by a developer

Average code complexity of files on which developers work on

* + 1. *spatio\_temporal\_network\_neighbor\_ego\_effect*

Get the neighbor and Ego effect on the developers

**File:** 03\_create\_monte\_carlo\_spatio\_temporal\_network.py

1. *null\_model\_shuffle\_edge\_weights*

Here we randomly shuffle the edge weights of the network, keeping the structure of the network fixed. So, the neighbors of a developer are the same, but the weights are randomly shuffled. Repeat 100 times.

1. *gen\_null\_network\_from\_raw\_data*

Create random samples of the commit data using Monte Carlo sampling. Here we randomly shuffle from the original commit data – A developer commits at time t1 and t2 on file *f1* at lines *L1-15*, would commit randomly at times t10, t25 on file *f4* at lines *L18-45*. Repeat these 100 times.

1. *spatio\_temporal\_network\_neighbor\_ego\_effects*

Get the neighbor and Ego effect on the developers using the networks obtained from Monte Carlo sampling

**File:** 03\_mediation\_monte\_carlo.R

(*R* script to run mediation model on networks obtained from Monte Carlo sampling and check how many times mediating role of spatial distance exist)

**File:** 03\_spatio-temporal\_networks\_descriptives.R

(*R* script to run descriptives of the spatio-temporal network of developers)

1. **Manuscript:** Exponential Random Graph Model (ERGM) in explaining the emergence of problem-centric knowledge collaboration (*Under Preparation*)

**File:** 03\_ERGM\_cumulative\_spatio-temporal\_networks.R

(*R* script to run ERGM on the spatio-temporal networks of developers )

1. **Manuscript:** Motifs and modularity in complex systems (Under Preparation)

**File:** 04\_code\_complexity\_motifs.py

* + 1. *countDuplicatesInList*

Counts the number of times a tuple appears in a list. This function is used to count the weight of an edge during formation of network links

* + 1. *network\_measures*

Algorithms for degree, betweenness centrality and k-shell index

* + 1. *get\_code\_complexity\_measures*

From the zip folder of nova get the code complexity measures – CC, HV, MI

* + 1. *get\_import\_dsm*

Algorithm to generate Design Structure Matrix from the folders of each commit

* + 1. *gen\_network\_function\_reuse*

Algorithm to create the network of files from functional reuse. Two files are connected if the use the same function. For example, two files using *mean* function of *numpy* gets connected by a link.

* + 1. *gen\_network\_architectural\_reuse*

Two files are connected if they import the same package. Two files importing *numpy* module in Python gets connected.

1. **Manuscript:** Motifs and modularity in complex systems (Under Preparation)

**File:** 05\_create\_DSM\_motif\_counts\_propagation\_cost.py

* + 1. *subgraph\_pattern*

Motif detection algorithm. Count all k-subgraphs in the network. Which of those subgraphs are isomorphic (i.e. topologically equivalent) and count only once every such isomorphic groups.

* + 1. *check\_cycles*

Count all cyclical subgraphs in the network

* + 1. *check\_Motifs*

Count all k-subgraphs in the network.

* + 1. *write\_network\_attr\_gml*

Node and edge attributes of each file-file dependency network (DSM)

* + 1. *gen\_visibility\_matrix\_dsm\_adjacency*

Generate the visibility matrix of DSM and evaluate the

propagation cost as defined by Baldwin

1. **Algorithm:** Community detection and Cartographic measures

**File:** algo\_commuity\_detection\_cartography.py

* + 1. *rolescartography*

Given a community structure, this function classifies the node into various roles

* + 1. *communityroledetectionInfomap*

Infomap algorithm to detect communities

* + 1. *findCommunitiesInfomap*

Annotates nodes with 'community' id and return number of communities found.

1. **Manuscript:** The role of collaboration diversity on knowledge contribution (Under Preparation)

**File:** 02\_code\_diversity\_network\_developers.py

* + 1. *histogram\_dates*

Create the bin of 2months/4months

* + 1. *bin\_by\_bin\_edges\_codes\_contribution*

Code contribution of developers in each bin

* + 1. *create\_diversity\_developers*

Herfindahl index, Insularity index and similarity index of developers: diversity measure