# MSc Project IIS1 | BWFLnet_Water Analysis

**Prepared by: Yuanfei Wang**

**Date: 17 July 2022**

```
clc
clear
close all
```

## Model Setup and Initialisation

Initialisation of EPANET Matlab toolkit and loading of .net file.

```
epanet_path = 'E:\Program Files\MATLAB\R2021b\toolbox\epanet\EPANET-Matlab-Toolkit-master';
net_id = 'BWFLnet_MSc_2022_calibrated';
run( [epanet_path,'\start_toolkit'])
```

```
EPANET-MATLAB Toolkit Paths Loaded.
```

```
net = epanet([net_id,'.inp']);
```

```
EPANET version {20200} loaded (EMT version {v2.2.0}).
Loading File "BWFLnet_MSc_2022_calibrated.inp"...
Input File "BWFLnet_MSc_2022_calibrated.inp" loaded sucessfuly.
```

```
load("wq_data.mat");
load("R_CData.mat");
```

## Network Data Loading

Load general network data, noting that this network has already been hydraulically calibrated, thus we can directly perform the quality analysis.

```
% Save element count variables
nn = net.NodeJunctionCount;
n0 = net.NodeReservoirCount;
np = net.LinkCount; % count of pipes and valves
D = net.getLinkDiameter';

% Save node and link index vectors
Reservoir_Idx = net.getNodeReservoirIndex;
Junction_Idx = net.getNodeJunctionIndex;
Link_Idx = double(net.getLinkIndex);

% Specify hydraulic and quality time steps (in seconds)
net.setTimeQualityStep(15*60); % 15 minute quality time steps
net.setTimeHydraulicStep(15*60); % 15 minute hydraulic time steps
nt = net.getTimeSimulationDuration./3600; % number of time steps in hours
days = 7; % simulation duration in days
net.setTimeSimulationDuration(nt*3600*days/7); % Set the simulation duration by changing days
nt = net.getTimeSimulationDuration./3600; % get the new nt

% Nodes name and chlorine concentrations
Junction_All_Name = {'BW9','BW7','BW1(Res)','BW2','BW3','BW4(Res)','BW5','BW6','BW12'};
```

```
Junction_SelectIdx = net.getNodeIndex(wq_data.node_ids([1,4,7,9]));
Junction_Name = Junction_All_Name([1,4,7,9]);
Junction_HydrantIdx = net.getNodeIndex(wq_data.node_ids([2,5,8]));
Junction_Hydrant_Name = Junction_All_Name([2,5,8]);
Junction_Reservoir_Name = Junction_All_Name([6,3]);
cext = wq_data.chlorine([6,3],:);
c_nodes_HydrantObs = wq_data.chlorine([2,5,8],:);
c_nodes_Obs = wq_data.chlorine([1,4,7,9],:);
```

## Graph Creation Preparation

Use graph theory to plot network connectivity and spatial coordinates.

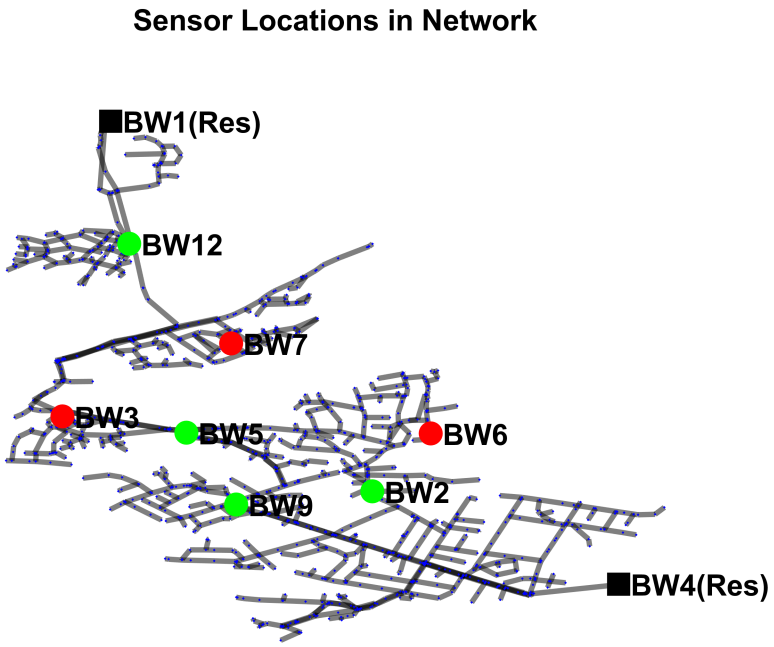
```
% Create A12 and A10 incidence matrices
LinkNodesList = net.getLinkNodesIndex;

A = zeros(np,nn+n0);
for k=1:np
    i = LinkNodesList(k,1);
    j = LinkNodesList(k,2);
    A(k,i) = -1;
    A(k,j) = 1;
end

A12 = A(:,Junction_Idx);
A10 = A(:,Reservoir_Idx);
A12 = sparse(A12);
A10 = sparse(A10);

% Obstain node XY and elevation information
XY = zeros(nn+n0,2);
XY(:,1) = net.getNodeCoordinates{1};
XY(:,2) = net.getNodeCoordinates{2};
elev = double(net.getNodeElevations(Junction_Idx))';

% Create adjacency matrix
A = [A12,A10];
AdjA = sparse(size(A,2),size(A,2));

for k = 1:size(A,1)
    node_in = find(A(k,:) == -1);
    node_out = find(A(k,:) == 1);
    AdjA(node_in,node_out) = 1;
    AdjA(node_out,node_in) = 1;
end
gr = graph(AdjA);
```

