Article

A Matlab program for finding shortest paths in the network: Application in the tumor pathway

WenJun Zhang
School of Life Sciences, Sun Yat-sen University, Guangzhou 510275, China; International Academy of Ecology and Environmental Sciences, Hong Kong
E-mail: zhwj@mail.sysu.edu.cn, wjzhang@iaees.org

Received 16 November 2015; *Accepted 18 December 2015*; *Published online 1 March 2016*

Abstract
The Floyd algorithm is used to find shortest paths in a graph or network. In present article I present full Matlab codes of the Floyd algorithm for using in the studies of network pharmacology. As an example, it is used to find shortest paths in a tumor pathway.

Keywords network; shortest path; Matlab; Floyd algorithm.

Network Pharmacology
ISSN 2415-1084
URL: http://www.iaees.org/publications/journals/np/online-version.asp
RSS: http://www.iaees.org/publications/journals/np/rss.xml
E-mail: networkpharmacology@iaees.org
Editor-in-Chief: WenJun Zhang
Publisher: International Academy of Ecology and Environmental Sciences

1 Introduction

The shortest path problem dedicates to find a path with minimal weight sum (i.e., shortest path) in a weighted graph or network (Zhang, 2012). The shortest path refers to the one between two given nodes, or the one between a given node to every nodes. The weights in weighted graph or network are cost, flux, distance, etc. In the present study I present full Matlab codes of the Floyd algorithm for using in the studies of network pharmacology.

2 Floyd Algorithm

In the Floyd algorithm (1962), each time insert a node, and compare the weight sum of the path between any two nodes with the weight sum of the path going through the two nodes and inserted node, if the former is larger, then replace the former path with the path having inserted the node (Zhang, 2012). Suppose the graph or network X has v nodes, and the weight matrix is $d=(d_{ij})$, where $d_{ij}=1$ for unweighted network and $d_{ij}=w_{ij}$ for weighted network (w_{ij} is the weight for the link v_i to v_j), if v_i and v_j are adjacent, and $d_{ij}=0$, if v_i and v_j are not adjacent; $i, j=1,2,\ldots, v$. The algorithm is (Zhang, 2012)

(1) Let $k=1$.
(2) Let $i=1$.

(3) Calculate

$$d_{ij}=\min(d_{ij}, d_{ik}+d_{kj}), \quad j=1,2,...,v.$$

(4) Let $i=i+1$, if $i\leqslant v$, then return to (3).

(5) Let $k=k+1$, if $k\leqslant v$, then return to (2), or else terminate the calculation.

The following are Matlab codes, Floyd.m, for Floyd algorithm:

```
d=input('Input the file name of adjacency matrix of the weighted network (e.g., adj.txt, adj.xls, etc. Adjacency matrix is
d=(dij)v*v, where v is the number of nodes in the network. dij=1 for unweighted network and dij=wij for weighted network (wij
is the weight for the link vi to vj), if vi and vj are adjacent, and dij=0, if vi and vj are not adjacent; i, j=1,2,…, m): ','s');
d=load(d);
% d: weighted adjacency matrix; distances: matrix of distances between different
% nodes; paths: string of paths and distances between any of two nodes
inf=1e+50;
v=size(d,1);
a=zeros(v);
b=zeros(1,v*(v-1)/2);
e=zeros(1,v*(v-1)/2);
h=zeros(1,v*(v-1)/2);
distances=zeros(v);
for i=1:v
for j=1:v
if ((d(i,j)==0) & (i~=j)) d(i,j)=inf; end
end; end
for i=1:v
for j=1:v
if (d(i,j)~=inf) a(i,j)=j; end
end; end
for i=1:v
for j=1:v
for k=1:v
c=d(j,i)+d(i,k);
if (c<d(j,k)) d(j,k)=c; a(j,k)=i; end
end; end; end
paths='';
for p=1:v
for q=1:v
if (p==q) continue; end
u=a(p,q);
m=1;
b(1)=u;
while (v>0)
m=m+1;
```

```
b(m)=a(b(m-1),q);
if (q==b(m)) break; end
if (b(m)==b(m-1)) break; end
if (m>v) break; end
end
n=1;
e(1)=u;
while (v>0)
n=n+1;
e(n)=a(p,e(n-1));
if (p==e(n)) break; end
if (e(n)==e(n-1)) break; end
if (n>v) break; end
end
for i=1:m+n-1
if (i==1) h(i)=p; end
if ((i<=n) & (i>1)) h(i)=e(n-i+1); end
if ((i>n) & (i<(m+n-1))) h(i)=b(i-n+1); end
if (i==(m+n-1)) h(i)=q; end
end
paths=strcat(paths,'Shortest path from ',num2str(p),' to ',num2str(q),':\n');
for i=1:m+n-1
if ((h(i)~=0) & (d(p,q)~=inf))
if ((h(i)==h(i+1)) & (i<m+n-1)) continue; end
if (i<m+n-1) paths=strcat(paths,num2str(h(i)),'->'); end
if (i>=m+n-1) paths=strcat(paths,num2str(h(i)),'\n'); end
end; end
if (d(p,q)~=inf) paths=strcat(paths,'Distance=',num2str(d(p,q)),'\n'); distances(p,q)=d(p,q); end
if (d(p,q)==inf) paths=strcat(paths,'No path','\n'); end
end; end
disp('Distances matrix')
distances
disp('Shortest paths')
fprintf(paths)
```

