APPENDIX C

(1) Mean values

```
% script file: thesis_neural
% This program is for calculating the mean values of each leak occurrence after 1,000 times of simulation.

% lamda = N(t0)* exp(A*t)- Mean value
% Nt0 = the number of leak in year to
% A_growth = growth rate coefficient
% t_optimal = THE YEAR 2004 - PIPE BUILT YEAR.

%Written by Juneseok Lee

% Nt0 = input('enter the Nt0 value: ');
% A_growth = input('enter the growth rate coeff: ');

%t_optimal = input ('enter the optimal t year(2004- Pipe Built Year): ');

lamda = 0.482; %Nt0 * exp(A_growth * t_optimal);



for iii = 1: 1000 % 1000 simulation

    %fprintf('\n\n')

    A = rand(1,3000); %create random number
    B = -log(A) / lamda; % create exponential r.v. - inter- arrival time

    for ii = 1:3000 % create the leak time.

        if ii == 1
            C(ii) = B(ii);

        else
            C(ii) = B(ii) + C(ii-1);

        end

    end
```

```
D = rand(1,3000); % create uniform r.v. again

for iiii = 1: 3000 % select the lambda(t) for each year

if C(iiii) <= 5
    LR(iiii) = 0.097;
elseif C(iiii) <= 10
    LR(iiii) = 0.104;
elseif C(iiii) <= 15
    LR(iiii) = 0.131;
elseif C(iiii) <= 20
    LR(iiii) = 0.214;
elseif C(iiii) <= 25
    LR(iiii) = 0.356;
elseif C(iiii) <= 30
    LR(iiii) = 0.458;
elseif C(iiii) <= 35
    LR(iiii) = 0.482;
else
    LR(iiii) = 0.483;
end

end


    for jj = 1: 3000 % compute the (lamda(t)/lamda).

        E(jj) = LR(jj) / lamda;

    end


    for kk = 1 : 3000 % using the probability of lamda(t)/ lamda, come up with C(kk).


        if E(kk) > D(kk)
            F(kk) = C(kk);
        else
            F(kk) = NaN;
```

```
        end

end

Fvalid = F(~isnan(F));



        temp = [Fvalid];
        inv_temp = temp';



        x_1(iii) = inv_temp(1);
        x_2(iii) = inv_temp(2);
        x_3(iii) = inv_temp(3);
        x_4(iii) = inv_temp(4);
        x_5(iii) = inv_temp(5);
        x_6(iii) = inv_temp(6);
        x_7(iii) = inv_temp(7);
        x_8(iii) = inv_temp(8);
        x_9(iii) = inv_temp(9);
        x_10(iii) = inv_temp(10);
        x_11(iii) = inv_temp(11);
        x_12(iii) = inv_temp(12);
        x_13(iii) = inv_temp(13);
        x_14(iii) = inv_temp(14);
        x_15(iii) = inv_temp(15);
        x_16(iii) = inv_temp(16);
        x_17(iii) = inv_temp(17);
        x_18(iii) = inv_temp(18);
        x_19(iii) = inv_temp(19);
        x_20(iii) = inv_temp(20);
        x_21(iii) = inv_temp(21);
        x_22(iii) = inv_temp(22);
        x_23(iii) = inv_temp(23);
        x_24(iii) = inv_temp(24);
        x_25(iii) = inv_temp(25);
        x_26(iii) = inv_temp(26);
        x_27(iii) = inv_temp(27);
        x_28(iii) = inv_temp(28);
```

```
%
end

    result_1 = mean([x_1]);
    result_2 = mean([x_2]);
    result_3 = mean([x_3]);
    result_4 = mean([x_4]);
    result_5 = mean([x_5]);
    result_6 = mean([x_6]);
    result_7 = mean([x_7]);
    result_8 = mean([x_8]);
    result_9 = mean([x_9]);
    result_10 = mean([x_10]);
    result_11 = mean([x_11]);
    result_12 = mean([x_12]);
    result_13 = mean([x_13]);
    result_14 = mean([x_14]);
    result_15 = mean([x_15]);
    result_16 = mean([x_16]);
    result_17 = mean([x_17]);
    result_18 = mean([x_18]);
    result_19 = mean([x_19]);
    result_20 = mean([x_20]);
    result_21 = mean([x_21]);
    result_22 = mean([x_22]);
    result_23 = mean([x_23]);
    result_24 = mean([x_24]);
    result_25 = mean([x_25]);
    result_26 = mean([x_26]);
    result_27 = mean([x_27]);
    result_28 = mean([x_28]);

fprintf('%f\n', result_1);
fprintf('%f\n', result_2);
fprintf('%f\n', result_3);
fprintf('%f\n', result_4);
fprintf('%f\n', result_5);
fprintf('%f\n', result_6);
fprintf('%f\n', result_7);
```

```
fprintf('%f\n', result_8);
fprintf('%f\n', result_9);
fprintf('%f\n', result_10);
fprintf('%f\n', result_11);
fprintf('%f\n', result_12);
fprintf('%f\n', result_13);
fprintf('%f\n', result_14);
fprintf('%f\n', result_15);
fprintf('%f\n', result_16);
fprintf('%f\n', result_17);
fprintf('%f\n', result_18);
fprintf('%f\n', result_19);
fprintf('%f\n', result_20);
fprintf('%f\n', result_21);
fprintf('%f\n', result_22);
fprintf('%f\n', result_23);
fprintf('%f\n', result_24);
fprintf('%f\n', result_25);
fprintf('%f\n', result_26);
fprintf('%f\n', result_27);
fprintf('%f\n', result_28);
```

(2) Distribution of each leak (Histogram generation)

```
% script file: thesis_neural
% the purpose of this program is to create the histogram for the distribution of leak occurrences

% lamda = N(t0)* exp(A*t)- Mean value
% USER ARE ASKED TO ENTER VALUES HERE BELOW.