## Graph Plotting

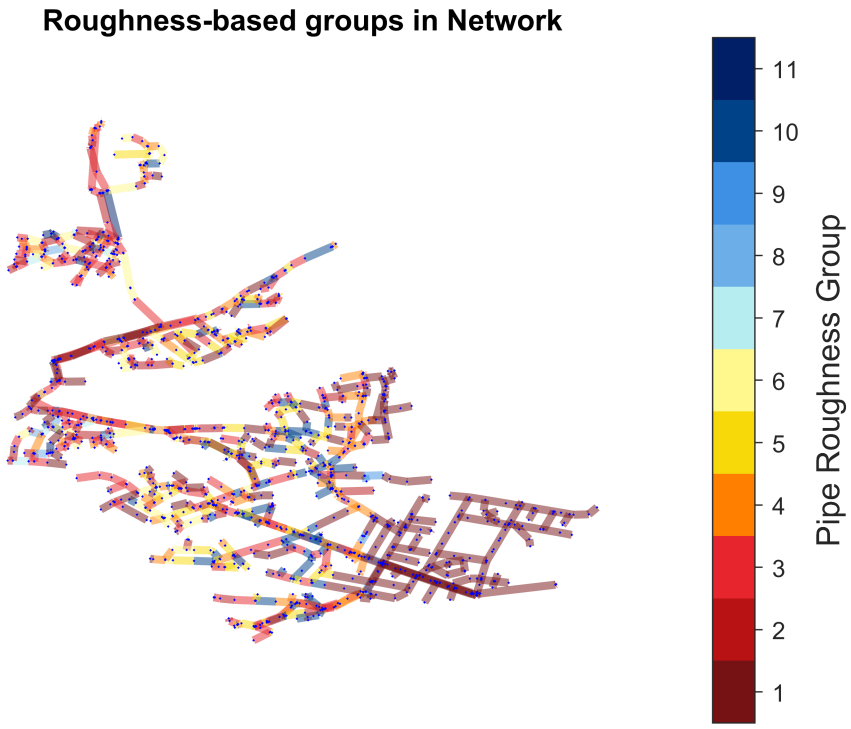
Setup general chemical simulation parameters.

```
% Plot the sensor location on the network map
```

```
figure
p1 = plot(gr);
p1.XData = XY(:,1);
p1.YData = XY(:,2);
p1.LineWidth = 2;
p1.EdgeColor = 'k';
p1.MarkerSize = 0.01;
p1.NodeColor = 'b';
p1.NodeLabel = '';
highlight(p1,Junction_SelectIdx,'NodeColor','green','MarkerSize',8);
labelnode(p1,Junction_SelectIdx,Junction_Name);
highlight(p1,Junction_HydrantIdx,'NodeColor','red','MarkerSize',8);
labelnode(p1,Junction_HydrantIdx,Junction_Hydrant_Name);
highlight(p1,Reservoir_Idx,'NodeColor','k','Marker','s','MarkerSize',10);
labelnode(p1,Reservoir_Idx,Junction_Reservoir_Name);
p1.NodeFontSize = 11;
p1.NodeLabelColor = 'k';
p1.NodeFontWeight = 'bold';
p1.EdgeFontSize = 11;
p1.EdgeLabelColor = 'k';
p1.EdgeFontWeight = 'bold';
% p1.EdgeCData = R_CData;
% hcb_R = colorbar;
% colormap('jet');
% hcb_R.Label.String = 'Pipe Roughness Group';
% hcb_R.Label.FontSize = 12;
axis('off')
title('Sensor Locations in Network')
```



```
% Plot the Roughness group on the network map
figure
p1 = plot(gr);
p1.XData = XY(:,1);
p1.YData = XY(:,2);
p1.LineWidth = 3;
p1.EdgeColor = 'k';
p1.MarkerSize = 1e-10;

p1.NodeColor = 'b';
p1.NodeLabel = '';
p1.NodeFontSize = 11;
p1.NodeLabelColor = 'k';
p1.NodeFontWeight = 'bold';
p1.EdgeFontSize = 11;
p1.EdgeLabelColor = 'k';
p1.EdgeFontWeight = 'bold';
p1.EdgeCData = R_CData;
cmap=[118 18 19;
     184 18 21;
     230 38 44;
     255 127 0;
     247 217 9;
     255 248 140;
     182 237 240;
     107 174 232;
     61 144 227;
     0 66 135;
     0 31 103];
cmap=cmap/255;
colormap(cmap);
hcb = colorbar;
hcb.TicksMode = 'manual';
set(hcb,'Tickdir','out');
caxis([1,11]);
set(hcb,'YTick',1+5/11:10/11:11-5/11);
set(hcb,'TickLabels',{'1','2','3','4','5','6','7','8','9','10','11'});
hcb.Label.String = 'Pipe Roughness Group';
hcb.Label.FontSize = 12;
axis('off')
title('Roughness-based groups in Network')
```




## Chlorine Simulation

Setup general chemical simulation parameters.

```
% Initialise EPANET simulation type
net.setQualityType('Chemical','mg/L');
net.setNodeSourceQuality(1:nn+n0,zeros(nn+n0,1)); % set all node source quality to zero

% % Link bulk reaction coefficients
% lambda = 0.5*ones(np,1); % units of days^-1
% net.setLinkBulkReactionCoeff(1:np, -lambda);
% net.setOptionsPipeBulkReactionOrder(1);

% Initial concentrations at nodes (mg/L)
c0 = zeros(nn+n0,nt);
net.setNodeInitialQuality(net.NodeIndex,c0);
base_cext = ones(n0,1);
pattern_cext = cext./(base_cext*ones(1,size(cext,2))); % extend vector over nt columns

% For loop to assign new patterns to source contrations at reservoirs and
% source type
for i=1:n0
    patternId = sprintf('Res_C_%d',i);
    net.addPattern(patternId,pattern_cext(i,:));
    net.setNodeSourcePatternIndex(net.NodeReservoirIndex(i),net.getPatternIndex(patternId));
    net.setNodeSourceQuality(net.NodeReservoirIndex(i),base_cext(i));
    net.setNodeSourceType(net.NodeReservoirIndex(i),'CONCEN');
```