3 Application Case

Here I use the PPAR tumor pathways to find shortest paths in the network (Huang and Zhang, 2012; Li and Zhang, 2013). The adjacency matrix (unweighted network) of PPAR tumor pathways is as the following

0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	1	0	0
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Running the Matlab program, the distances matrix is as the following

0	2	4	9	5	1	8	2	7	3	6	9	5	8	9	7	10	6	10	5	9	8	8	7	6	4	6
2	0	4	9	5	1	8	2	7	3	6	9	5	8	9	7	10	6	10	5	9	8	8	7	6	4	6
4	4	0	5	3	3	4	2	3	1	2	5	1	4	5	3	6	2	6	3	5	4	4	3	4	2	2
9	9	5	0	8	8	1	7	2	6	3	8	4	7	8	6	9	5	9	6	8	7	7	6	7	7	5
5	5	3	8	0	4	7	3	6	2	5	6	4	5	4	4	6	3	5	2	4	3	3	2	1	1	3
1	1	3	8	4	0	7	1	6	2	5	8	4	7	8	6	9	5	9	4	8	7	7	6	5	3	5
8	8	4	1	7	7	0	6	1	5	2	7	3	6	7	5	8	4	8	5	7	6	6	5	6	6	4
2	2	2	7	3	1	6	0	5	1	4	7	3	6	7	5	8	4	8	3	7	6	6	5	4	2	4
7	7	3	2	6	6	1	5	0	4	1	6	2	5	6	4	7	3	7	4	6	5	5	4	5	5	3
3	3	1	6	2	2	5	1	4	0	3	6	2	5	6	4	7	3	7	2	6	5	5	4	3	1	3
6	6	2	3	5	5	2	4	1	3	0	5	1	4	5	3	6	2	6	3	5	4	4	3	4	4	2
9	9	5	8	6	8	7	7	6	6	5	0	4	1	6	2	1	3	2	4	3	5	4	4	5	5	5
5	5	1	4	4	4	3	3	2	2	1	4	0	3	4	2	5	1	5	2	4	3	3	2	3	3	1
8	8	4	7	5	7	6	6	5	5	4	1	3	0	5	1	2	2	3	3	4	4	4	3	4	4	4
9	9	5	8	4	8	7	7	6	6	5	6	4	5	0	4	6	3	5	4	4	1	3	2	3	5	5
7	7	3	6	4	6	5	5	4	4	3	2	2	1	4	0	3	1	4	2	4	3	3	2	3	3	3
10	10	6	9	6	9	8	8	7	7	6	1	5	2	6	3	0	4	1	5	2	5	3	4	5	6	6
6	6	2	5	3	5	4	4	3	3	2	3	1	2	3	1	4	0	4	1	3	2	2	1	2	2	2
10	10	6	9	5	9	8	8	7	7	6	2	5	3	5	4	1	4	0	5	1	4	2	3	4	6	6

5	5	3	6	2	4	5	3	4	2	3	4	2	3	4	2	5	1	5	0	4	3	3	2	1	1	1
9	9	5	8	4	8	7	7	6	6	5	3	4	4	4	4	2	3	1	4	0	3	1	2	3	5	5
8	8	4	7	3	7	6	6	5	5	4	5	3	4	1	3	5	2	4	3	3	0	2	1	2	4	4
8	8	4	7	3	7	6	6	5	5	4	4	3	4	3	3	3	2	2	3	1	2	0	1	2	4	4
7	7	3	6	2	6	5	5	4	4	3	4	2	3	2	2	4	1	3	2	2	1	1	0	1	3	3
6	6	4	7	1	5	6	4	5	3	4	5	3	4	3	3	5	2	4	1	3	2	2	1	0	2	2
4	4	2	7	1	3	6	2	5	1	4	5	3	4	5	3	6	2	6	1	5	4	4	3	2	0	2
6	6	2	5	3	5	4	4	3	3	2	5	1	4	5	3	6	2	6	1	5	4	4	3	2	2	0