% Nt0 = the number of leak in year to
% A_growth = growth rate coefficient
% t_optimal = THE YEAR 2004 - PIPE BUILT YEAR.

%Written by Juneseok Lee

% Nt0 = input('enter the Nt0 value: ');
% A_growth = input('enter the growth rate coeff: ');

%t_optimal = input ('enter the optimal t year(2004- Pipe Built Year): ');

lamda = 0.482; %Nt0 * exp(A_growth * t_optimal);



for iii = 1: 1000 % 1000 simulation

    %fprintf('\n\n')

    A = rand(1,3000); %create random number
    B = -log(A) / lamda; % create exponential r.v. - inter- arrival time

    for ii = 1:3000 % create the leak time.

        if ii == 1
            C(ii) = B(ii);

        else
            C(ii) = B(ii) + C(ii-1);

        end

    end
```

```
D = rand(1,3000); % create uniform r.v. again

for iiii = 1: 3000

if C(iiii) <= 5
    LR(iiii) = 0.097;
elseif C(iiii) <= 10
    LR(iiii) = 0.104;
elseif C(iiii) <= 15
    LR(iiii) = 0.131;
elseif C(iiii) <= 20
    LR(iiii) = 0.214;
elseif C(iiii) <= 25
    LR(iiii) = 0.356;
elseif C(iiii) <= 30
    LR(iiii) = 0.458;
elseif C(iiii) <= 35
    LR(iiii) = 0.482;
else
    LR(iiii) = 0.483;
end

end


    for jj = 1: 3000 % compute the (lamda(t)/lamda).

        E(jj) = LR(jj) / lamda;

    end


    for kk = 1 : 3000 % using the probability of lamda(t)/ lamda, come up with C(kk).


        if E(kk) > D(kk)
            F(kk) = C(kk);
        else
```

```
            F(kk) = NaN;
        end

end

Fvalid = F(~isnan(F));


        temp = [Fvalid];
        inv_temp = temp';


        x_1(iii) = inv_temp(1);
        x_2(iii) = inv_temp(2);
        x_3(iii) = inv_temp(3);
        x_4(iii) = inv_temp(4);
        x_5(iii) = inv_temp(5);
        x_6(iii) = inv_temp(6);
        x_7(iii) = inv_temp(7);
        x_8(iii) = inv_temp(8);
        x_9(iii) = inv_temp(9);
        x_10(iii) = inv_temp(10);
        x_11(iii) = inv_temp(11);
        x_12(iii) = inv_temp(12);
        x_13(iii) = inv_temp(13);
        x_14(iii) = inv_temp(14);
        x_15(iii) = inv_temp(15);
        x_16(iii) = inv_temp(16);
        x_17(iii) = inv_temp(17);
        x_18(iii) = inv_temp(18);
        x_19(iii) = inv_temp(19);
        x_20(iii) = inv_temp(20);
        x_21(iii) = inv_temp(21);
        x_22(iii) = inv_temp(22);
        x_23(iii) = inv_temp(23);
        x_24(iii) = inv_temp(24);
        x_25(iii) = inv_temp(25);
        x_26(iii) = inv_temp(26);
        x_27(iii) = inv_temp(27);
```

```
        x_28(iii) = inv_temp(28);
x= 0:2:100;
hist(x_11,x);

end
```

(3) Mode values for leaks

```
% script file: thesis_neural
% The purpose of this program is to find the mode values for the leak occurrences time

% lamda = N(t0)* exp(A*t)- Mean value
% USER ARE ASKED TO ENTER VALUES HERE BELOW.
% Nt0 = the number of leak in year to
% A_growth = growth rate coefficient
% t_optimal = THE YEAR 2004 - PIPE BUILT YEAR.

%Written by Juneseok Lee

%Nt0 = input('enter the Nt0 value: ');
%A_growth = input('enter the growth rate coeff: ');

%t_optimal = input ('enter the optimal t year(2004- Pipe Built Year): ');

lamda = 0.482; %Nt0 * exp(A_growth * t_optimal);



for iii = 1: 1000 % 1000 simulation

    %fprintf('\n\n')

    A = rand(1,3000); %create random number
    B = -log(A) / lamda; % create exponential r.v. - inter- arrival time

    for ii = 1:3000 % create the leak time.