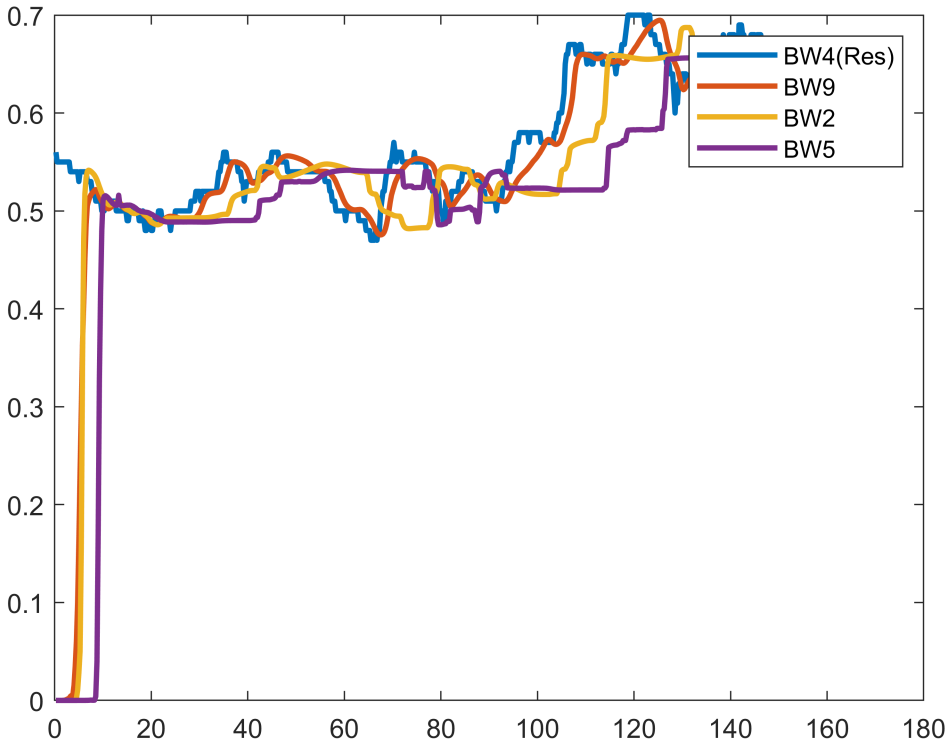
```
end
```

## Results from EPANET Simulation

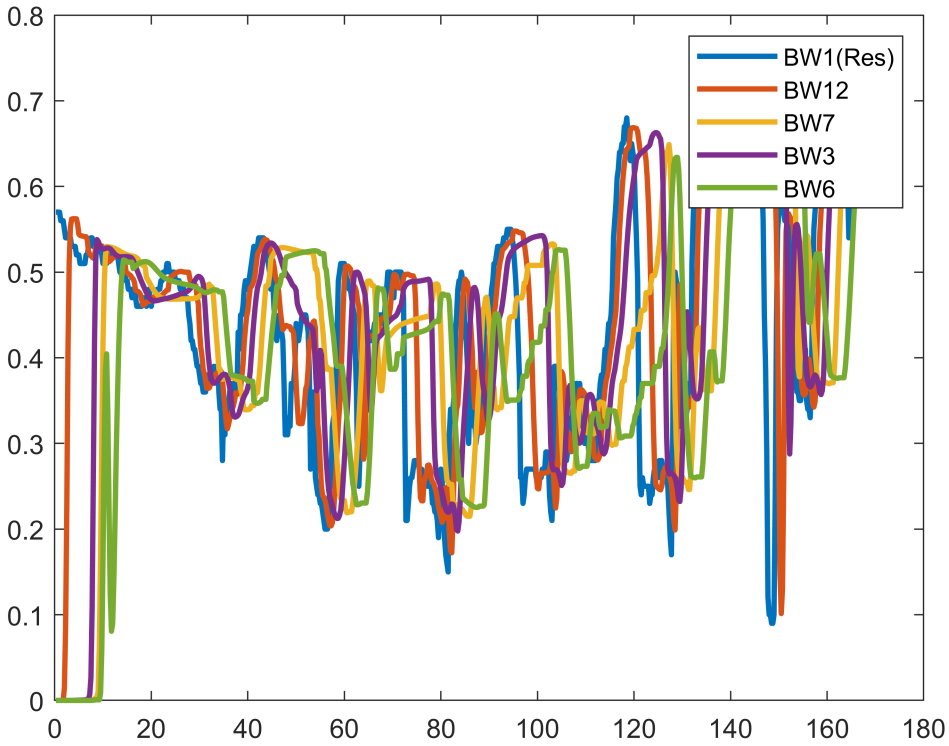
Simulate hydraulic and quality analyses using EPANET's solvers.

```
hydraulic_res = net.getComputedHydraulicTimeSeries;
quality_res = net.getComputedQualityTimeSeries;

% Assign hydraulic results to network elements
h = hydraulic_res.Head(1:1+4*nt,1:nn).';
q = 1e-3*hydraulic_res.Flow(1:1+4*nt,:).';

% Assign quality results to network elements
c_nodes = quality_res.NodeQuality';
c_pipes = quality_res.LinkQuality';

% Get the simulated and observed data
c_nodes_Sim = c_nodes(Junction_SelectIdx,:);
c_nodes_Sim = c_nodes_Sim(:,1:end-1);

c_nodes_HydrantSim = c_nodes(Junction_HydrantIdx,:);
c_nodes_HydrantSim = c_nodes_HydrantSim(:,1:end-1);

c_nodes_Reservoir = pattern_cext;
```

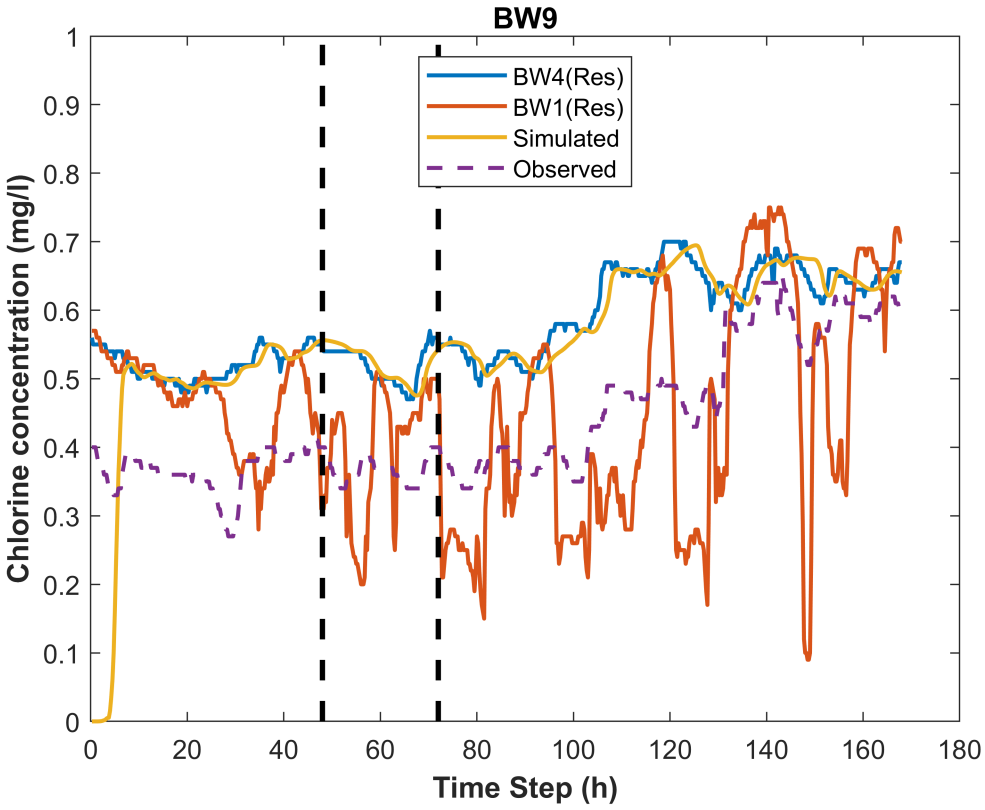
```
% Plot the simulated data without assigning decay coefficient
figure,
plot(0.25:0.25:days*24,c_nodes_Reservoir(1,:),'-','LineWidth',2)
hold on
plot(0.25:0.25:days*24,c_nodes_Sim([1,2],:),'-','LineWidth',2)
hold on
plot(0.25:0.25:days*24,c_nodes_Sim(3,:),'-','LineWidth',2)
legend(Junction_Reservoir_Name{1},Junction_Name{1},Junction_Name{2},...
    Junction_Name{3});
```