Some of the shortest paths are achieved as follows

Shortest paths

Shortest path from1 to2:

1->6->2

Distance=2

Shortest path from1 to3:

1->6->8->10->3

Distance=4

Shortest path from1 to4:

1->6->8->10->13->11->9->7->4

Distance=9

Shortest path from1 to5:

1->6->8->10->26->5

Distance=5

Shortest path from1 to6:

1->6

Distance=1

Shortest path from1 to7:

1->6->8->10->13->11->9->7

Distance=8

Shortest path from1 to8:

1->6->8

Distance=2

Shortest path from1 to9:

1->6->8->10->13->11->9

Distance=7

Shortest path from1 to10:

1->6->8->10

Distance=3

Shortest path from1 to11:

1->6->8->10->13->11

Distance=6

Shortest path from1 to12:

1->6->8->10->13->18->16->14->12

Distance=9

Shortest path from1 to13:
1->6->8->10->3->13
Distance=5
Shortest path from1 to14:
1->6->8->10->13->18->16->14
Distance=8
Shortest path from1 to15:
1->6->8->10->13->18->24->22->15
Distance=9
Shortest path from1 to16:
1->6->8->10->13->18->16
Distance=7
Shortest path from1 to17:
1->6->8->10->13->18->16->14->12->17
Distance=10
Shortest path from1 to18:
1->6->8->10->13->18
Distance=6
Shortest path from1 to19:
1->6->8->10->13->18->24->23->21->19
Distance=10
Shortest path from1 to20:
1->6->8->10->26->20
Distance=5
Shortest path from1 to21:
1->6->8->10->13->18->24->23->21
Distance=9
Shortest path from1 to22:
1->6->8->10->13->18->24->22
Distance=8
Shortest path from1 to23:
1->6->8->10->13->18->24->23
Distance=8
Shortest path from1 to24:
1->6->8->10->13->18->24
Distance=7
Shortest path from1 to25:
1->6->8->10->26->5->25
Distance=6
Shortest path from1 to26:
1->6->8->10->26
Distance=4
Shortest path from1 to27:
1->6->8->10->13->27
Distance=6

Shortest path from2 to1:
2->6->1
Distance=2
Shortest path from2 to3:
2->6->8->10->3
Distance=4
Shortest path from2 to4:
2->6->8->10->13->11->9->7->4
Distance=9
Shortest path from2 to5:
2->6->8->10->26->5
Distance=5
Shortest path from2 to6:
2->6
Distance=1
Shortest path from2 to7:
2->6->8->10->13->11->9->7
Distance=8
Shortest path from2 to8:
2->6->8
Distance=2
Shortest path from2 to9:
2->6->8->10->13->11->9
Distance=7
Shortest path from2 to10:
2->6->8->10
Distance=3
Shortest path from2 to11:
2->6->8->10->13->11
Distance=6
Shortest path from2 to12:
2->6->8->10->13->18->16->14->12
Distance=9
Shortest path from2 to13:
2->6->8->10->3->13
Distance=5
Shortest path from2 to14:
2->6->8->10->13->18->16->14
Distance=8
Shortest path from2 to15:
2->6->8->10->13->18->24->22->15
Distance=9
Shortest path from2 to16:
2->6->8->10->13->18->16
Distance=7

Shortest path from2 to17:
2->6->8->10->13->18->16->14->12->17
Distance=10
Shortest path from2 to18:
2->6->8->10->13->18
Distance=6
Shortest path from2 to19:
2->6->8->10->13->18->24->23->21->19
Distance=10
Shortest path from2 to20:
2->6->8->10->26->20
Distance=5
Shortest path from2 to21:
2->6->8->10->13->18->24->23->21
Distance=9
Shortest path from2 to22:
2->6->8->10->13->18->24->22
Distance=8
Shortest path from2 to23:
2->6->8->10->13->18->24->23
Distance=8
Shortest path from2 to24:
2->6->8->10->13->18->24
Distance=7
Shortest path from2 to25:
2->6->8->10->26->5->25
Distance=6
Shortest path from2 to26:
2->6->8->10->26
Distance=4
Shortest path from2 to27:
2->6->8->10->13->27
Distance=6
Shortest path from3 to1:
3->10->8->6->1
Distance=4
Shortest path from3 to2:
3->10->8->6->2
Distance=4
Shortest path from3 to4:
3->13->11->9->7->4
Distance=5
Shortest path from3 to5:
3->10->26->5
Distance=3