        if ii == 1
            C(ii) = B(ii);

        else
            C(ii) = B(ii) + C(ii-1);

        end

    end
```

```
D = rand(1,3000); % create uniform r.v. again

for iiii = 1: 3000

if C(iiii) <= 5
    LR(iiii) = 0.097;
elseif C(iiii) <= 10
    LR(iiii) = 0.104;
elseif C(iiii) <= 15
    LR(iiii) = 0.131;
elseif C(iiii) <= 20
    LR(iiii) = 0.214;
elseif C(iiii) <= 25
    LR(iiii) = 0.356;
elseif C(iiii) <= 30
    LR(iiii) = 0.458;
elseif C(iiii) <= 35
    LR(iiii) = 0.482;
else
    LR(iiii) = 0.483;
end

end



    for jj = 1: 3000 % compute the (lamda(t)/lamda).

        E(jj) = LR(jj) / lamda;

    end



    for kk = 1 : 3000 % using the probability of lamda(t)/ lamda, come up with C(kk).

        if E(kk) > D(kk)
            F(kk) = C(kk);
        else
```

```
            F(kk) = NaN;
        end

end

Fvalid = F(~isnan(F));


        temp = [Fvalid];
        inv_temp = temp';


        x_1(iii) = int8(inv_temp(1));
        x_2(iii) = int8(inv_temp(2));
        x_3(iii) = int8(inv_temp(3));
        x_4(iii) = int8(inv_temp(4));
        x_5(iii) = int8(inv_temp(5));
        x_6(iii) = int8(inv_temp(6));
        x_7(iii) = int8(inv_temp(7));
        x_8(iii) = int8(inv_temp(8));
        x_9(iii) = int8(inv_temp(9));
        x_10(iii) = int8(inv_temp(10));
        x_11(iii) = int8(inv_temp(11));
        x_12(iii) = int8(inv_temp(12));
        x_13(iii) = int8(inv_temp(13));
        x_14(iii) = int8(inv_temp(14));
        x_15(iii) = int8(inv_temp(15));
        x_16(iii) = int8(inv_temp(16));
        x_17(iii) = int8(inv_temp(17));
        x_18(iii) = int8(inv_temp(18));
        x_19(iii) = int8(inv_temp(19));
        x_20(iii) = int8(inv_temp(20));
        x_21(iii) = int8(inv_temp(21));
        x_22(iii) = int8(inv_temp(22));
        x_23(iii) = int8(inv_temp(23));
        x_24(iii) = int8(inv_temp(24));
        x_25(iii) = int8(inv_temp(25));
        x_26(iii) = int8(inv_temp(26));
        x_27(iii) = int8(inv_temp(27));
```

```
            x_28(iii) = int8(inv_temp(28));

%
end

      result_1 = mode([x_1]);
      result_2 = mode([x_2]);
      result_3 = mode([x_3]);
      result_4 = mode([x_4]);
      result_5 = mode([x_5]);
      result_6 = mode([x_6]);
      result_7 = mode([x_7]);
      result_8 = mode([x_8]);
      result_9 = mode([x_9]);
      result_10 = mode([x_10]);
      result_11 = mode([x_11]);
      result_12 = mode([x_12]);
      result_13 = mode([x_13]);
      result_14 = mode([x_14]);
      result_15 = mode([x_15]);
      result_16 = mode([x_16]);
      result_17 = mode([x_17]);
      result_18 = mode([x_18]);
      result_19 = mode([x_19]);
      result_20 = mode([x_20]);
      result_21 = mode([x_21]);
      result_22 = mode([x_22]);
      result_23 = mode([x_23]);
      result_24 = mode([x_24]);
      result_25 = mode([x_25]);
      result_26 = mode([x_26]);
      result_27 = mode([x_27]);
      result_28 = mode([x_28]);


            %fprintf('%f\n', B);

fprintf('%f\n', result_1);
fprintf('%f\n', result_2);
```

```
fprintf('%f\n', result_3);
fprintf('%f\n', result_4);
fprintf('%f\n', result_5);
fprintf('%f\n', result_6);
fprintf('%f\n', result_7);
fprintf('%f\n', result_8);
fprintf('%f\n', result_9);
fprintf('%f\n', result_10);
fprintf('%f\n', result_11);
fprintf('%f\n', result_12);
fprintf('%f\n', result_13);
fprintf('%f\n', result_14);
fprintf('%f\n', result_15);
fprintf('%f\n', result_16);
fprintf('%f\n', result_17);
fprintf('%f\n', result_18);
fprintf('%f\n', result_19);
fprintf('%f\n', result_20);
fprintf('%f\n', result_21);
fprintf('%f\n', result_22);
fprintf('%f\n', result_23);
fprintf('%f\n', result_24);
fprintf('%f\n', result_25);
fprintf('%f\n', result_26);
fprintf('%f\n', result_27);
fprintf('%f\n', result_28);
```