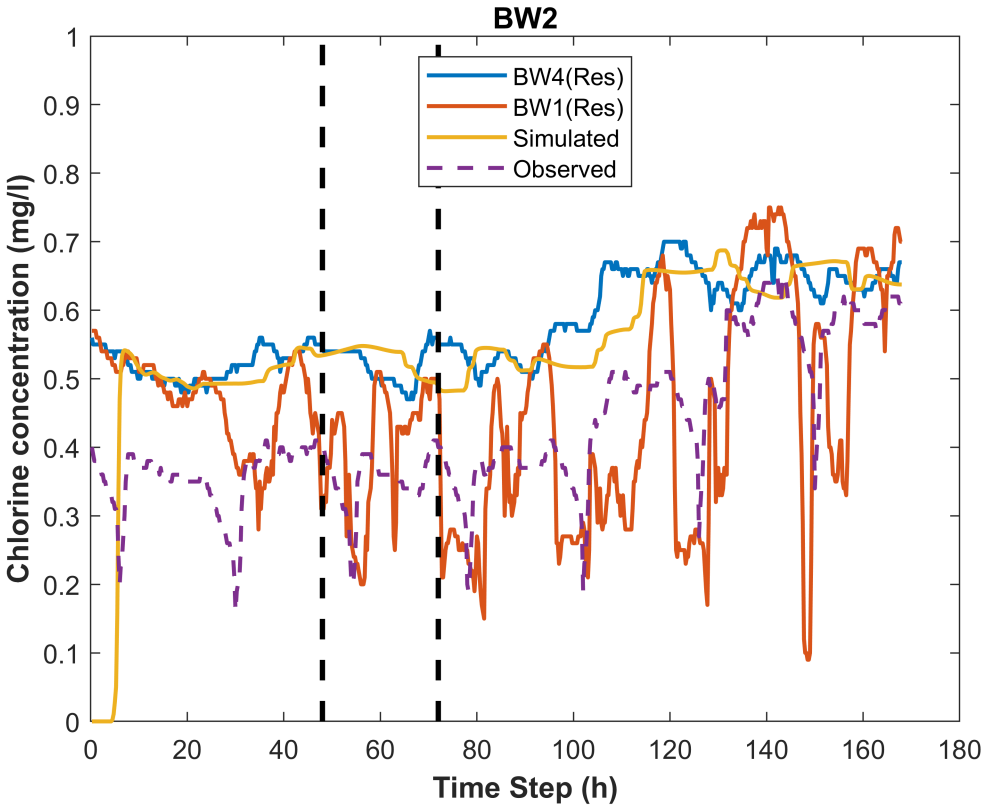


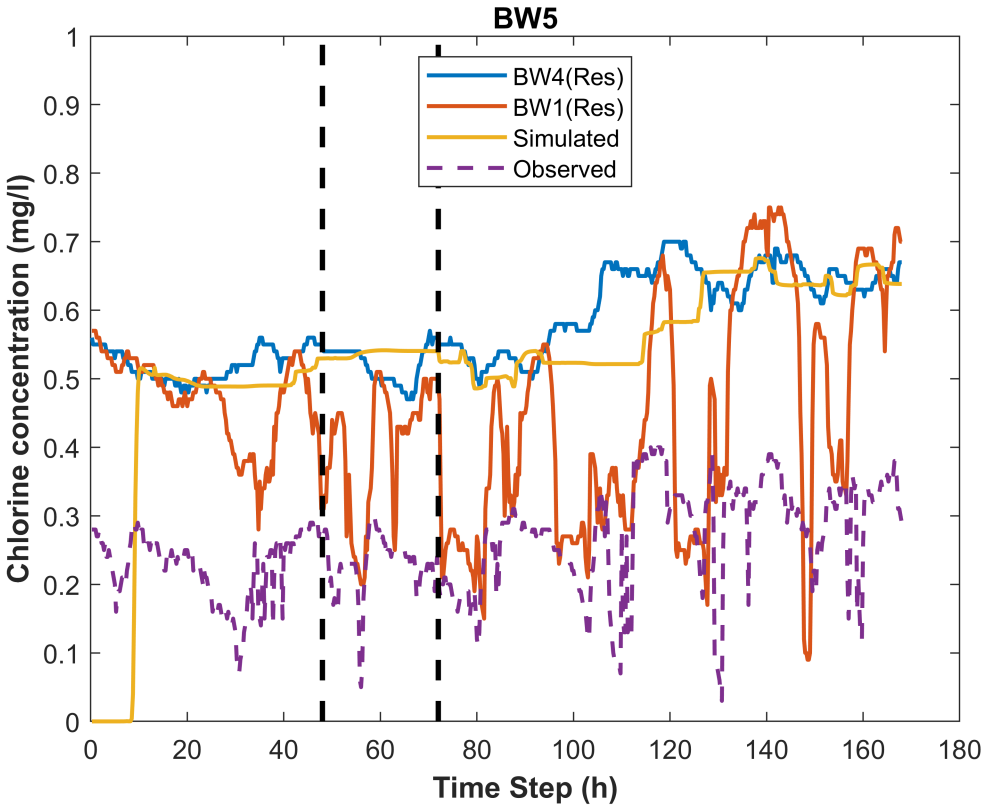

```
figure,
plot(0.25:0.25:days*24,c_nodes_Reservoir(2,:),'-','LineWidth',2)
hold on
plot(0.25:0.25:days*24,c_nodes_Sim(4,:),'-','LineWidth',2)
hold on
plot(0.25:0.25:days*24,c_nodes_HydrantSim([1,2],:),'-','LineWidth',2)
hold on
plot(0.25:0.25:days*24,c_nodes_HydrantSim(3,:),'-','LineWidth',2)

legend(Junction_Reservoir_Name{2},Junction_Name{4},Junction_Hydrant_Name{1},...
    Junction_Hydrant_Name{2},Junction_Hydrant_Name{3});
```