Shortest path from3 to6:
3->10->8->6
Distance=3
Shortest path from3 to7:
3->13->11->9->7
Distance=4
Shortest path from3 to8:
3->10->8
Distance=2
Shortest path from3 to9:
3->13->11->9
Distance=3
Shortest path from3 to10:
3->10
Distance=1
Shortest path from3 to11:
3->13->11
Distance=2
Shortest path from3 to12:
3->13->18->16->14->12
Distance=5
Shortest path from3 to13:
3->13
Distance=1
Shortest path from3 to14:
3->13->18->16->14
Distance=4
Shortest path from3 to15:
3->13->18->24->22->15
Distance=5
Shortest path from3 to16:
3->13->18->16
Distance=3
Shortest path from3 to17:
3->13->18->16->14->12->17
Distance=6
Shortest path from3 to18:
3->13->18
Distance=2
Shortest path from3 to19:
3->13->18->24->23->21->19
Distance=6
Shortest path from3 to20:
3->13->18->20
Distance=3

Shortest path from3 to21:
3->13->18->24->23->21
Distance=5
Shortest path from3 to22:
3->13->18->24->22
Distance=4
Shortest path from3 to23:
3->13->18->24->23
Distance=4
Shortest path from3 to24:
3->13->18->24
Distance=3
Shortest path from3 to25:
3->13->18->20->25
Distance=4
Shortest path from3 to26:
3->10->26
Distance=2
Shortest path from3 to27:
3->13->27
Distance=2
Shortest path from4 to1:
4->7->9->11->13->10->8->6->1
Distance=9
Shortest path from4 to2:
4->7->9->11->13->10->8->6->2
Distance=9
Shortest path from4 to3:
4->7->9->11->13->3
Distance=5
Shortest path from4 to5:
4->7->9->11->13->18->20->25->5
Distance=8
Shortest path from4 to6:
4->7->9->11->13->10->8->6
Distance=8
Shortest path from4 to7:
4->7
Distance=1
Shortest path from4 to8:
4->7->9->11->13->10->8
Distance=7
Shortest path from4 to9:
4->7->9
Distance=2

Shortest path from4 to10:
4->7->9->11->13->3->10
Distance=6
Shortest path from4 to11:
4->7->9->11
Distance=3
Shortest path from4 to12:
4->7->9->11->13->18->16->14->12
Distance=8
Shortest path from4 to13:
4->7->9->11->13
Distance=4
Shortest path from4 to14:
4->7->9->11->13->18->16->14
Distance=7
Shortest path from4 to15:
4->7->9->11->13->18->24->22->15
Distance=8
Shortest path from4 to16:
4->7->9->11->13->18->16
Distance=6
Shortest path from4 to17:
4->7->9->11->13->18->16->14->12->17
Distance=9
Shortest path from4 to18:
4->7->9->11->13->18
Distance=5
Shortest path from4 to19:
4->7->9->11->13->18->24->23->21->19
Distance=9
Shortest path from4 to20:
4->7->9->11->13->18->20
Distance=6
Shortest path from4 to21:
4->7->9->11->13->18->24->23->21
Distance=8
Shortest path from4 to22:
4->7->9->11->13->18->24->22
Distance=7
Shortest path from4 to23:
4->7->9->11->13->18->24->23
Distance=7
Shortest path from4 to24:
4->7->9->11->13->18->24
Distance=6

Shortest path from4 to25:

4->7->9->11->13->18->20->25

Distance=7

Shortest path from4 to26:

4->7->9->11->13->10->26

References

Floyd RW. 1962. Algorithm 97: shortest path. Comm ACM, 5: 345

Huang JQ, Zhang WJ. 2012. Analysis on degree distribution of tumor signaling networks. Network Biology, 2(3): 95-109

Li JR, Zhang WJ. 2013. Identification of crucial metabolites/reactions in tumor signaling networks. Network Biology, 3(4): 121-132

Zhang WJ. 2012. Computational Ecology: Graphs, Networks and Agent-based Modeling. World Scientific, Singapore