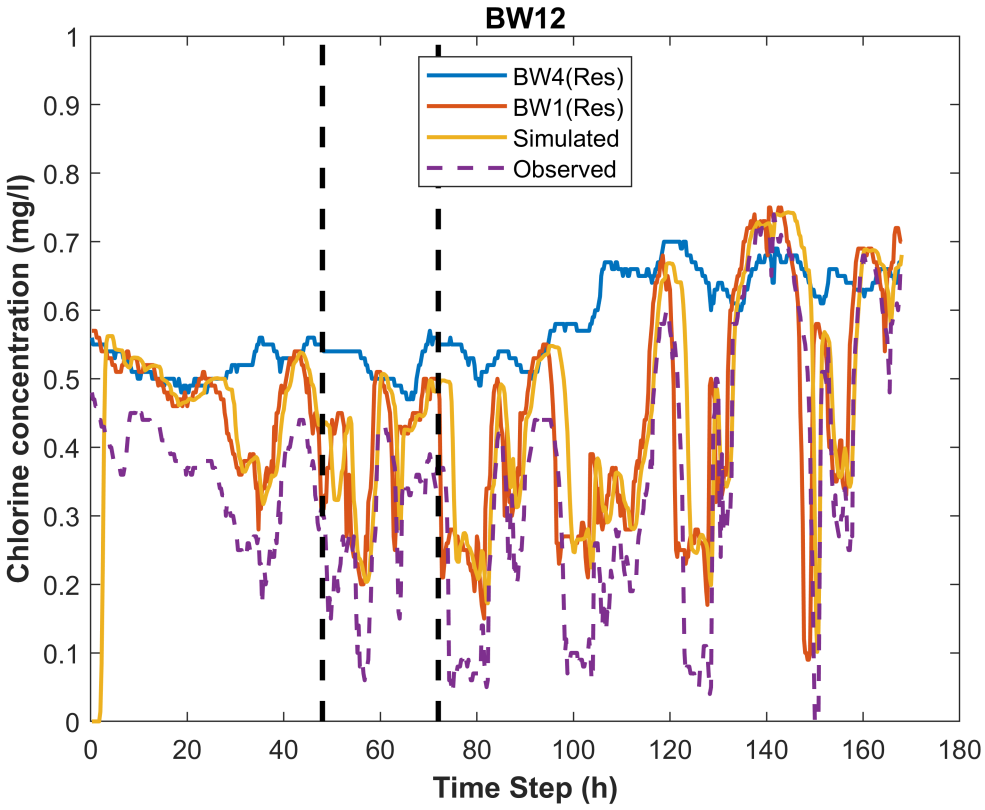



```
% Plot the simulated data and observed data and data of two reservoirs of 7 nodes
for i = 1:size(c_nodes_Obs,1)
    figure,
    plot(0.25:0.25:days*24,c_nodes_Reservoir,'-','LineWidth',1.5)
    hold on
    plot(0.25:0.25:days*24,c_nodes_Sim(i,:),'-','LineWidth',1.5)
    hold on
    plot(0.25:0.25:days*24,c_nodes_Obs(i,:),'LineStyle','--','LineWidth',1.5)
    hold on
    plot([48,48],[0,1],'Color','k','LineStyle','--','LineWidth',2)
    hold on
    plot([72,72],[0,1],'Color','k','LineStyle','--','LineWidth',2)
    legend([Junction_Reservoir_Name,{'Simulated','Observed'}],'location','best');
    xlabel('Time Step (h)','fontweight','bold')
    ylabel('Chlorine concentration (mg/l)','fontweight','bold')
    title(Junction_Name{i})
end
```


BW9
BW4(Res)
BW1(Res)
Simulated
Observed
Chlorine concentration (mg/l)
Time Step (h)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
20
40
60
80
100
120
140
160
180

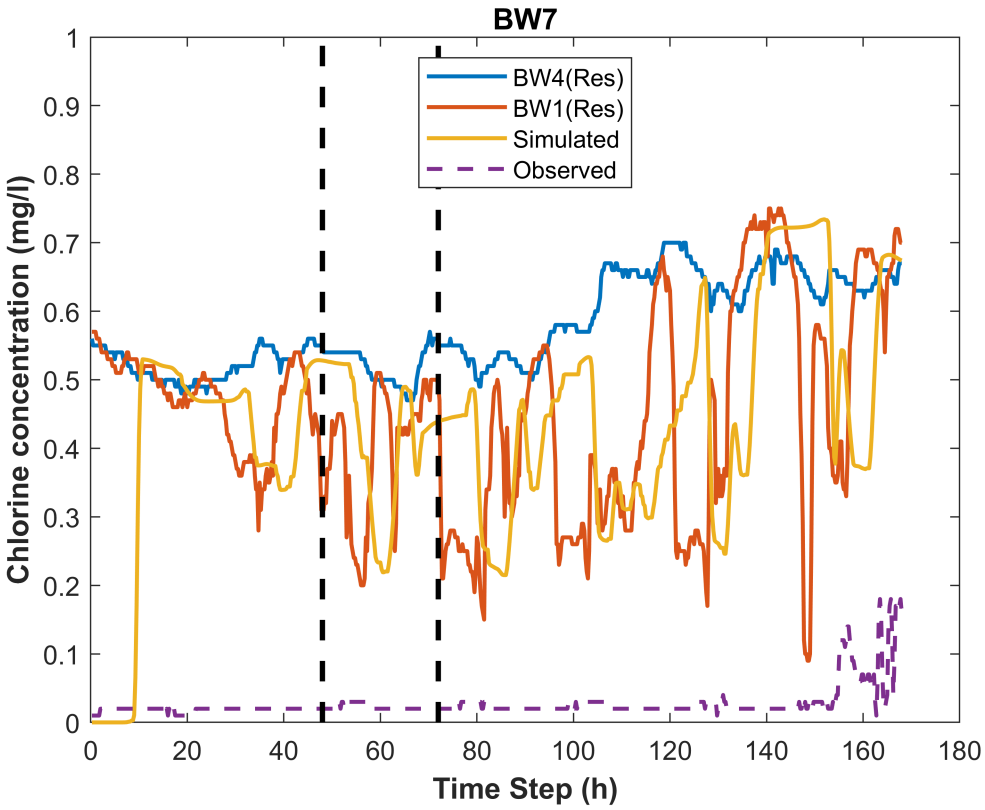

BW2
BW4(Res)
BW1(Res)
Simulated
Observed
Chlorine concentration (mg/l)
Time Step (h)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
20
40
60
80
100
120
140
160
180

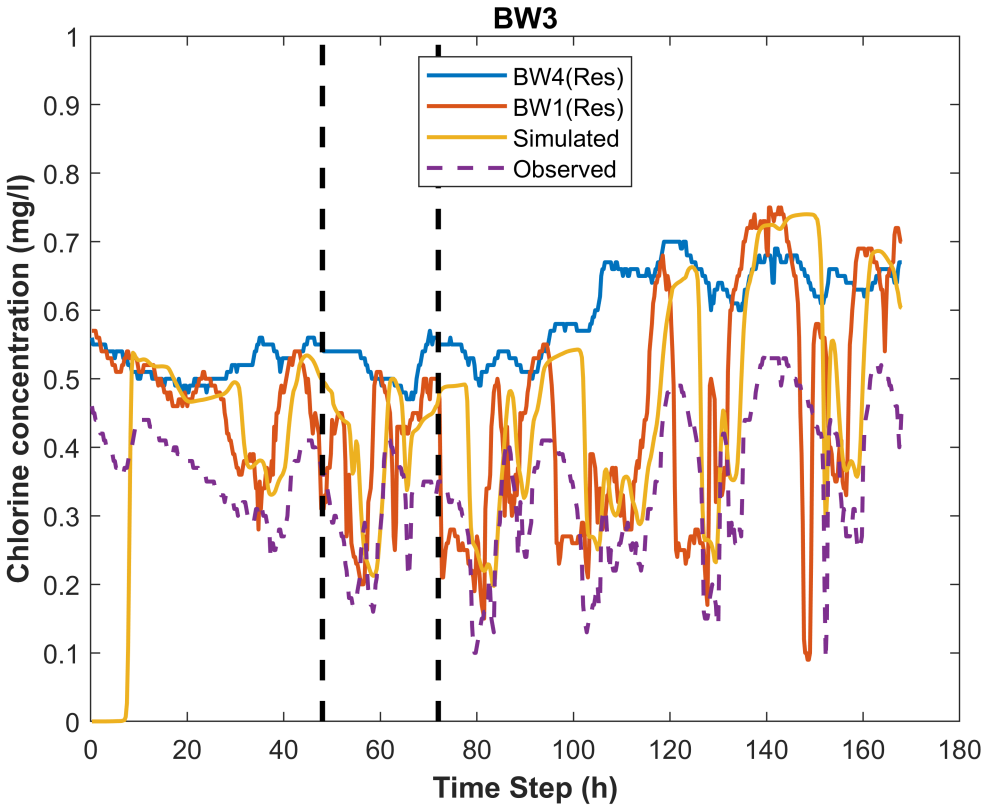


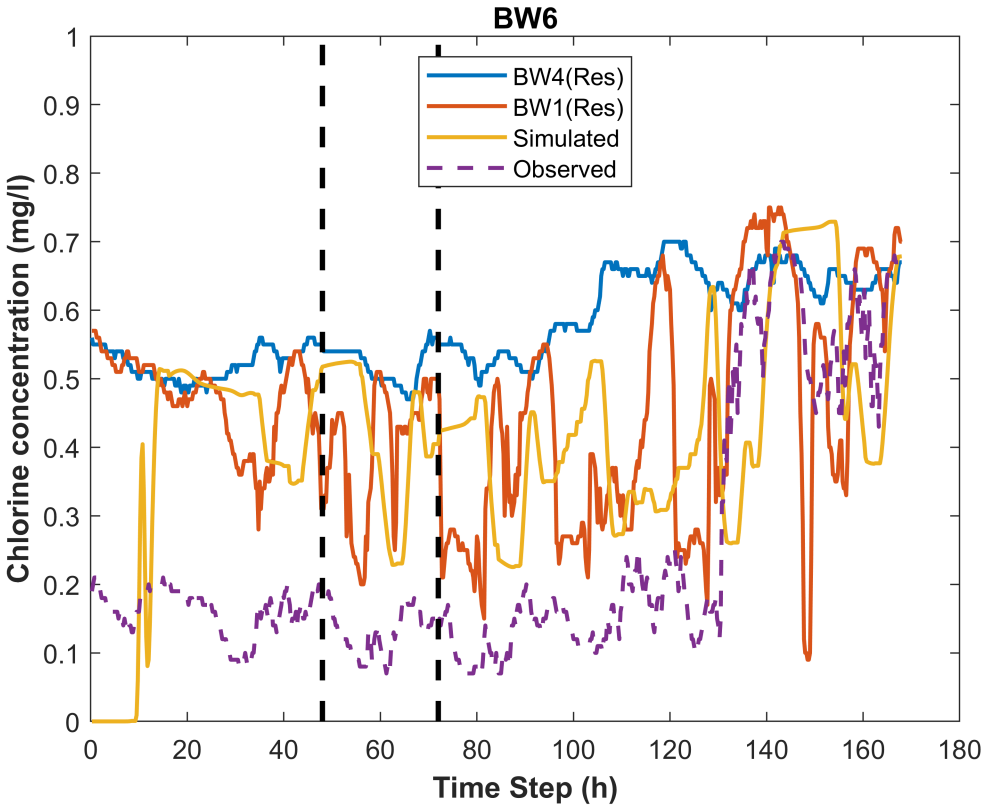




```
for i = 1:size(c_nodes_HydrantObs,1)
    figure,
```

```
    plot(0.25:0.25:days*24,c_nodes_Reservoir,'-','LineWidth',1.5)
    hold on
    plot(0.25:0.25:days*24,c_nodes_HydrantSim(i,:),'-','LineWidth',1.5)
    hold on
    plot(0.25:0.25:days*24,c_nodes_HydrantObs(i,:),'LineStyle','--','LineWidth',1.5)
    hold on
    plot([48,48],[0,1],'Color','k','LineStyle','--','LineWidth',2)
    hold on
    plot([72,72],[0,1],'Color','k','LineStyle','--','LineWidth',2)
    legend([Junction_Reservoir_Name,{'Simulated','Observed'}],'location','best');
    xlabel('Time Step (h)','fontweight','bold')
    ylabel('Chlorine concentration (mg/l)','fontweight','bold')
    title(Junction_Hydrant_Name{i})
end
```

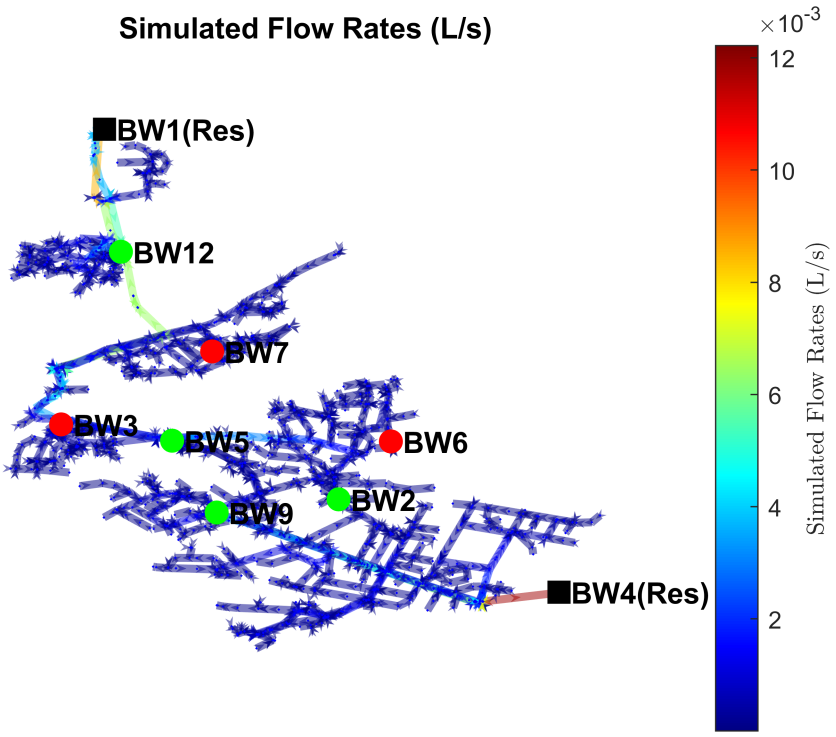





```
% Plot the flow direction
edgeweights = abs(q(:,1)); % t=1, steady state
```

```
A_Dir = [A12,A10];
AdjA_Dir = sparse(size(A_Dir,2),size(A_Dir,2));
for i = 1:size(A_Dir,1)
    u = find(A_Dir(i,:) == -1);
    v = find(A_Dir(i,:) == 1);
    edgemap(i,:) = [u v i];
    if isnan(edgeweights)
        AdjA_Dir(u,v) = 1;
    else
        AdjA_Dir(u,v) = edgeweights(i);
    end
end

edgemap = sortrows(edgemap);
G1 = digraph(AdjA_Dir);

figure,
p1 = plot(G1,'XData',XY(:,1),'YData',XY(:,2));
p1.ShowArrows = 'on';
p1.MarkerSize = 0.05;
p1.LineWidth = 3;
p1.NodeLabel = '';
axis('off')
p1.EdgeColor = 'k';
p1.NodeColor = 'b';
highlight(p1,Junction_SelectIdx,'NodeColor','green','MarkerSize',8);
labelnode(p1,Junction_SelectIdx,Junction_Name);
highlight(p1,Junction_HydrantIdx,'NodeColor','red','MarkerSize',8);
labelnode(p1,Junction_HydrantIdx,Junction_Hydrant_Name);
highlight(p1,Reservoir_Idx,'NodeColor','k','Marker','s','MarkerSize',10);
labelnode(p1,Reservoir_Idx,Junction_Reservoir_Name);
p1.NodeFontSize = 11;
p1.NodeLabelColor = 'k';
p1.NodeFontWeight = 'bold';
p1.EdgeFontSize = 11;
p1.EdgeLabelColor = 'k';
p1.EdgeFontWeight = 'bold';
p1.EdgeCData = G1.Edges.Weight;

hcb = colorbar;
hcb.Label.String = 'Simulated Flow Rates (L/s)';
hcb.Label.Interpreter = 'latex';

colormap("jet")

axis('off')
title('Simulated Flow Rates (L/s)')
```




## Water Age Simulation

Setup general water age simulation parameters.

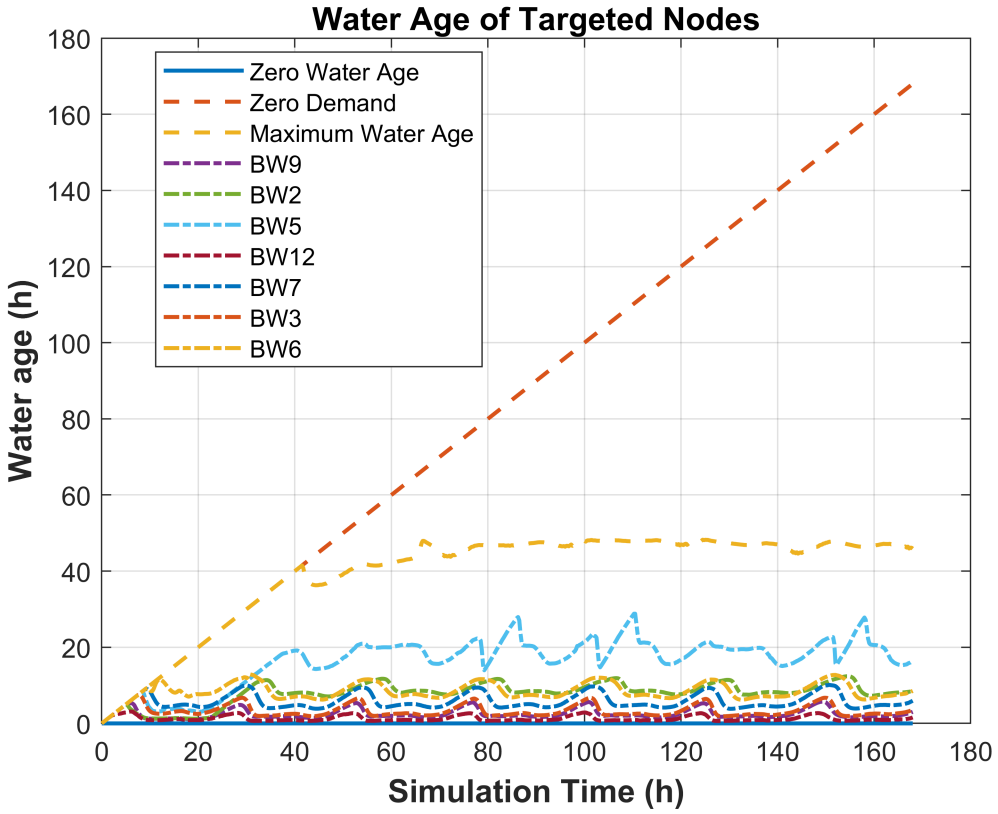
```
% Get Water age analysis
net.setQualityType('Age');
net.setNodeSourceQuality(1:nn+n0,zeros(nn+n0,1)); % set all node source quality to zero
quality_res_age = net.getComputedQualityTimeSeries;
c_nodes_age = quality_res_age.NodeQuality';
c_time_age = (quality_res_age.Time./3600)';
c_time_age = repmat(c_time_age,nn+n0,1);
```

Find the nodes with zero water age (Link/Valve closed) or the maximum water age (Age equals to simulation time, no demand).

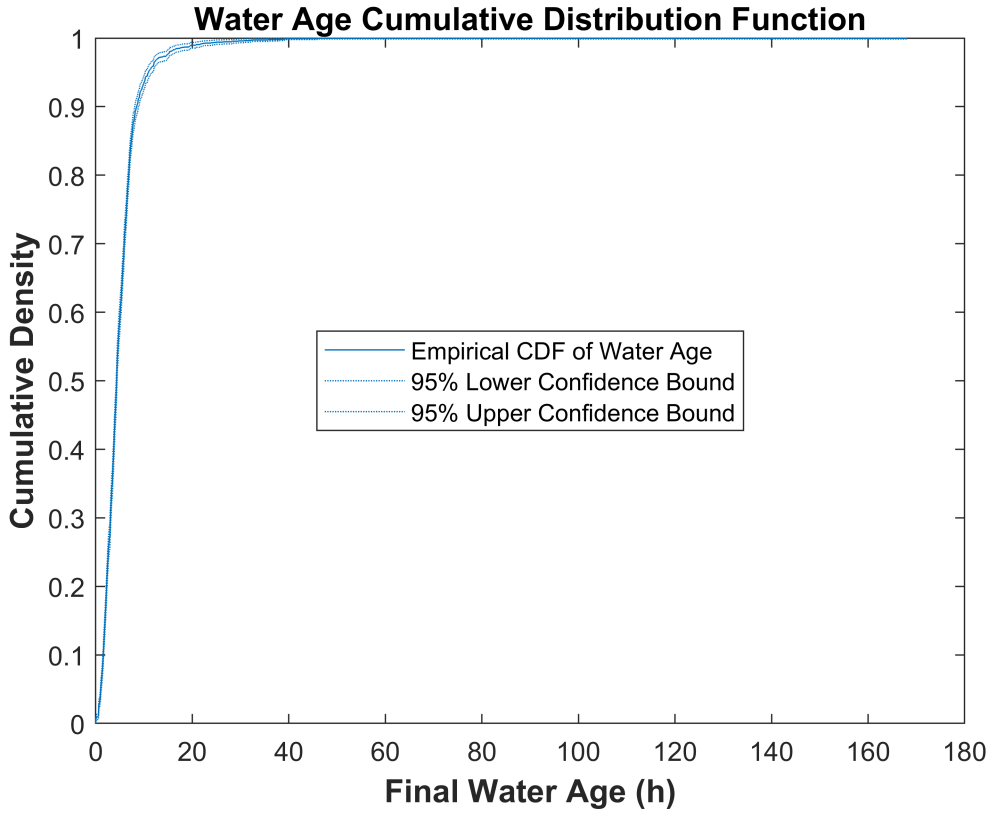
```
% Find the maximum water age of every node, if it equals to 168 (simulation time), there is no
% if it equals to 0, the node is closed or the node is a reservoir.
max_c_nodes_age = zeros(size(c_nodes_age,1),1);
for i = 1:size(c_nodes_age,1)
    max_c_nodes_age(i,1) = max(c_nodes_age(i,:));
end

% The node with zero water age
zero_age_c_nodeIdx = find(max_c_nodes_age == 0);
for i = 1:length(zero_age_c_nodeIdx)
    Junc_zeroage_Name{i} = sprintf('Node %d (Zero Age)',zero_age_c_nodeIdx(i));
```

```
end

% The node with maximum water age, not absolutely equals to 168, thus we add a
% constriant tolerance 0.01.
zerodemand_age_c_nodeIdx = find(abs(max_c_nodes_age - days*24) <= 0.01);
for i = 1:length(zerodemand_age_c_nodeIdx)
    Junc_zerodemand_Name{i} = sprintf('Node %d (Zero Demand)',zerodemand_age_c_nodeIdx(i));
end

% Fine the maximum water age before periodic behaviour
c_nodes_age_residual = abs(c_nodes_age - c_time_age);
c_nodes_period = zeros(nn+n0,1);
for i = 1:size(c_nodes_age_residual,1)
    for j = 1:size(c_nodes_age_residual,2)
        if c_nodes_age_residual(i,j) >= 0.01
            c_nodes_period(i,1) = j;
            break
        end
    end
end
max_age_c_nodeIdx = find(c_nodes_period == max(c_nodes_period));
for i = 1:length(max_age_c_nodeIdx)
    Junc_maxage_Name{i} = sprintf('Node %d (Max Age)',max_age_c_nodeIdx(i));
end
```

```
% Plot the node with zero age, zero demand and maximum age.
figure,
plot(0:0.25:days*24,c_nodes_age(zero_age_c_nodeIdx(1),:)','LineWidth',1.5)

hold on
plot(0:0.25:days*24,c_nodes_age(zerodemand_age_c_nodeIdx,:)', ...
        'LineStyle','--','LineWidth',1.5);
hold on
plot(0:0.25:days*24,c_nodes_age(max_age_c_nodeIdx,:)','LineStyle','--','LineWidth',1.5);
hold on
plot(0:0.25:days*24,c_nodes_age(Junction_SelectIdx,:)','LineStyle','-.','LineWidth',1.5);
hold on
plot(0:0.25:days*24,c_nodes_age(Junction_HydrantIdx,:)','LineStyle','-.','LineWidth',1.5);
% scatter(days*24,c_nodes_age(zero_age_c_nodeIdx(1),end),'red','Marker','o','LineWidth',1.5,'Si
% hold on
% scatter(days*24,c_nodes_age(zerodemand_age_c_nodeIdx,end),'green','Marker','square','LineWidt
% hold on
% scatter(days*24,c_nodes_age(max_age_c_nodeIdx,end),'blue','Marker','x','LineWidth',1.5,'SizeD

grid on
xlabel('Simulation Time (h)','FontWeight','bold',FontSize=12)
ylabel('Water age (h)','FontWeight','bold',FontSize=12)
title('Water Age of Targeted Nodes',FontSize=12)
legend(['Zero Water Age','Zero Demand','Maximum Water Age',Junction_Name,Junction_Hydrant_Name]
```




```
y = c_nodes_age(:,end);
x = linspace(min(y),max(y));
figure,
ecdf(y,'Bounds','on');
hold on
% plot(x,evcdf(x,0,3))
grid on

legend('Empirical CDF of Water Age','95% Lower Confidence Bound','95% Upper Confidence Bound',
hold off

xlabel('Final Water Age (h)','FontWeight','bold',FontSize=12)
ylabel('Cumulative Density','FontWeight','bold',FontSize=12)
title('Water Age Cumulative Distribution Function',FontSize=12)
% set(cdf_age,'Color','b','LineStyle','-.','LineWidth',1.5)
grid off
```




## Results Plotting

```
% Plot of network with zero demand junctions for visualisation
c_nodes_age_plot = c_nodes_age;
c_nodes_age_plot(zerodemand_age_c_nodeIdx,:) = zeros(1,size(c_nodes_age_plot,2));
figure
i = 673;
p1 = plot(gr);
p1.XData = XY(:,1);
p1.YData = XY(:,2);
p1.LineWidth = 1;
p1.EdgeColor = 'k';
p1.MarkerSize = (c_nodes_age_plot(:,i) + 5)./5;
p1.NodeColor = 'b';
p1.NodeLabel = '';
highlight(p1,zero_age_c_nodeIdx,'Marker','s','MarkerSize',10);
% for m = 1:length(zero_age_c_nodeIdx)
%     labelnode(p1,zero_age_c_nodeIdx(m),Junc_zeroage_Name{m});
% end
highlight(p1,zerodemand_age_c_nodeIdx,'Marker','s','MarkerSize',10);
% labelnode(p1,zerodemand_age_c_nodeIdx,Junc_zerodemand_Name);
% highlight(p1,max_age_c_nodeIdx,'MarkerSize',10);
% labelnode(p1,max_age_c_nodeIdx,Junc_maxage_Name);
% highlight(p1,max_age_c_nodeIdx);
% labelnode(p1,max_age_c_nodeIdx,Junc_maxage_Name);
p1.NodeFontSize = 9;
% p1.NodeLabelColor = [0.6350 0.0780 0.1840];
```

```
p1.NodeFontWeight = 'bold';
p1.NodeCData = c_nodes_age_plot(:,i);
hcb = colorbar;
colormap('jet');
hcb.Label.String = 'Final water age (h)';
hcb.Label.FontSize = 10;
hcb.Label.FontWeight = 'bold';
hcb.Location = 'southoutside';
title('The Final Water Age of BWFL Network')
axis('off')
```